АВТОКАД 10

КУРС
ПРАКТИЧЕСКОЙ
РАБОТЫ

С.ГЛАДКОВ, Ю.КРЕЧКО,
К.МОЛОДЦОВ, В.ПОЛИЩУК, Г.СУЧКОВ

КУРС ПРАКТИЧЕСКОЙ РАБОТЫ С СИСТЕМОЙ АВТОКАД 10

МОСКВА
■ "ДИАЛОГ-МИФИ" ■
1991

ББК 32.9
К78

К78 **Курс практической работы с системой Автокад 10**: Учебное пособие/ Гладков С.А., Кречко Ю.А., Молодцов К.И., Полищук В.В., Сучков Г.А.—М.: "ДИАЛОГ-МИФИ", 1991.—288 с.

ISBN 5-86404-008-8

Оригинальное учебное пособие по курсу практической работы с пакетом Автокад®, версия 10 (рассматриваются как английский, так и русифицированный варианты). Описаны команды системы Автокад, приемы работы с ними. Приводится описание языка программирования Автолисп® и примеры его использования при модификации меню Автокада и для создания собственных команд.

Методика изложения материала успешно используется при обучении различных категорий специалистов в Учебном центре акционерного общества "ДИАЛОГ-МИФИ".

ББК 32.9

ISBN 5-86404-008-8

Оригинал-макет данного издания подготовлен с помощью текстового редактора Microsoft Word 5.0 и отпечатан на лазерном принтере HP LaserJet series II. Иллюстративный материал подготовлен с помощью системы Автокад 10 и выполнен на плоттере Hewlett-Packard, модель 7475.

Уважаемые читатели, к моменту выхода настоящего издания в свет Научно-производственный и учебно-консультационный центр совместного советско-американского предприятия "Диалог" на базе Московского инженерно-физического института (Центр МИФИ СП "Диалог") преобразован в акционерное общество "ДИАЛОГ-МИФИ", которое является полным правопреемником Центра МИФИ СП "Диалог".

ВВЕДЕНИЕ

Автокад® представляет собой прикладную систему автоматизации чертежно-графических работ с удобными и эффективными средствами исправления допускаемых в ходе работы ошибок. Название системы образовано от сокращенного английского словосочетания "Automated Computer Aided Drafting and Design", что в переводе с английского означает "Автоматизированное черчение и проектирование с помощью ЭВМ" и является в некотором смысле эквивалентом понятия "программная система автоматизированного проектирования".

Системы автоматизированного проектирования (САПР) - признанная область применения вычислительной техники. Компьютер может предоставить конструкторам и технологам полный набор возможностей САПР и, освободив их от рутинной работы, дать возможность заниматься творчеством, что резко повышает производительность труда.

Приближение САПР к конструктору позволило резко повысить производительность самих САПР, распространение которых сдерживалось трудностью алгоритмизации конструкторских задач. Действительно, невозможно к каждому конструктору "приставить" программиста. Это противоречие может быть устранено только широким распространением прикладных программных средств, общающихся с конструктором на "естественном" языке. Следует отметить, что это справедливо не только для области компьютерной графики. Практически все современное программное обеспечение ориентируется на пользователя, дружелюбно общаясь с ним понятным ему способом и предоставляя ему полную свободу действий. Такое "общение" человека с компьютером возможно только в интерактивном (диалоговом) режиме, когда пользователь тут же на экране видит результат своих действий. САПР также ориентированы на работу в интерактивном режиме, предоставляя проектировщику оперативный доступ к графической информации, простой и эффективный язык управления ее обработкой с практически неограниченными возможностями контроля результатов. В первую очередь это относится к графическому диалогу, поскольку именно графика (чертежи, схемы, диаграммы и т.п.) как наиболее эффективный способ представления информации занимает привилегированное положение в САПР. Таким образом удается автоматизировать самую трудоемкую часть работы - по оценкам зарубежных конструкторских бюро, в процессе традиционного проектирования на разработку и оформление чертежей приходится около 70% от общих трудозатрат конструкторской работы (ср.: 15% - на организацию и ведение архивов, и 15% - собственно на проектирование, включающее в себя разработку конструкции, расчеты, согласование со смежными областями и т.д.).

Ввод графической информации осуществляется как с клавиатуры, так и считыванием значений координат с экрана графического дисплея или планшета. Непосредственное отображение на экране всего чертежа или его части создает привычную атмосферу работы руками и позволяет осуществлять редактирование

изображения и эффективное управление процессом проектирования. Многие современные программные системы, ориентированные на проектирование промышленных изделий, имеют достаточно большой арсенал возможностей интерактивной графики, обеспечивая возможность создания и редактирования двумерных изображений, состоящих из проекций изделия, штриховки, размеров и т.д., а также формирования реалистичных трехмерных изображений проектируемых изделий, построенных из исходных данных чертежа с удалением невидимых линий, с учетом различных способов освещения, задания параметров структуры поверхностей и т.п. При этом САПР предоставляют невиданные и принципиально недостижимые ранее возможности. Фактически конструктор попадает в новую среду - среду компьютерной графики. И качество пакета САПР едва ли не в первую очередь определяется тем, насколько труден для конструктора переход к новой технологии при использовании того или иного пакета.

В настоящее время существует огромное количество САПР различной сложности и назначения. Очевидно, что пользователь будет выбирать систему, согласовывая необходимость графических возможностей со стоимостью системы и технических средств, которые обладают требуемыми возможностями. Так, например, стоимость АРМ (автоматизированное рабочее место - "workstation") Apollo или SUN-2, обладающих всеми мыслимыми на сегодняшний день возможностями, существенно выше стоимости любого персонального компьютера - это просто другой класс машин. Для большинства чертежно - конструкторских работ требуются более скромные, однако все же достаточно широкие возможности, и ряд систем способен их удовлетворить. Такие системы, как CADDS3 или DDM позволяют создавать в интерактивном режиме каркасные, так называемые "проволочные", модели изделий, а также одновременно получать несколько проекций на экранном поле (как с использованием проекционной связи, так и без нее) и манипулировать каждой проекцией с соответствующими изменениями на остальных проекциях. Примерно такими же возможностями обладают французские системы GRAFIXI, CATIA, западногерманская система COMPAC, английская система ROMULUS, японская система TIPS-1, система фирмы "Сименс" CADIS, западноберлинская система GEOMETRIE, система EUCLID фирмы Matra Datavision и другие. Системы отличаются друг от друга ориентацией на те или иные геометрические построения, способом работы с изображениями и ведения диалога с системой.

Фирма Autodesk™ является одним из признанных лидеров в области разработки систем САПР. Созданный ею пакет Автокад является одним из лучших - это сложная и разветвленная по своей структуре система, которая в то же время легко управляется при помощи простых и ясных команд. Эта система дает пользователю микрокомпьютера возможности, ранее доступные только на больших и дорогих вычислительных системах. Автокад обладает эффективной системой ведения диалога с пользователем при помощи нескольких меню: главного, экранного, падающих и т.п. Использование слоев также предоставляет дополнитель

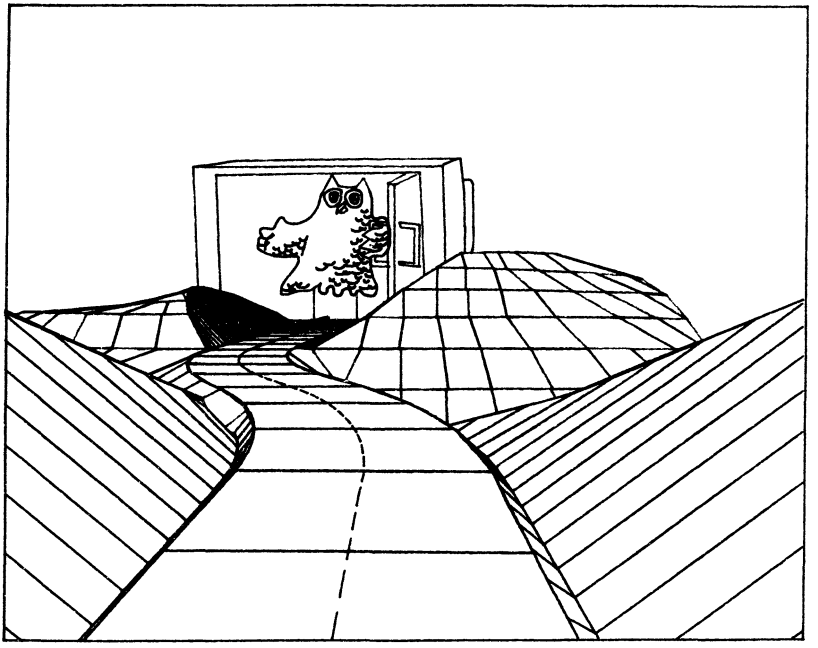
ные удобства для проектировщика, позволяя при наложении слоев с нарисованными на них изображениями отдельных деталей контролировать их совместимость при общей компоновке, а также держать "про запас" любое число различных вариантов деталей и, включая либо выключая слои, выборочно вводить их в общую компоновку. Законченные чертежи можно хранить в виде комплекта слайдов с возможностью их автоматического просмотра, причем доступность большого количества цветов (до 255) делает работу с такой системой эстетически приятной.

Кроме автоматизации собственно чертежно-графических работ, Автокад с его расширениями (AutoShade®, AutoFlix®, GLISP, ABASE) предоставляет следующие принципиально новые возможности:

- графическое моделирование, т.е. использование компьютера в САПР в качестве мощного вычислительного средства, позволяющего без особых навыков программирования работать со сложными пространственными моделями
- создание и ведение информационной базы данных (архива) чертежей
- создание библиотеки стандартных элементов чертежей, относящихся к какой-то предметной области, с тем чтобы строить новые чертежи из уже созданных ранее элементов
- параметризация чертежей - построение деталей и чертежей с новыми размерами на основе один раз нарисованного чертежа (модели)
- создание демонстрационных иллюстраций и мультфильмов

Фирма Autodesk на протяжении нескольких лет совершенствует систему Автокад - в настоящее время существует несколько версий, отличающихся своими функциональными возможностями. Все они совместимы "снизу вверх", т.е. чертежи, созданные на ранних версиях, обрабатываются на более поздних версиях Автокада. Несмотря на то, что в основу пособия положена работа с версией 10, оно может быть использовано как для изучения более ранних версий, так и для изучения версии 11, коммерческую версию которой фирма Autodesk предполагает выпустить в конце 1990 года.

Настоящий курс рассчитан на широкий круг специалистов и раскрывает некоторые тонкости практического использования Автокада. Научиться работать с системой Автокад достаточно просто. В зависимости от квалификации пользователя и его способностей Автокад может эффективно использоваться для решения очень широкого круга задач: проектирования, конструирования, черчения, оформительских работ, создания мульт- и слайдфильмов и многого другого.



ГЛАВА 1

ОБЩИЕ СВЕДЕНИЯ

Автокад - это мощный универсальный пакет САПР. Автокад не предназначен только для машиностроителя или только для архитектора, это инструмент любого специалиста, работающего с технической графикой. Фирма Autodesk, ориентируясь на самый широкий круг пользователей, заложила в пакет широкие возможности настройки Автокада на любую предметную область. Именно поэтому Автокад завоевал широкую популярность и продолжает сохранять свои позиции, несмотря на все более и более жесткую конкуренцию. Однако адаптацию Автокада для решения конкретно определенного на предприятии круга задач может выполнить только достаточно опытный пользователь.

Такая адаптация, включающая в себя создание блоков и слайдов, написание программ на Автолиспе, модификацию меню и т.п., потребует некоторого времени и усилий. Поэтому, как правило, на производстве такая работа выполняется специально занимающимся этими вопросами человеком. Именно он готовит систему, проводит обучение специалистов, которые будут применять ее в своей деятельности, и модифицирует систему по их замечаниям. Такая организация труда позволяет быстро и эффективно применять Автокад на практике и не тратить время на обучение людей тому, чем они никогда не будут пользоваться.

И все же есть вещи, которые должен знать каждый, кто собирается пользоваться Автокадом. Нужно знать, как войти в систему, как загрузить ранее созданный рисунок, как создать новый и отредактировать его, а затем сохранить на диске, какие имеются возможности работы со слоями и цветом и т.п. Именно эти вопросы будут рассматриваться в последующих главах.

1.1. Требования к оборудованию и краткие сведения об установке пакета

Для работы на ПК, совместимых с компьютерами фирмы IBM (PC-XT, или PC-AT, или PS/2 моделей 50, 60 или 80), с Автокадом версии 10 требуется:

- электронный замок
- оперативная память не менее 640 Кбайт; для работы с расширенным Автолиспом требуется по крайней мере 512 Кбайт дополнительной памяти типа IBM AT extended memory (*не* типа Lotus/Intel/Microsoft expanded memory), не занятой для других целей (например, виртуальный диск)
- жесткий диск (не менее 30 Мбайт)
- сопроцессор обработки чисел с плавающей точкой (8087 или 80287)
- операционная система PC-DOS/MS-DOS версии не ниже 2.0 (для PS/2 - не ниже 3.3)

- параллельный адаптер связи типа Centronics для подключения электронного замка
- наличие последовательного порта для дигитайзера и некоторых плоттеров
- графопостроитель или принтер с возможностью вывода графики (необязательно)
- чтобы система поддерживала работу с графическими адаптерами CGA, EGA, IBM 8514, VGA, Hercules Graphics Card и многими другими
- устройство указания (дигитайзер) - манипулятор "мышь" или цифровой планшет. Любое из этих устройств обеспечивает возможность быстрого и удобного ввода команд

Для работы системы Автокад требуется электронный замок - маленькая коробочка с электрическими разъемами спереди и сзади. Замок поставляется вместе с дискетами и документацией. Купленная версия Автокада не будет работать без замка, правильно подключенного к вашему компьютеру.

Автокад поддерживает большое количество различных типов мониторов и периферийных устройств. Указания по их установке, а также другие сведения по конфигурации системы приводятся в одной из книг фирменной документации: "Руководстве по установке и эксплуатации системы Автокад".

В комплект поставки русской версии Автокада не входит, в отличие от английской версии, программа установки системы Автокад на жестком диске. Такую установку предлагается осуществить вручную, скопировав все файлы Автокада в один подкаталог с именем ACAD10. Как показывает практика, это не очень удобно. Мы рекомендуем хранить файлы различных типов раздельно, как показано ниже (предлагаемая структура каталога несколько отличается от предлагаемой фирмой):

Структура каталога	*Файлы*	*Назначение файлов*
C:\ACAD10	acad.bat	Командный файл запуска Автокада (описан ниже)
	acad.cfg	Файл конфигурации Автокада
		Выполняемые файлы Автокада:
	acad.exe	главный выполняемый файл
	extlisp.exe	расширенный Автолисп
	remlisp.exe	разгрузчик Автолиспа
	*.ovl:	Оверлейные файлы Автокада:
	acad.ovl	главный оверлейный файл
	acad0.ovl	главный оверлейный файл
	acad2.ovl	файл к ADE-3
	acad3.ovl	файл к ADE-3
	acadl.ovl	файл Автолиспа к ADE-3
	acadlx.ovl	к расширенному Автолиспу
	acadvs.ovl	оверлейный файл сообщений

		Конфигурационные оверлеи:
	acadm.ovl	аппаратных настроек
	acadds.ovl	монитора
	acaddg.ovl	дигитайзера
	acadpp.ovl	принтера
	acadpl.ovl	плоттера
-DWG	*.dwg	Файлы чертежей Автокада
-DRV	*.drv	Файлы драйверов устройств
-LSP	*.lsp	Исходные тексты программ на Автолиспе
-SOURCE		Файлы адаптации Автокада:
	acad.mnd	файл описания меню Автокада
	acad.mnu	вторичный файл описания меню Автокада
	*.shp	исходные файлы форм и шрифтов Автокада
-SUPPORT		Служебные файлы и утилиты:
	mc.exe	компилятор меню
	acad.mnx	скомпилированный файл меню Автокада
	acad.hlp	файл справок (помощи)
	acad.pat	файл стандартных типов штриховок
	acad.lin	файл определения типов линий
	acad.pgp	файл определений внешних команд
	slidelib.exe	библиотекарь слайдов
	acad.slb	стандартная библиотека слайдов Автокада
	*.sld	слайды Автокада
	*.shx	скомпилированные файлы форм и шрифтов Автокада

Поскольку для нормальной работы Автокада он должен иметь доступ ко всем файлам пакета, желательно запускать Автокад через командный файл, в котором следует определить путь доступа во все подкаталоги. Кроме того, если вы работаете с расширенным Автолиспом, то перед запуском программы acad.exe следует загрузить в память расширенный Автолисп (запустить на выполнение программу extlisp.exe). Ниже приводится текст командного файла acad.bat, соответствующего приведенной в таблице структуре каталогов (считаем, что каталог ACAD10 находится на устройстве C):

```
rem Отключение эха (на ДОС версии 3.3 и выше);
@echo off
rem Очистка экрана и вывод сообщения;
cls
echo Автокад загружается, ждите...
rem Переход на устройство, на котором находится Автокад;
C:
rem Переход в каталог Автокада;
cd ACAD10
rem Установление в переменной ac пути к каталогу ACAD10;
set ac=C:\ACAD10
rem Установление пути к подкаталогам Автокада;
path %ac%;%ac%\DWG;%ac%\DRV;%ac%\LSP;%ac%\SOURCE;
```

```
%ac%\SUPPORT
rem Установление параметров для Автолиспа (необязательно);
set lispstack=5000
set acadxmem=1024k,512k
rem Загрузка Автолиспа;
extlisp>nul
rem Загрузка Автокада;
acad
```

Теперь для того, чтобы начать работу с Автокадом, нужно вызвать на выполнение командный файл acad.bat.

1.2. Основные понятия

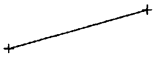
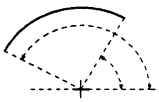
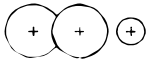
В данном разделе будут рассмотрены общие понятия, которые встретятся в процессе работы с системой Автокад.

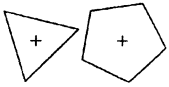
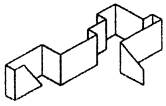
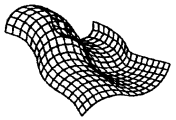
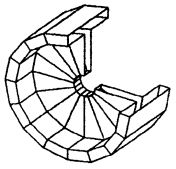
1.2.1. Примитивы

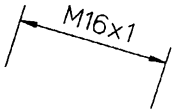
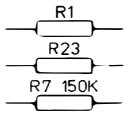
В отличие от "художественных" графических редакторов (например, Paintbrush) Автокад работает не с изображением как таковым, а с геометрическим описанием объектов, составляющих изображение, что обусловлено задачами САПР. Так, например, отрезок во внутреннем представлении графического редактора Автокада описывается двумя точками, круг описывается центром и радиусом и т.п. У такого представления есть как преимущества, так и недостатки. С одной стороны, графическое представление изделия с его геометрическим описанием более компактно и, что особенно важно, позволяет производить различные геометрические преобр ..зания (которые постоянно требуются в инженерной практике), а также напрямую использовать такое описание в автоматизированных системах технологической подготовки производства. С другой стороны, этот способ, на первый взгляд, сильно ограничивает наши возможности - мы не можем рисовать, скажем, кривые произвольной формы. Однако, как показывает практика, геометрического представления достаточно для любого технического изображения. Кроме того, как вы убедитесь сами, из простых элементов можно создать практически любое изображение.

Все примитивы Автокада обладают рядом свойств, (принадлежность слою, цвет, тип линии, ширина и т.п.). Некоторые из этих свойств (например, цвет) присущи всем примитивам; есть ряд примитивов со своими специфическими свойствами. Графический редактор Автокада предоставляет в распоряжение пользователя следующий набор геометрических элементов (примитивов):

Точка (Point) • + □ ⊗	Простейший примитив Автокада. Точки могут изображаться на экране дисплея различными графическими знаками. Точка определяется координатами (X, Y, Z)

Отрезок (Line) 	Часть прямой линии, определяемая двумя крайними точками. Отрезки в Автокаде не имеют толщины
Фигура (Solid) 	Часть плоскости, ограниченная четырьмя (тремя) отрезками, определяемыми по четырем точкам, две из которых могут совпадать. На отрисовку фигуры влияет установка режима закрашивания
Полоса (Trace) 	Примитив того же типа, что и фигура. Как и фигура, он определен в системе по четырем точкам, однако отрисовывается как отрезок задаваемой ширины
Дуга (Arc) 	Часть окружности, определяемая центром, радиусом и двумя центральными углами. Несмотря на единый способ хранения геометрической информации, дугу в Автокаде можно построить 10 способами
Круг (Circle) 	Часть плоскости, ограниченная окружностью. При удалении скрытых линий круг воспринимается как непрозрачный объект
Полилиния (Polyline)   	Составная линия, включающая в себя прямолинейные и дуговые сегменты. Отличительной особенностью полилинии является то, что ее сегменты могут иметь не только постоянную, но и переменную ширину. На отрисовку полилинии также влияет режим закрашивания. Для работы с этим примитивом существует специальная команда редактирования полилинии, ПОЛРЕД (PEDIT). Разновидностью полилинии является трехмерная полилиния (3D Polyline), которая не может включать в себя дуги и сегменты которой не обладают шириной. Трехмерная полилиния также может редактироваться командой ПОЛРЕД (PEDIT)

Многоугольник (Polygon) 	Объект, создаваемый командой МН-УГОЛ (POLYGON). Отрисовывается одной полилинией, состоящей из прямолинейных сегментов равной длины
Эллипс (Ellips) 	Объект, создаваемый командой ЭЛЛИПС (ELLIPS). Отрисовывается в Автокаде одной полилинией, сегменты которой являются дугами двенадцатицентрового овала
Кольцо (Donut) 	Объект, создаваемый командой КОЛЬЦО (DONUT). Отрисовывается замкнутой полилинией, состоящей из одного имеющего ширину дугового сегмента
3М Грань (3D Face) 	Часть плоскости, ограниченная четырьмя отрезками. Основным свойством граней является их непрозрачность при удалении скрытых линий. Если четыре определяющие грань точки не лежат в одной плоскости, то свойство непрозрачности утрачивается
3М Сеть (3D Mesh)  	Трехмерный объект, определяемый двупараметрическим массивом вершин (сеткой MxN вершин). Трехмерная сеть является развитием трехмерной грани и может быть расчленена на самостоятельные трехмерные грани. Трехмерные сети предназначены для аппроксимации трехмерных поверхностей (в основном не вручную, а при помощи Автолиспа). В Автокаде предоставлена возможность простого формирования поверхностей (сфера, конус, цилиндр, поверхность вращения, поверхность сдвига и др.)
Форма (Shape) 	Представляют собой специальные графические примитивы, составленные из отрезков, дуг и окружностей. Этот графический объект создается вне Автокада как текстовый файл, содержащий описание способа отрисовки графического объекта

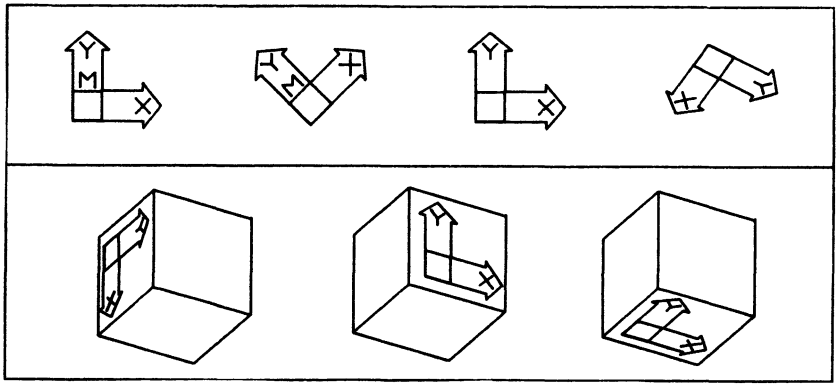
Текст (Text) 	Примитив, являющийся совокупностью строки текста и специфических для текста свойств: гарнитуры, угла наклона, точки вставки и др. В Автокаде используется векторный способ построения шрифтовых знаков, которые создаются как формы
Размер (Dimension) 	Составной примитив, состоящий из отрезков, дуг, стрелок и размерного текста. Подобно блоку размер может быть расчленен на составные примитивы, после чего он теряет свои специфические свойства
Блоки (Block) 	Составной примитив, сформированный из других примитивов или их совокупностей и имеющий имя
Атрибут (Attribute) 	Специальный примитив, предназначенный для работы с блоками. Атрибут можно рассматривать как некую текстовую переменную, которая, будучи записана в блок, позволяет связывать с каждым вхождением блока в рисунок некоторую постоянную или переменную текстовую информацию

1.2.2. Система координат

В Автокаде используется традиционная система декартовых координат. Все координаты точек Автокад хранит в своей собственной внутренней системе координат - т. н. мировой системе координат, МСК (WCS). Однако точки мы можем задавать не только в ней: пользователю предоставляется возможность определять свои собственные системы координат, определения которых Автокад может запомнить - занести в список определенные в рисунке системы координат, в котором всегда находится как минимум одна система координат - мировая. Такие введенные пользователем системы координат называются пользовательскими системами координат, ПСК (UCS). Таким образом, при работе с рисунком можно пользоваться разными системами координат. Однако в каждый момент времени пользователь работает только с одной предварительно выбранной системой координат, которая называется текущей. Вся работа с изображением осуществляется в текущей системе координат.

В левом нижнем углу графической зоны экрана Автокад постоянно показывает пиктограмму текущей системы координат. Глядя на нее, можно сразу

понять, в какой системе координат мы находимся в настоящий момент и куда направлены оси координат X и Y.



Если мировая система координат является текущей, на пиктограмме отображается буква "M"; ее отсутствие говорит о том, что текущей является пользовательская система координат. В момент начала сеанса работы с новым чертежом Автокад делает мировую систему координат текущей. Точка (0,0) располагается в левом нижнем углу графической зоны экрана, а оси X и Y направлены соответственно влево и вверх, как указывает пиктограмма. При этом ось Z направлена перпендикулярно экрану монитора в сторону пользователя (правосторонняя система координат). Позже вы увидите, как можно определить свою ПСК.

1.2.3. Единицы измерения и масштаб

Расстояния между точками на рисунке измеряются в условных единицах. Конкретный формат представления размеров (дюймы, футы, сантиметры, миллиметры и др.) не имеет значения для Автокада. Иначе говоря, при создании объектов в чертеже Автокад "измеряет" все расстояния в относительных единицах. Соответствие между условными единицами Автокада и конкретной системой (метрической, дюймовой и т.п.) устанавливается выбором формата представления. В Автокаде нет масштаба в обычном понимании конструктора: создавая масштабный чертеж на кульмане, конструктор вынужден пересчитывать реальные размеры в зависимости от соотношения формата листа и размеров изделия, которыми определяется масштаб (например, чертеж здания делается обычно в масштабе 1:50, а механизм наручных часов - в масштабе 10:1). В Автокаде конструктор задает все расстояния и координаты в реальных единицах - мы как бы работаем в масштабе 1:1.

Масштабирование различных частей изображения в соответствии с желаемым форматом документа может осуществляться в момент компоновки чертежа (для этого следует использовать команду МАСШТАБ (SCALE) или при выводе чертежа или его части на плоттер (принтер).

1.2.4. Вид

Когда вы создаете чертеж на Автокаде, то работаете с изображением части (или всего) чертежа, выводимой на дисплей. Будем называть эту часть изображения *видом*. Эту видимую часть чертежа (окно зрения) можно увеличивать (при этом изображение чертежа будет уменьшаться), уменьшать (изображение будет увеличиваться) или перемещать по полю чертежа без изменения масштаба отображения (панорамирование). Изменение вида осуществляется командой ПОКАЖИ (ZOOM).

1.2.5. Слой

Автокад дает возможность распределять выбранные фрагменты чертежа по различным *слоям*. Работая за кульманом, конструктор имеет дело с одним листом бумаги и располагает изображения объектов на нем и только на нем. В Автокаде располагать изображение можно как бы на нескольких совмещенных в пространстве носителях (это можно сравнить с наложенными друг на друга прозрачными кальками). Например, чертеж может содержать на одном слое план здания, на другом - схему электропроводки, а на третьем - водопроводные коммуникации. Таким образом, грамотное использование слоев позволяет систематизировать и удобно организовать чертеж.

Количество слоев не ограничивается. Слои можно делать видимыми и невидимыми. С каждым слоем чертежа связывается цвет и тип линий. По мере создания чертежа вы можете вводить новые слои, менять свойства существующих. При создании слоя вы должны присвоить ему имя, по которому будете отличать один слой от другого. Имена существующих в рисунке слоев хранятся в специальном списке. Подобно системе координат в любом чертеже Автокада всегда существует по крайней мере один слой с именем "0". Этот слой отличается от создаваемых вами слоев некоторыми свойствами, которые будут рассмотрены ниже.

1.2.6. Чертеж

Чертеж - это файл с информацией, описывающей графический объект. Следует принять во внимание тот факт, что в результате общения конструктора с системой Автокад (как, впрочем, и с любым другим пакетом САПР) получается не чертеж в общепринятом понимании этого слова, а собственно файл, содержащий некую геометрическую и вспомогательную информацию, полностью описывающую графический объект. В этом смысле черчение на листе ватмана кардинально отличается от создания проекта при помощи системы Автокад, хотя благодаря реализованной в Автокаде "дружественной" модели общения с системой это различие не так бросается в глаза.

Автокад позволяет создавать и редактировать чертеж множеством различных способов. Любое изображение создается с помощью базового набора примитивов, описанных выше. Каждый примитив, как правило, создается своей коман-

дой. Команды создания примитивов будут рассмотрены в следующих главах. Каждая команда предоставляет пользователю несколько способов построения одного и того же объекта по заданным геометрическим параметрам. Например, правильный многоугольник может быть задан координатами центра, числом сторон и радиусом описанной (вписанной) окружности либо длиной стороны - в зависимости от задачи. Аналогично командам отрисовки примитивов в Автокаде существует базовый набор команд (способов) редактирования чертежа. Любую часть изображения можно стирать, перемещать, "размножать", растягивать, осуществлять зеркальное отображение и пр.

При работе традиционным способом точность геометрических построений определяется набором чертежных инструментов, имеющихся в распоряжении конструктора (циркуль, линейка, угольник, транспортир и пр.). Те из конструкторов, которым приходится на листе ватмана проводить геометрические вычисления (определять сложные профили, строить сопряжения, сечения, осуществлять трехмерное моделирование и пр.), знают, насколько это трудоемкая задача и как быстро теряется точность. Автокад предоставляет в распоряжение конструктора все вычислительные ресурсы микроЭВМ, а значит, и новые средства создания и редактирования чертежа. Помимо высокой точности построений (задаваемой пользователем), Автокад позволяет упростить геометрические построения, определяя геометрически характерные точки объектов, предоставляет информацию о чертеже (координаты, расстояния, площади и т.п.).

САПР вносит в работу конструктора элементы современной технологии обработки данных - возможность накопления архива и использования его при создании новых разработок. Именно необходимостью активного использования новых средств работы с чертежом и объясняются те трудности, которые возникают на начальном этапе освоения Автокада. Для того чтобы общение с системой было наиболее эффективным, следует ознакомиться со всеми предоставляемыми средствами и возможностями и выбирать оптимальный, с точки зрения временных затрат, способ решения задачи.

1.3. Главное Меню Автокада

Первое, что появляется на экране при вызове Автокада, - это *Главное Меню*. Главное Меню обеспечивает доступ к различным частям системы Автокад, а также завершает сеанс работы с системой. Основная часть системы Автокад - графический редактор. *Графический редактор* - это программа, в которой собственно создается и редактируется чертеж. В Автокад входят также различные вспомогательные утилиты (работы с файлами, настройки Автокада и др.).

После загрузки программы на экране в текстовом режиме появляется Главное Меню системы Автокад. Данное меню обеспечивает доступ к различным частям системы. Выбор из меню осуществляется нажатием клавиши с цифрой, соответствующей той ветви, на которую требуется перейти, а затем клавиши исполнения команды. В зависимости от клавиатуры компьютера эта клавиша

может называться по-разному. Мы будем называть ее RETURN. Главное Меню содержит следующие разделы:

0. Выйти из Автокада
1. Начать НОВЫЙ рисунок
2. Отредактировать СУЩЕСТВУЮЩИЙ рисунок
3. Вычертить рисунок на плоттере
4. Распечатать рисунок на принтере
5. Настроить Автокад
6. Работа с файлами
7. Компиляция файлов форм и шрифтов
8. Обновление рисунка, созданного старой версией Автокада

Рассмотрим все варианты выбора, предлагаемые Главным Меню.

Выбор "0" - система завершает свою работу и возвращает управление операционной системе.

Выбор "1" - приглашает к созданию нового чертежа. При этом система запрашивает имя создаваемого чертежа. Присвоенное будущему чертежу имя становится именем того файла, который используется для хранения чертежа на диске.

Выбор "2" - редактирование существующего чертежа: внесение изменений и дополнений в существующий чертеж или только отображение чертежа на экране графического монитора. За один сеанс работы с системой Автокад можно создавать, редактировать и выводить на графопостроитель множество чертежей или выполнять многократную обработку одного и того же чертежа. Для обеспечения повторного редактирования система помнит самое последнее имя чертежа и считает его "подразумеваемым чертежом". Таким образом, выборы "1" и "2" приглашают к созданию, редактированию и сохранению графических файлов, т.е. основное время пользователь проводит в графическом редакторе Автокада.

Выборы "3" и "4" используются для получения "твердых копий" чертежей и их фрагментов при помощи графопостроителя или принтера.

Выбор "5" - конфигурирует систему Автокад. С помощью этой процедуры можно указать системе, какое оборудование и с какими параметрами будет использоваться в работе.

Выбор "6" - это возможность просмотра каталога графических файлов и выполнения операций с этими файлами. Перейдя на ветвь "6", мы войдем в новое меню - меню "Работа с файлами", из которого можно произвести настройку системы:

О. Выход в Главное Меню
1. Вывод списка имеющихся рисунков
2. Вывод списка файлов по спецификации пользователя
3. Удаление файлов
4. Переименование файлов
5. Копирование файлов

Выборы "7" и "8" выходят за рамки настоящего руководства.

Для создания нового чертежа необходимо вызвать графический редактор, определив имя файла чертежа. Имя графического файла должно состоять из латинских букв (или цифр). Имя файла может включать в себя некоторые разрешенные символы (например, тире или знак подчеркивания) и не должно содержать пробелы. Расширение указывать не нужно - Автокад по умолчанию присвоит указанному вами имени файла стандартное расширение .dwg.

Вызовем графический редактор:

```
Enter Selection: 1
Enter NAME of drawing: TEST
```

```
Ваш выбор: 1
Введите ИМЯ рисунка: TEST
```

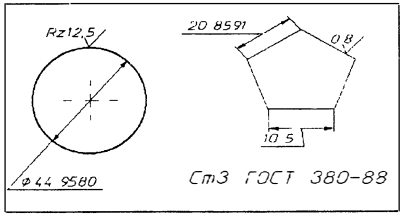
После этого Автокад создает чертеж с новым именем и вызывается графический редактор.

Система для выполнения чертежей в Автокаде с учетом требований ЕСКД

- Выполнение чертежей в Единой Системе Конструкторской Документации
- Быстрая перенастройка ранее выполненных чертежей с последующими изменениями ГОСТа и требований ЕСКД
- Оперативное составление спецификаций и заполнение штампов в автоматическом и диалоговом режиме
- Значительное сокращение сроков выполнения чертежей и повышение качества разработки и оформления сопровождающей документации.

Характеристики и возможности

- Стандартные средства автоматической простановки линейных размеров, диаметров и угловых размеров в соответствии с требованиями ЕСКД



- Черчение кривых линий обрыва, нанесение осевых линий окружностей и дуг с пересечением пунктиров в центре кривизны, рисование выносных линий с полками и номерами позиций для дальнейшего автоматизированного составления спецификаций, прорисовка линий сгиба на листовых развертках деталей, простановка специальных знаков и надписей (см. рисунок)
- Расширенное меню с простым доступом к командам, повышающим скорость черчения, например: для получения гарантированной стыковки примитивов, для переброски фрагментов чертежа в нерегенерируемые стеки с целью сокращения времени регенерации чертежа, для размещения текста вдоль радиусной кривой, для удобного выбора параметров штриховки и ее размещения, для быстрого ввода стандартных форматов с заполнением штампа и привязки к чертежу, для оперативного набора собственной базы символов, знаков, форм с их отображением в меню и для многого другого
- Собственные файлы с дополнительной библиотекой чертежных символов, наименований часто используемых надписей и материалов по ГОСТу, а также макросов специальных функций обработки текста
- Возможность просмотра и редактирования файлов шрифтов и дополнительной библиотеки
- Возможность наполнения отраслевой базы фрагментов чертежей (AutoГОСТ)
- Максимальный объем пакета не превышает 360 Кбайт

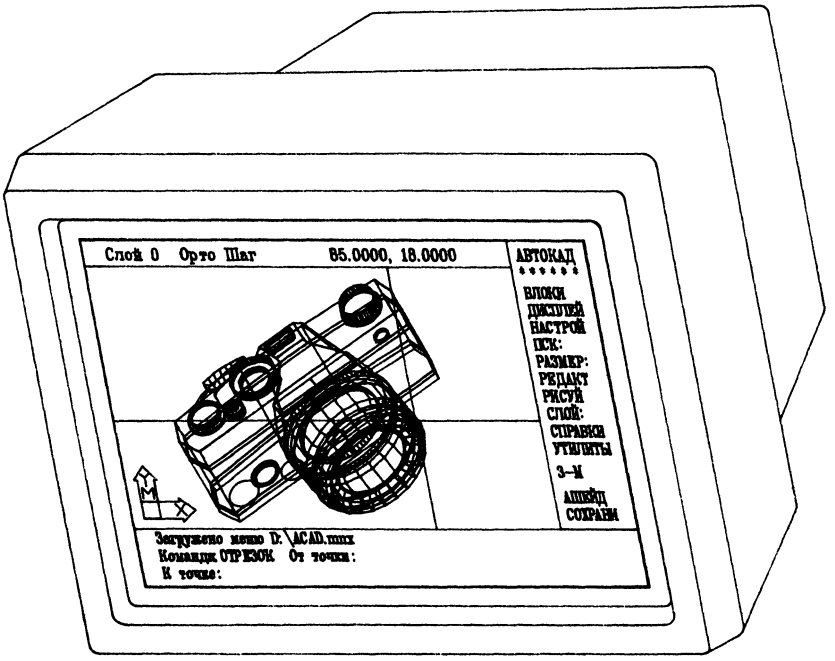
Требования к оборудованию и программному обеспечению

- Персональная ЭВМ типа IBM PC/XT, PC/AT, серии PS/2 и PC-386
- Операционная система MS DOS версии 3.0 и выше
- Пакет автоматизированного проектирования AutoCAD любого варианта версии 10

Пакет AutoЕСКД придает новые возможности системе AutoCAD

Научно-производственный
и учебно-консультационный Центр
совместного советско-американского предприятия "Диалог"
на базе Московского инженерно-физического института

МИФИ
СССР, 115409 Москва, ул. Москворечье 31 2
Телефон (095) 3243055, (095) 3247166
Телефакс (095) 3243055


Слой 0 Орто Шаг 85.0000, 18.0000
АВТОКАД
БЛОКИ
ДИСПЛЕЙ
НАСТРОЙ
ИСК:
РАЗМЕР:
РЕДАКТ
РИСУЙ
СЛОЙ:
СПРАВКИ
УТИЛИТЫ
3-М
ПЛЕЙД
СОХРАНИ
Загружено меню D:\ACAD.mnx
Команда: ОТРЕЗОК От точки:
К точке:

ГЛАВА 2

ГРАФИЧЕСКИЙ РЕДАКТОР

В этой главе будут рассмотрены основные приемы работы с графическим редактором Автокада. *Графический редактор* - это программа, отображающая на экране графическую информацию и исполняющая команды создания, изменения, просмотра чертежа и вывода его на плоттер или принтер.

Привлекательность Автокада заключается в том, что он предоставляет удобную среду общения, в которой создается и редактируется чертеж. При создании Автокада было приложено максимум усилий для того, чтобы снизить психологический барьер, возникающий при переходе к автоматизированному выполнению чертежей.

При вызове графического редактора на экране появляется изображение, состоящее из четырех функциональных зон:

- *графической зоны*, которая содержит в себе часть воображаемой экранной плоскости чертежа (координаты графической зоны в чертеже определяют вид)
- *зоны экранного меню*
- *зоны строки состояния и падающих меню*, отображающей некоторые текущие настройки или строку падающих меню
- *зоны командной строки*

При работе с Автокадом *всегда следите за сообщениями в командной строке* - в ней происходит формирование команд и в нее выводятся все сообщения графического редактора.

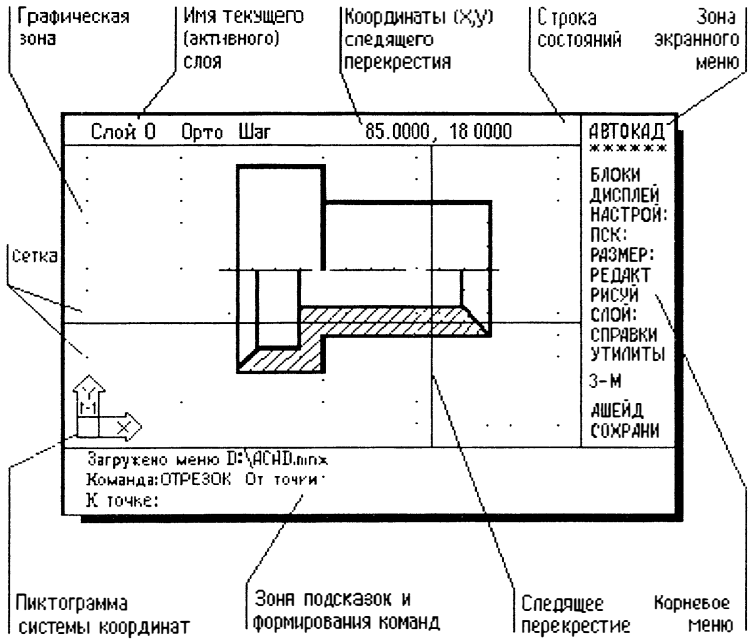
Общение с Автокадом происходит посредством команд. Условно можно выделить два режима работы Автокада - режим выполнения команды и режим ожидания команд. После того, как вы даете команду и отвечаете на все вопросы, Автокад ее выполняет и переходит в режим ожидания следующей команды. В момент ожидания Автокадом команды вы видите в зоне командной строки экрана подсказку:

```
Command:
```

```
Команда:
```

Подвигайте "мышь", обращая внимание на то, что в разных зонах экрана курсор имеет разный вид. В пределах графической зоны вы видите перекрестие графического курсора, которое предназначено для указания точек в поле чертежа. Зона командной строки для курсора недоступна - с ней вы работаете только при помощи клавиатуры. В зоне экранного меню курсор имеет вид выделения другим цветом строки пункта меню. Переместите курсор в зону строки состояния, вы вызовете появление строки падающего меню, в котором можете при помощи

левой клавиши "мыши" выбирать нужное падающее меню. Отметим, что если в экранное меню вы можете попасть нажатием клавиши выбора меню (обычно это клавиша INSERT), то падающее меню без "мыши" недоступно.



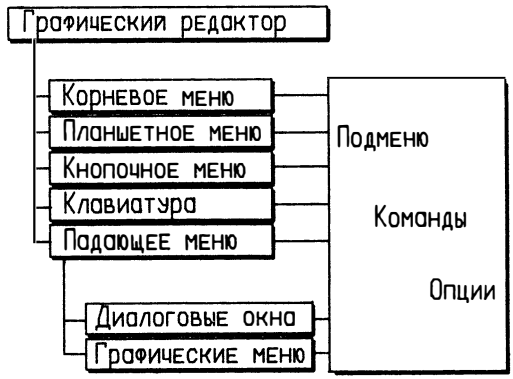
Автокад обладает гибкой структурой команд, что позволяет выполнять команды в любой последовательности. Ввод команд осуществляется набором с клавиатуры или выбором из меню.

Прямым способом вызова команды является набор имени команды с клавиатуры - вы набираете ее имя и затем нажимаете любую клавишу с функцией ИСПОЛНЕНИЕ КОМАНДЫ. В Автокаде эту функцию имеют три клавиши - RETURN (иногда эта клавиша называется ENTER), ПРОБЕЛ и правая клавиша "мыши".

Команды имеют простую и ясную мнемонику и легко запоминаются, составляя подмножество естественного языка. В системе Автокад имеется более 120 основных команд и большое количество дополнительных. Большинство этих команд связано с выполнением конкретных действий по черчению,

редактированию, нанесению размеров, изменению текущих настроек и пр. Дружественный интерфейс Автокада не требует от вас зубрежки имен команд: все команды могут быть вызваны с помощью меню.

Меню - это иерархическая структура, позволяющая быстро найти нужную команду. Любая из 120 команд может быть выбрана из меню самое большее за четыре шага. Система меню организована таким образом, что однотипные команды (например, команды отрисовки примитивов) располагаются в одном подменю. Такое группирование команд также значительно ускоряет поиск команд. Подменю может включать в себя не только стандартные команды Автокада, но и другие подменю, более низкого уровня (если пункт вызывает команду, то он заканчивается двоеточием; если при выборе пункта просто осуществляется переход в подменю, то двоеточия нет). Поэтому при выборе пунктов меню может не вызываться никакая команда, а просто осуществляется переход в подменю. Кроме того, в пунктах меню может располагаться не только собственно команда Автокада, но и некоторая комбинация команд и, во многих случаях, их ключей, что позволяет в разных пунктах описать несколько режимов работы одной команды. Для программистов скажем, что пункт меню является по сути макроопределением работы с клавиатурой, в его определение могут быть помещены выражения Автолиспа или даже целые функции.



Система меню графического редактора

Как уже отмечалось ранее, Автокад - открытая система. Одним из средств, обеспечивающих открытость Автокада, является возможность модификации стандартного меню.

Графический редактор является для чертежа тем же самым, чем является редактор текстов для какого-либо документа. Когда создается новый чертеж или редактируется уже существующий, система автоматически осуществляет загрузку редактора чертежей, который предоставляет в распоряжение пользователя команды для создания, модификации, просмотра чертежей и вычерчивания их на бумаге. После окончания работы с чертежом перед возвратом в Главное Меню можно сохранить чертеж или удалить любые внесенные изменения. Хранение базы данных изображения (размеры и позиция каждого примитива, размеры всего чертежа, его характеристики и др.) осуществляется системой путем записи в файл чертежа при выходе из системы. Ввод точек может быть осуществлен как

с клавиатуры, так и при помощи устройства указания, что позволяет не заботиться о значениях координат точек.

2.1. Экранное меню

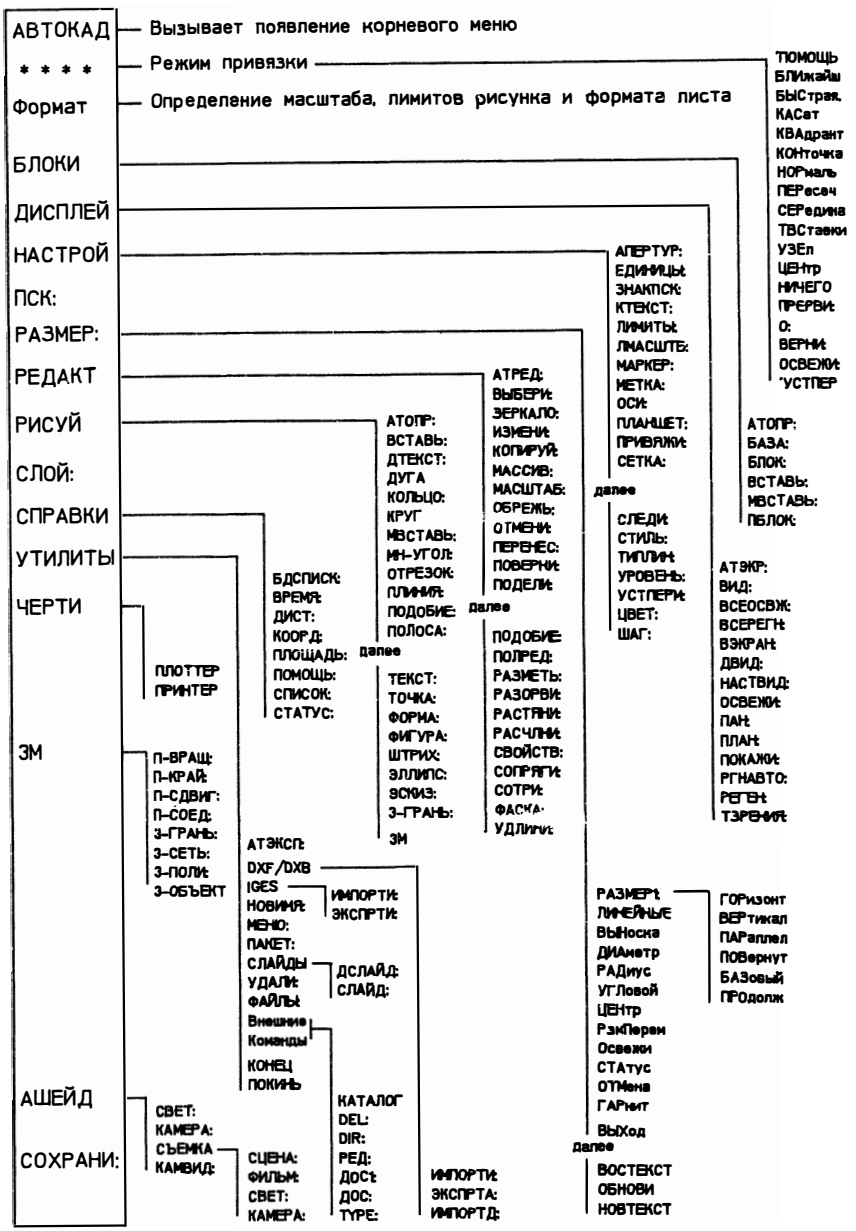
Наиболее часто используемым меню является экранное меню. Оно расположено с правой стороны экрана дисплея. Если выбрать самую верхнюю строку - АВТОКАД, мы попадаем в *корневое меню*. Строки корневого меню вызывают следующие меню, которые вызывают более детальные меню. Таким образом, группы меню образуют древовидную структуру, заканчивающуюся отдельными командами. Выбор строк меню осуществляется указанием пунктов меню курсором и последующим нажатием клавиши с функцией "исполнение команды".

Работа с клавиатурой	*Работа с "мышью"*
Нажмите кнопку с функцией "выбор меню" (обычно это клавиша INS), и курсор переходит в зону экранного меню. Используя клавиши перемещения курсора, вы можете выбирать различные пункты. При повторном нажатии клавиши выбора меню выбранный пункт выполняется.	С "мышью" выбор из меню значительно ускоряется - вам достаточно поместить курсор на нужный пункт и нажать на левую клавишу.

Почти все пункты корневого меню являются корнями подменю более низкого уровня. Некоторые строки в меню заканчиваются двоеточием, например, DIM: (РАЗМЕР:), LAYER: (СЛОЙ:), SAVE: (СОХРАНИ:). Выбор этих строк приводит не только к переходу на подменю, но и к вызову соответствующей команды. Любая строка выбора, которая автоматически приводит к выполнению команды, сопровождается двоеточием.

В корневое меню можно вернуться из любого подменю - для этого достаточно выбрать строку AUTOCAD (АВТОКАД), записанную в верхней части страницы меню. Поскольку наиболее часто используемые команды - это команды отрисовки и редактирования, практически из любого подменю можно перейти сразу в подменю DRAW (РИСУЙ) и EDIT (РЕДАКТ), для этого достаточно выбрать соответствующие пункты, расположенные в нижней части большинства страниц меню. Там же расположен пункт _LAST_ (ПРЕДМЕНЮ), позволяющий вернуться в предыдущее меню. В некоторых подменю так много пунктов, что они не помещаются на одной странице меню. Пункты next (далее) и previous (предыдущ) дают возможность листать страницы меню. Наконец, на каждой странице меню под строкой АВТОКАД есть пункт, помеченный звездочками - ****. Этот пункт, кроме средств объектной привязки (что это такое, мы рассмотрим позднее), позволяет получить справку по той команде, с которой вы работаете в настоящий момент, причем попросить справку можно прямо в процессе выполнения команды.

Структура стандартного экранного меню



2.2. Падающие меню

В Расширенный Пользовательский Интерфейс (РПИ) входят такие средства, как падающие меню, графические меню и диалоговые окна.

Падающие меню дополняют экранное меню, облегчая поиск и выбор команд и их опций. Падающие меню доступны пользователю только в том случае, если в системе установлено устройство указания - "мышь" или планшет.

Переместите курсор в зону строки состояния, находящуюся над графической зоной экрана. Строка состояния исчезнет, а на ее месте появится строка меню. Пункты этого меню являются заголовками падающих меню. Двигая "мышь", добейтесь выделения цветом одного из пунктов строки меню и нажмите на левую кнопку. При этом на экране появится падающее меню, из которого можно выбирать команды так же, как и из экранного меню.

Хотя пункты падающих меню в общем повторяют пункты экранного меню, одни и те же команды могут работать в этих меню по-разному, что предоставляет пользователю дополнительные возможности выбора.

Обратите внимание также на то, что при выборе команд из падающего меню в зоне экранного меню появляется меню выбранной команды - выбрав команду из падающего меню, можно далее указывать опции команды из экранного меню.

В стандартной поставке Автокада версии 10 в строке меню семь пунктов, т.е. доступно семь падающих меню. В более поздних поставках Автокада дополнительно прилагается расширенное падающее меню (с именем acaduk), в котором число падающих меню увеличено до девяти и несколько изменена организация команд в падающих меню. Как и все меню Автокада, падающие меню могут быть модифицированы пользователем - вы можете добавлять в падающие меню свои команды и даже добавлять свои собственные падающие меню. Следует помнить однако, что число падающих меню не должно превышать десяти.

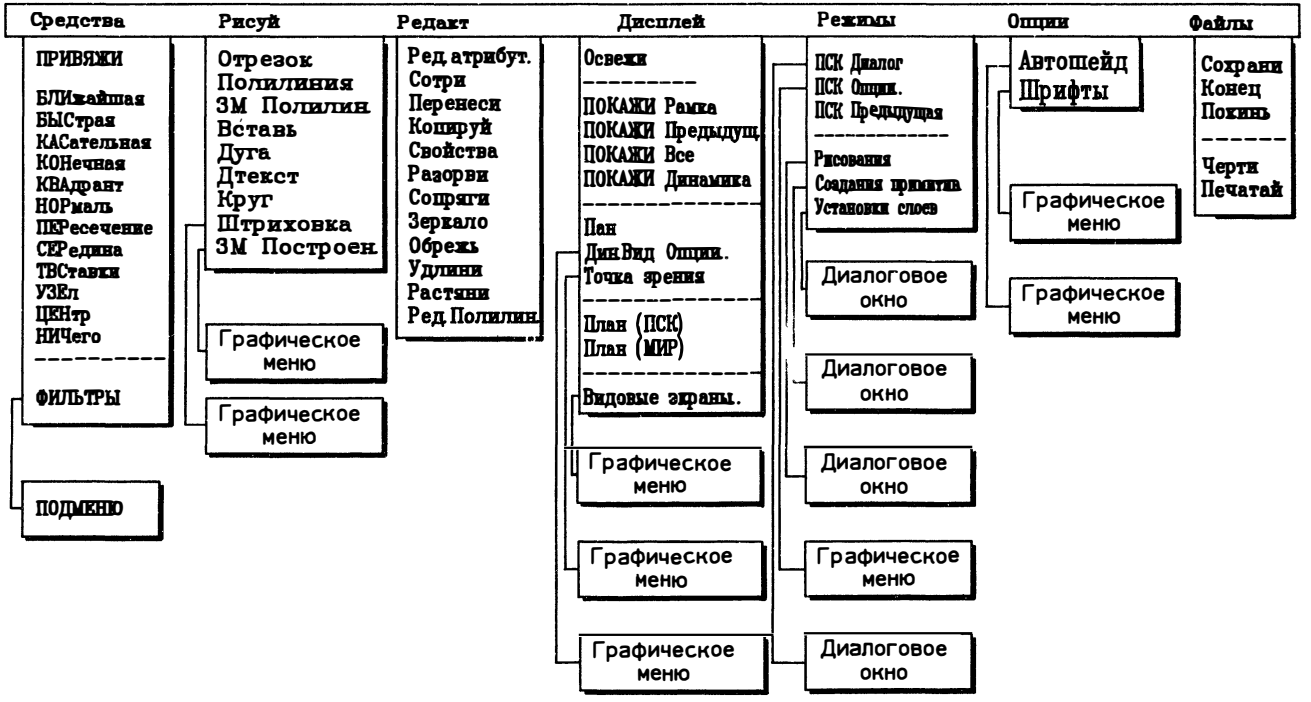
Убедиться, поддерживает ли ваша система падающие и графические меню, вы можете и без "мыши" - для этого необходимо проверить значение системной переменной POPUPS:

```
Command: SETVAR
Variable name or ?: POPUPS
POPUPS = 1 (read only)
```

```
Команда: УСТПЕРЕМ
Имя переменной или ?: POPUPS
POPUPS = 1 (только для чтения)
```

Если получен ответ POPUPS = 1, то установленный драйвер монитора поддерживает РПИ и при установленной "мыши" вам будут доступны падающие и графические меню, диалоговые окна. Если POPUPS = 0, то для работы РПИ необходим соответствующий драйвер дисплея или дисплей, который имеет такую возможность. Если система ответила "Unknown variable name" ("Неизвестное имя переменной"), проверьте, правильно ли набрано имя переменной POPUPS.

Строка меню и падающее меню



Если синтаксической ошибки нет, то вы пользуетесь старой версией Автокада, в которой такой переменной нет. Номер версии Автокада хранится в системной переменной ACADVER:

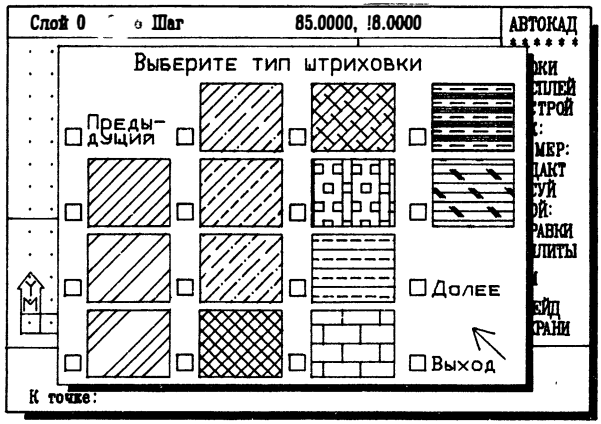
```
Command: SETVAR
Variable name or ?: ACADVER
ACADVER = "10" (read only)
```

```
Команда: УСТПЕРЕМ
Имя переменной или ?: ACADVER
ACADVER = "10" (только для чтения)
```

2.3. Графические меню

Известно, что человек визуально легче и быстрее воспринимает графическую информацию, чем текстовую; и графические меню призваны сделать общение с Автокадом еще более наглядным.

Выберите из строки падающих меню пункт Draw (Рисуй), а затем из падающего меню пункт Hatch... (Штриховка). На экране появится графическое меню, в котором показаны типы возможных стандартных штриховок (см. рисунок ниже). Как видим, в графических меню возможности выбора представлены в виде графических изображений (эти изображения не что иное, как слайды Автокада).

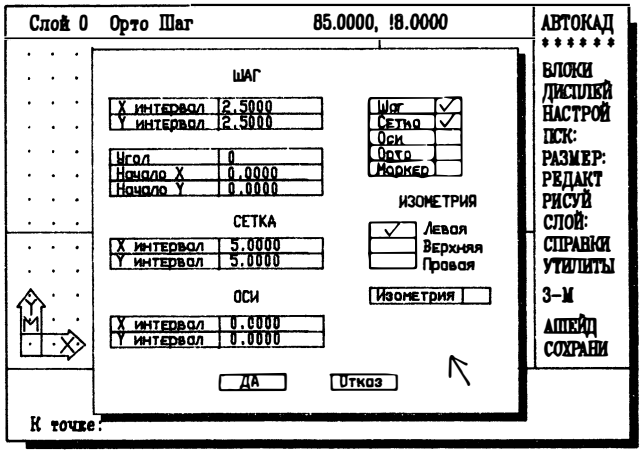


Курсор в графическом меню имеет форму стрелки. Выберите понравившийся вам тип штриховки и укажите стрелкой на маленький квадратик слева от слайда. Квадратик изменит цвет, а выбранный слайд будет взят в рамку. Нажав на левую кнопку "мыши", вы осуществите выбор: в командной строке появится выбранная команда (опция) или даже последовательность команд. В стандартном меню acad

имеются графические меню типов штриховок, шрифтов и др. В меню acaduk есть наглядное графическое меню размерных переменных.

2.4. Диалоговые окна

Иногда требуется просмотреть или установить несколько параметров одновременно, что в командной строке сделать невозможно. Диалоговые окна реализуют удобную среду, представляющую значения параметров в табличной форме, что позволяет охватить взглядом связанные параметры и за один сеанс работы с диалоговым окном изменить значения нескольких переменных. На рисунке ниже показано диалоговое окно команды ДИАЛСРЕД (DDRMODES), служащее для установки общих режимов рисования.



Диалоговые окна вызываются специальными командами, начинающимися с букв "ДИАЛ":

'ДИАЛСРЕД ('DDRMODES)

установка общих режимов рисования (шаг, сетка, оси, изометрия)

'ДИАЛПРИМ ('DDEMODES)

установка режимов создания примитивов (слой, цвет, тип линий, уровень, высота)

'ДИАЛСЛОЙ ('DDLMODES)

окно работы со слоями (создание, включение и отключение слоев, изменение цвета и типа линий, установка текущего слоя)

ДИАЛПСК (DDUCS)

окно работы с пользовательскими системами координат (просмотр списка определенных ПСК, создание новых ПСК, установка текущей ПСК, изменение имени существующей ПСК)

ДИАЛАТР (DDATTE)

окно редактирования атрибутов (эта команда просит указать блок, в котором есть атрибуты)

Окно ввода значений атрибутов

не вызывается отдельной командой, поскольку ввод значений атрибутов блока возможен только при вставке блока; для того чтобы при вставке блока атрибуты можно было вводить через диалоговое окно, системная переменная ATTDIA должна иметь значение 1

Команды, перед именем которых вы видите апостроф, могут быть вызваны в прозрачном режиме, т.е. в процессе выполнения других команд (подробнее о прозрачных командах см. ниже). Установки, сделанные при прозрачном выполнении команд, отменяются, если отменяется основная команда. Диалоговые окна, которые вызываются специальными командами, включены в падающее меню.

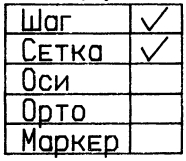
Несмотря на то, что эти команды могут быть вызваны из меню, пакетов или из Автолиспа, все они требуют непосредственного диалога с пользователем. Исключены, например, возможность заполнения из Автолиспа граф в таблицах диалоговых окон, а также указание на клавиши.

Так же как и в графических меню, курсор имеет вид стрелки. Все диалоговые окна включают изображение клавиши ДА, и большинство из них - изображение клавиши ОТКАЗ. До тех пор пока диалоговое окно находится на экране, работа с меню, вызов команд с клавиатуры и клавиши управления режимами рисования блокируются. Курсор можно перемещать не только "мышью", но и клавишами перемещения курсора. Нажатие клавиш CTRL C и ESC равносильно указанию клавиши ОТКАЗ.

Ряд диалоговых окон имеет вложенные окна, которые в случае необходимости появляются на экране, заслоняя собой основное окно. Для того чтобы продолжить работу с основным окном, нужно завершить работу с вложенным.

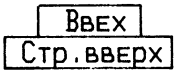
Ниже описываются клавиши и графы диалоговых окон.

Клавиши управления



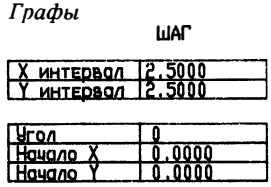
Изображаются прямоугольниками, которые могут быть пустыми или содержать знак управления (√). Клавиши управления выполняют две функции - управляют переключателями режимов (например, режимом отображения сетки) и позволяют выбирать один из возможных вариантов (например, текущий слой).

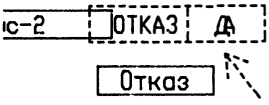
Клавиши исполнения



Служат для управления диалоговым окном - подтверждения или отказа от действия или окна (клавиши ДА или ОТКАЗ), просмотра длинных списков в графах (клавиши ВВЕРХ, ВНИЗ, СТР ВВЕРХ, СТР ВНИЗ).

Графы





Служат для указания в них числовых значений (размер шага привязки, номер текущего цвета) и текста (имя слоя, ПСК и пр.). Указание на графу "раскрывает" ее, делая ее доступной для ввода. При этом появляются изображения дополнительных, относящихся только к этой графе клавиш ДА и ОТКАЗ. При вводе осуществляется контроль типа вводимых значений; так, если вы ввели букву там, где разрешен только ввод чисел, то вам не удастся закрыть графу до тех пор, пока вы не исправите ввод (или не откажетесь от ввода клавишей ESC).

Графы вызова

Служат для вызова вложенного диалогового окна, в котором организован выбор значения, заносимого в графу вызова.

2.5. Планшетные меню

Если у вас есть планшет, то вы можете использовать его не только для сколки чертежей, но и как устройство выбора команд, для чего в Автокаде существует меню планшета.

Основным преимуществом планшетного меню является то, что большое рабочее поле планшета дает прямой доступ к любой команде без необходимости долгого "блуждания" по иерархической структуре меню. Вместе с тем вся система меню Автокада (экранное, падающие и планшетное меню) являются, по существу, единым целым, поскольку экранное меню "привязано" как к падающим, так и к планшетному меню. Выбор многих операций с помощью планшетного меню вызывает появление соответствующей страницы экранного меню, что позволяет при помощи экранного меню сделать выбор нужной подкоманды (опции).

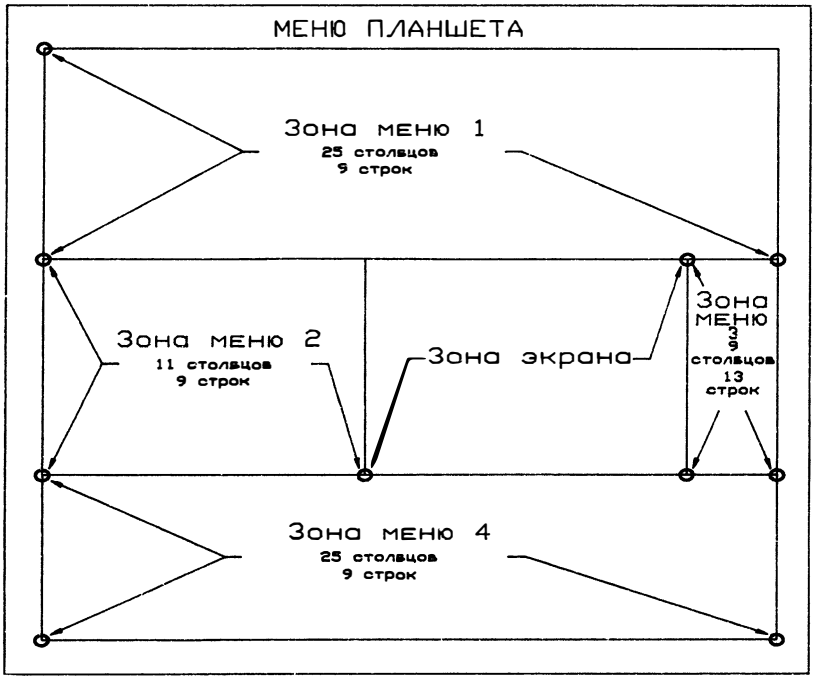
Хотя на темплет вынесены не все команды Автокада, тем не менее всегда возможен доступ к любой команде, входящей в некоторую группу. Для этого

достаточно выбрать клетку "Экранное меню", тогда все остальные команды появятся в экранном меню.

В планшетном меню вместо многих имен команд используются пиктограммы, что значительно облегчает работу.

В стандартную поставку Автокада входит темплет формата А4. В файлах описания меню (acad.mnd и acaduk.mnd) организовано меню для этого темплета. Изображение темплета в виде рисунка Автокада (tablet.dwg) находится на одной из дистрибутивных дискет. Если у вас планшет других размеров, этот рисунок можно вывести на бумагу в нужном масштабе.

Меню планшета разделяется на прямоугольные зоны, в свою очередь состоящие из прямоугольных ячеек, в которых "размещаются" команды. Такая организация планшетного меню позволяет гибко компоновать на планшете зоны меню и зону экрана. В стандартной поставке меню планшета имеет четыре зоны. Для настройки планшета следует закрепить темплет на поверхности планшета, а затем выполнить команду TABLET Config (ПЛАНШЕТ Настрой). Каждая зона меню определяется тремя точками, а экрана - двумя. Эти точки показаны на рисунке внизу. При настройке необходимо указать, сколько строк и столбцов имеется в каждой зоне меню (нужные значения также показаны на рисунке).



2.6. Кнопочное меню и функции клавиатуры

Большинство компьютеров, работающих с системой Автокад, оборудованы тем или иным устройством указания (визир планшета или "мышь"). На этих устройствах имеется ряд клавиш. "Мышь" имеет две или три клавиши, а на визире планшета может располагаться до шестнадцати клавиш. В системе Автокад одна из них резервируется для указания точек и выбора команд из меню (для "мыши" это левая клавиша). Остальным клавишам можно назначить любую функцию путем модификации стандартного меню Автокада. Некоторые функции (отказ и переключение различных режимов) определяются в файле меню так, как показано в таблице:

Комбинации клавиш на клавиатуре	*Обозначение в файле меню (.mnd)*	*Функция*
CTRL C	^C	Отказ от выполнения команды
CTRL B или F9	^B	Переключение режима ШАГ
CTRL O или F8	^O	Переключение режима ОРТО
CTRL G или F7	^G	Переключение режима СЕТКА
CTRL D или F6	^D	Переключение отображения координат
CTRL E	^E	Переключение плоскости изометрии
CTRL T или F10	^T	Переключение режима ПЛАНШЕТ
F1	Нет	Переключение экрана
INS	Нет	Курсор меню
HOME	Нет	Курсор экрана
END	Нет	Отмена перемещения курсора
PG_UP	Нет	Увеличение шага перемещения курсора
PG_DN	Нет	Уменьшение шага перемещения курсора
← ↑ ↓ →	Нет	Управление перемещением курсора

В меню стандартной поставки пунктам кнопочного меню (раздел BUTTONS) соответствуют функции, приведенные в таблице:

Номер пункта	*Запись в файле меню*	*Функция*
1	;	Исполнение команды (RETURN)
2	$p1=*	Отображение падающего меню "средства"
3	^C^C	Отказ от выполнения команды
4	^B	Переключение режима ШАГ
5	^O	Переключение режима ОРТО
6	^G	Переключение режима СЕТКА
7	^D	Переключение отображения координат
8	^E	Переключение плоскости изометрии
9	^T	Переключение режима ПЛАНШЕТ

Как правило, кнопкам устройств указания последовательно ставятся в соответствие пункты кнопочного меню. Так, если вы правильно установили в системе MS-DOS трехкнопочную "мышь", а затем правильно установили эту "мышь" в Автокаде, то вторая кнопка будет выполнять функцию "исполнение команды", а третья - отображать падающее меню "средства". Однако если у вас, например, имеется "мышь" типа "Genius Mouse", которая работает под управлением драйвера, эмулирующего работу "Microsoft Mouse", то в Автокаде ее также следует устанавливать как "мышь" "Microsoft Mouse", и следовательно, функцию "исполнение команды" будет выполнять третья кнопка, а второй кнопкой вы пользоваться не сможете.

2.7. Командная строка и строка состояния

Командная строка

Командная строка - это самая главная зона графического экрана. В ней отображается весь ваш диалог с Автокадом, вне зависимости от того, как вы выбираете команды. *Всегда следите за командной строкой!* Вводить команду с клавиатуры можно только тогда, когда Автокад приглашает вас к этому подсказкой:

```
Command:
```

```
Команда:
```

Если вы видите в командной строке другое сообщение, значит, предыдущая команда еще не завершена, и вы должны либо завершить, либо отменить ее. При выборе команд из экранных, падающих меню и меню планшета любая незавершенная команда отменяется автоматически. Любая команда может быть отменена также указанием на первый пункт любой страницы экранного меню - AutoCAD (АВТОКАД) (при этом вы одновременно возвращаетесь в корень экранного меню). Нажмите клавишу CTRL и, не отпуская ее, клавишу C (далее CTRL C) - это еще один способ отмены команды.

После вызова команды в строке состояния может отображаться также список опций вызванной команды. Выбор опций осуществляется либо вводом с клавиатуры выделенных (прописных) букв опций, либо выбором опции из экранного меню. В большинстве случаев Автокад предлагает одно из значений по умолчанию, помещенное в угловые скобки. Если нажать RETURN, то будет выбрано именно это значение.

В зоне команд помещаются только три строки. Нажав на клавишу F1 (ПЕРЕКЛЮЧЕНИЕ ЭКРАНА), вы расширите зону команд на весь экран и увидите часть протокола диалога (текстовый режим). Повторное нажатие на клавишу F1 вернет вас в графический режим.

Строка состояния

В верхней части графического экрана расположена строка состояния, в которой отображаются некоторые текущие настройки (имя текущего слоя, режимы ШАГ и ОРТО) и координаты перекрестия графического курсора.

Отображением координат перекрестия курсора управляет клавиша F6 (см. таблицу выше). Возможны два режима слежения за координатами перекрестия: включено и выключено. Клавиша F6 переключает из одного режима в другой. Подвигайте курсор по экрану. Если цифры в строке состояния изменяются, то слежение за координатами включено. Нажмите на клавишу F6 и подвигайте курсор. Теперь Автокад не следит за его перемещением и в строке состояния отображаются координаты последней указанной точки, полученной при последнем нажатии левой клавиши "мыши".

Теперь рассмотрим действие клавиши F6 на примере команды LINE (ОТРЕЗОК). Вызовите эту команду при отключенном слежении за координатами и укажите начальную точку "мышью" или вводом координат с клавиатуры. Подвигайте курсор, но следующую точку не вводите. При выключенном слежении за координатами в строке состояния отображаются координаты первой точки. Нажмите [illegible] F6 - теперь в строке состояния вместо текущего положения курсора в текущей ПСК указываются полярные координаты текущего положения курсора относительно первой точки. Луч оси полярных координат начинается в предыдущей точке и направлен параллельно оси X текущей ПСК (угол отсчитывается по умолчанию против часовой стрелки). Нажмите еще раз на клавишу F6 и подвигайте курсор. Теперь в строке состояния отображаются абсолютные координаты перекрестия курсора в текущей ПСК.

2.8. Работа с файлами

Утилита работы с файлами, которая вызывается по выборе пункта 6 Главного Меню доступна также из графического редактора - она вызывается командой FILES (ФАЙЛЫ). Утилита работы с файлами позволяет, не выходя из редактора чертежей, просмотреть содержимое раздела диска, удалять, переименовывать и копировать файлы. По этой команде на экран дисплея выводится уже знакомое нам меню:

```
File Utility Menu:

0. Exit File Utility Menu
1. List Drawing files
2. List user specified files
3. Delete files
4. Rename files
5. Copy file

Enter selection (0 to 5) <0>:
```

```
Меню работы с файлами

0. Выход в Главное Меню
1. Вывод списка имеющихся рисунков
2. Вывод списка файлов по спецификации пользователя
3. Удаление файлов
4. Переименование файлов
5. Копирование файлов

Ваш выбор (от 0 до 5) <0>:
```

Рассмотрим подробнее это меню.

2.8.1. Вывод перечня имен файлов чертежей

Пункт 1 меню утилиты работы с файлами позволяет просмотреть список файлов, находящихся на указываемом вами устройстве в нужном каталоге. Имя устройства или каталога вы указываете в ответ на вопрос:

```
Enter drive or directory:
```

```
Введите имя дисковода или каталога:
```

В ответ можно ввести полный путь поиска файла (например, A:\DRAWINGS\, т.е. искать файл на гибком диске в подкаталоге DRAWINGS) или только имя устройства (например, D:, т.е. искать файл на жестком диске D). Если в ответ просто нажать клавишу RETURN, то файл будет (или не будет, если его там нет) найден в текущем подкаталоге. Если на диске содержится много файлов чертежей и их имена не помещаются на экране дисплея, то система делает паузу и выдает запрос:

```
-MORE-
```

```
-Еще-
```

Для продолжения необходимо нажать какую-то клавишу. После завершения просмотра каталога система выводит суммарное количество файлов и повторяется сообщение "Press RETURN" ("Нажмите RETURN"). После нажатия клавиши RETURN на экран снова выводится меню утилиты работы с файлами.

2.8.2. Вывод списка файлов по спецификации пользователя

Используя выбор 2 меню утилиты работы с файлами, можно попросить систему найти любой файл (не только файл рисунка с расширением .dwg). Для этого нужно в ответ на запрос

```
Enter file search specification:
```

```
Введите имя файла для поиска:
```

ввести полное имя файла (с расширением). Например, можно попросить найти программу-библиотекарь слайдов slidelib.exe. Если система найдет указанный файл, то вы получите ответ:

```
SLIDELIB.EXE
```

Не слишком информативно! Действительно, этот пункт меню используется обычно в том случае, когда нужно найти на диске файлы с похожими именами; в этом пункте меню имя файла можно задавать с использованием стандартных символов поиска ДОС (т.н. "wildcards") - "*" и "**?**".

? Система считает, что на месте этого символа может находиться любой допустимый знак.

* Система считает, что этот знак заменяет любую группу символов.

Предположим, что на диске хранится много файлов рисунков с похожими именами, например 1.dwg, 11.dwg и 121.dwg. Используя возможности этого пункта подменю, можно просмотреть список всех файлов рисунков, имя которых начинается с единицы, например:

```
Enter file search specification: 1*.dwg
```

```
Введите имя файла для поиска: 1*.dwg
```

При этом вы получите ответ:

```
1.DWG
11.DWG
121.DWG
```

Использование специальных символов "**?**" и "*" поможет вам разобраться в хаосе рисунков на диске. Пользуясь ими, можно выводить перечень имен файлов, которые использовались или были созданы для различных целей системой Автокад. Перечислим те типы файлов, которые используются системой Автокад:

BAK - резервная копия файла чертежа
DWG - файл чертежа
DXB - двоичный файл обмена графической информацией
DXF - файл обмена графической информацией
DXK - файл извлечения атрибутов (формат DXF)
LIN - библиотечный файл типов линий
MNU - файл меню
PAT - библиотечный файл штриховки
SCR - файл командного "сценария"
SHP - исходный файл определения формы шрифта

SHX - скомпилированный файл определения формы шрифта
SLD - слайдовый файл
TXT - файл извлечения атрибутов или трафаретов
RF - векторный файл

2.8.3. Удаление файлов

Выбор меню 3 даст возможность удалять файлы. Если, например, вы в процессе работы заняли весь диск и места там для работы уже не осталось, то можно освободить место, удаляя ненужные файлы. Запрос выглядит так:

```
Enter file deletion specification:
```

```
Введите имя файла для удаления:
```

Как и в пункте 1, необходимо указать полное имя удаляемого файла.

2.8.4. Переименование файлов

Выбор 4 дает возможность изменять имя существующего файла (имя файла нужно указывать полностью):

```
Enter current filename: B:WIDGET.DWG
Enter new filename: B:THINGO.DWG
```

```
Введите имя файла: B:WIDGET.DWG
Введите новое имя файла: B:THINGO.DWG
```

После этого система изменит имя файла widget.dwg, находящегося на накопителе B, на thingo.dwg.

2.8.5. Копирование файлов

Используя выбор *5*, можно сделать копию существующего файла:

```
Source filename: ABC.DWG
Destination filename: ABC.SAV
```

```
Введите имя исходного файла: ABC.DWG
Введите имя файла копии: ABC.SAV
```

В примере выше копия файла abs.dwg записывается в файл abs.sav.

2.9. Выход из Графического редактора

Когда работа с чертежом закончена, необходимо выйти из редактора чертежей. Для завершения работы с чертежом в Автокаде существуют три команды с

различными функциями - это команды SAVE (СОХРАНИ), END (КОНЕЦ) и QUIT (ПОКИНЬ).

Имейте в виду, что из графического редактора нужно выходить правильно, используя команды выхода. *Не выключайте компьютер, находясь в графическом редакторе*, даже если вы сохранили свой чертеж. Дело в том, что в процессе работы Автокад создает на жестком диске временные файлы, которые закрываются (удаляются) в момент выхода из редактора. Если вы не вышли правильно из графического редактора, то эти файлы остаются на диске как "потерянные" ("lost clusters"), уменьшая свободное пространство.

Если произошел сбой по питанию, когда вы находились в графическом редакторе, для удаления таких файлов используйте команду ДОС CHKDSK:

```
d:\>chkdsk d: /f
Volume DISK1_VOL2 created Aug 26, 1989 7:46p

260 lost clusters found in 40 chains.
Convert lost chains to files (Y/N)?y
```

В результате работы команды CHKDSK на диске образуются файлы с расширениями .chk, которые следует удалить.

2.9.1. Сохранение изменений в процессе работы

Для гарантии того, что ваша работа не пропадет даром из-за какого-нибудь досадного случая (сбой по питанию, зависание компьютера), желательно периодически сохранять вносимые изменения без выхода из редактора чертежей. Это можно сделать при помощи команды SAVE (СОХРАНИ). По команде SAVE состояние чертежа на текущий момент записывается на диск. Команда SAVE не имеет опций и просит указать имя файла, в который будет записан текущий чертеж. По умолчанию берется имя текущего чертежа:

```
Command: SAVE
File name <current>:
```

```
Команда: СОХРАНИ
Имя файла <current>:
```

Нажав на клавишу ENTER, вы подтвердите выбор по умолчанию.

Команда SAVE может использоваться также для создания копии чертежа.

2.9.2. Выход с сохранением изменений

Команда END (КОНЕЦ) обеспечивает возврат к Главному Меню с автоматическим сохранением чертежа в последней редакции. Старая копия чертежа будет иметь тип, измененный с .dwg на .bak (предыдущий файл с

расширением .bak будет удален). Обновленный файл теперь становится файлом чертежа .dwg и в дальнейшем, если требуется, его снова можно редактировать средствами системы Автокад.

Если вы хотите восстановить предыдущий файл, то ему следует средствами ДОС присвоить расширение .dwg (переименовать его).

2.9.3. Выход без сохранения изменений

Команда QUIT (ПОКИНЬ) обеспечивает возврат к Главному Меню без обновления файла чертежа. Если вы только просматриваете чертеж или не хотите записывать внесенные в чертеж изменения, то следует выйти из графического редактора с помощью команды QUIT (ПОКИНЬ). Чтобы исключить случайный ввод команды и не потерять все сделанное в течение сеанса работы, перед исполнением команды система спрашивает:

```
Command: QUIT
Really want to discard all changes to drawing?
```

```
Команда: ПОКИНЬ
Действительно не нужны все изменения в рисунке?
```

Если вы отвечаете "ДА" или "Д", то после обработки команды ПОКИНЬ на экран выводится главное меню и файл .dwg остается в том виде, который он принял в результате действия последней команды SAVE (СОХРАНИ).

Система для выполнения чертежей в Автокаде с учетом требований ЕСПД

- Выполнение чертежей в Единой Системе Проектной Документации
- Быстрая перенастройка ранее выполненных чертежей с последующими изменениями ГОСТа и требований ЕСПД
- Оперативное составление спецификаций и заполнение штампов в автоматическом и диалоговом режиме
- Значительное сокращение сроков выполнения чертежей и повышение качества разработки и оформления сопровождающей документации

Характеристики и возможности

- Простановки размеров, соответствующая требованиям ЕСПД
- Черчение сдвоенных линий, полых стен и боксов сдвоенными линиями
- Автоматическое размещение окон и дверей в стенах, а также их передвижение и удаление без нарушения сплошности стен
- Расширенное меню с простым доступом к командам, повышающим скорость черчения, например: переброска фрагментов чертежа в нерегенерируемые стеки с целью сокращения времени регенерации чертежа, размещение текста вдоль радиусной кривой, удобный выбор параметров штриховки и ее размещение, быстрый ввод стандартных форматов с заполнением штампа и привязки к чертежу, оперативный набор собственной базы символов, знаков, форм с их отображением в меню и многое другое
- Собственные файлы с дополнительной библиотекой чертежных символов, строительных элементов, санитарных, электрических и механических фитингов, мебели, вспомогательных приспособлений и деталей, используемых на месте работ, изображений лестниц, эскалаторов, оконных и дверных проемов, плит потолка и многих других деталей, а также наименований часто используемых надписей и материалов по ГОСТу
- Возможность просмотра и редактирования файлов шрифтов и дополнительной библиотеки
- Возможность наполнения отраслевой базы фрагментов чертежей (AutoГОСТ)
- Максимальный объем пакета не превышает 1,2 Мбайт

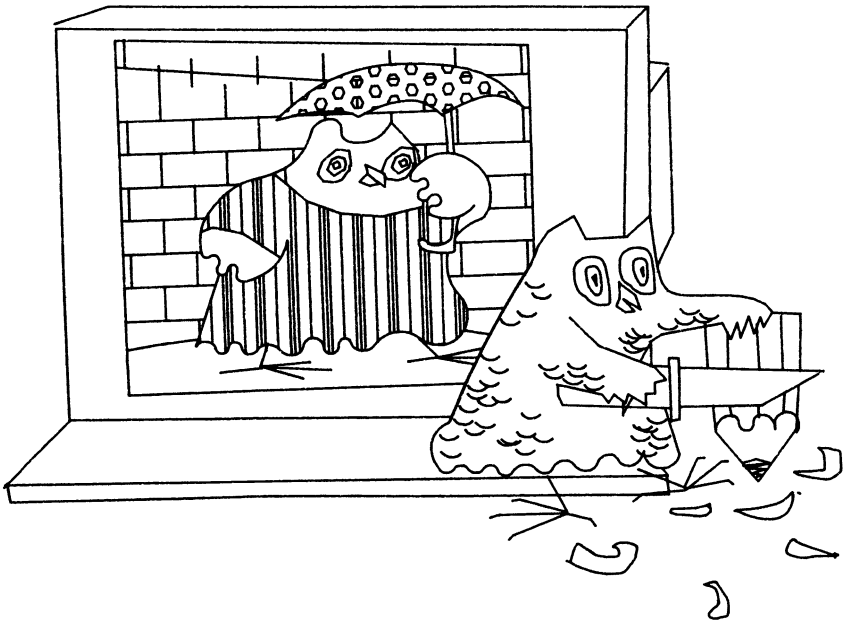
Требования к оборудованию и программному обеспечению

- Персональная ЭВМ типа IBM PC/XT, PC/AT, серии PS/2 и PC-386
- Операционная система MS DOS версии 3.0 и выше
- Пакет автоматизированного проектирования AutoCAD любого варианта версии 10

Пакет AutoСПДС придает новые возможности системе AutoCAD

Научно-производственный
и учебно-консультационный Центр
совместного советско-американского предприятия "Диалог"
на базе Московского инженерно-физического института

СССР, 115409 Москва, ул. Москворечье, 31-2
Телефон: (095) 3243055, (095) 3247166
Телефакс: (095) 3243055



ГЛАВА 3

СОЗДАНИЕ ЧЕРТЕЖА

Автокад вносит в работу конструктора элементы современной технологии обработки данных. Вычислительные возможности компьютера используются уже на самом первом этапе создания чертежа. С одной стороны, это широкое использование вычислений, производимых в процессе геометрических построений изображений, состоящих из отрисовки примитивов традиционного черчения (отрезков, дуг, многоугольников и др.) и последующего их редактирования. С другой стороны, специфика компьютерной графики привела к использованию новых объектов, таких, как фигура, полилиния, блок и др. С этими новыми объектами связаны новые понятия, способы работы и возможности.

Так, например, полилиния позволяет автоматически строить фаски и сопряжения, легко варьировать ширину линии и пр.

Кроме преимуществ, укладывающихся в рамки традиционного черчения, САПР предоставляет возможность создания архива типовых фрагментов чертежей и использования его при создании новых разработок (блоки).

Существенным преимуществом компьютерной графики является автоматизация большинства рутинных операций, связанных с оформлением чертежной документации: штриховки, нанесения размеров, выполнения надписей, отрисовки штампов и т.п. Эти операции также легко автоматизируются в рамках "идеологии" Автокада введением новых типов примитивов: блока, формы и текста.

3.1. Примитивы как элементы чертежа

В системе Автокад любое изображение создается с помощью базового набора примитивов, перечисленных в главе 1 (стр. 10). Каждый примитив создается "своей" командой; все команды отрисовки базовых примитивов находятся в подменю DRAW (РИСУЙ). Каждая команда предоставляет пользователю несколько способов построения одного и того же объекта по заданным геометрическим параметрам, производя необходимые вычисления непосредственно в процессе отрисовки. Например, окружность может быть задана по трем лежащим на ней точкам, по центру и радиусу, по центру и диаметру и пр., в зависимости от задачи. Для того чтобы быстро и качественно строить чертеж, следует знать все возможности, предоставленные командами отрисовки, и в каждом конкретном случае выбирать наиболее удобный способ решения задачи.

Для того чтобы вам удобнее было работать, Автокад в процессе отрисовки показывает вам то, что получится в результате завершения команды. Такой режим работы называется режимом слежения и может быть включен или выключен. Установка режима слежения осуществляется командой DRAGMODE (СЛЕДИ), подробно рассмотренной в следующей главе. В настройке по умолчанию стандартной поставки режим слежения включен.

При указании точек на экране появляются небольшие крестики. Эти крестики (т.н. маркеры) не элемент чертежа, а вспомогательные экранные знаки, указывающие местоположение вводимых точек. Эти знаки будут удалены после любой перерисовки текущего видового экрана (в результате действия команд REDRAW (ОСВЕЖИ), REDRAWALL (ВСЕОСВЕЖИ), REGEN (РЕГЕН), REGNALL (ВСЕРЕГЕН). В файл чертежа маркеры не записываются и на печать не выводятся. Так же как и режим слежения, отрисовка маркеров при указании точек может быть включена или выключена. Управление режимом отображения маркеров осуществляется командой BLIPMODE (МАРКЕР), которая рассматривается в следующей главе. В настройке по умолчанию стандартной поставки режим отображения маркеров включен.

3.1.1. Точка

Точка - самый простой геометрический объект. Геометрическая точка характеризуется тремя координатами. Точка как примитив Автокада имеет дополнительные свойства (общие свойства примитивов - принадлежность слою, цвет, высота). На экране точка может отображаться не только светлым (темным) пятнышком, но и различными знаками (их около 20).

Не следует путать точку с маркером: точка - примитив, элемент чертежа; к точке можно привязаться как к объекту (режим привязки NODE (УЗЕЛ); с ней можно производить многие операции редактирования; точки играют вспомогательную роль при построении чертежа. В Автокаде точка используется, например, командами DIVIDE (ПОДЕЛИ) и MEASURE (РАЗМЕТЬ).

Примитив точка отрисовывается командой POINT (ТОЧКА):

```
Command: POINT
Point:
```

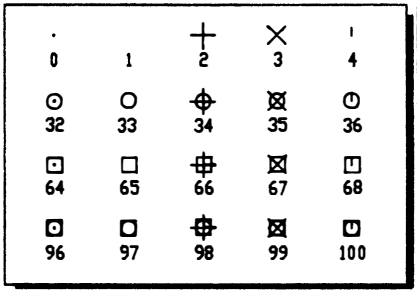
```
Команда: ТОЧКА
Точка:
```

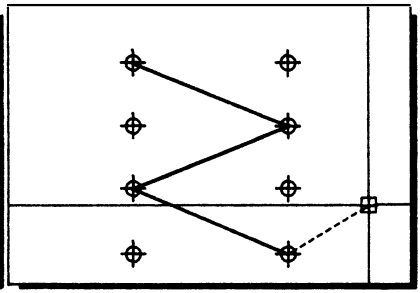
В ответ на запрос надо ввести координаты точки с клавиатуры (два числа, разделенных запятой; вместо десятичной запятой используйте точку) или непосредственно указать точку в графической зоне экрана, поместив курсор в нужное место и нажав на левую клавишу "мыши". При указании точки курсором можно воспользоваться числовыми значениями координат перекрестья курсора, которые отображаются в строке состояния. Напомним, что эти значения отражают абсолютные координаты перекрестья курсора в текущей системе координат.

Точка характеризуется тремя координатами, но при указании точки с помощью "мыши" можно непосредственно задавать только две координаты - X и Y. Координата Z берется при этом равной нулю (если в режимах создания примитивов установлена нулевая высота) в текущей системе координат. Заметим, что в Автокаде имеется специальное средство раздельного ввода значений координат с помощью "мыши", т.н. фильтры. О них мы поговорим позже.

При указании числовых значений координат с клавиатуры также могут быть заданы только две координаты; координата Z также принимается равной нулю.

В системе предусмотрено отображение точек 20 различными графическими знаками:





Вид знака отображения точек задается значением системной переменной PDMODE, а размеры - переменной PDSIZE. На приведенном выше рисунке число под изображением знака соответствует значению переменной PDMODE.

Значения системных переменных PDMODE и PDSIZE можно устанавливать, как и все другие системные переменные, с помощью команды SETVAR (УСТПЕРЕМ). Гораздо удобнее, однако, воспользоваться экранным меню команды POINT (ТОЧКА), в котором содержатся одноименные опции, позволяющие устанавливать новые значения этих переменных.

Если вы забыли коды точек, выберите из экранного меню пункт Complex Points example: (Примеры сложных точек:). Для удаления примера используйте пункт Remove example: (Удали примеры:) или команду REDRAW (ОСВЕЖИ).

Предположим, вы хотите рисовать точками типа 34. Вызвав команду POINT (ТОЧКА), в ответ на запрос Point: (Точка:) выберите из экранного меню опцию Pdmode (Тчкреж). При этом система автоматически использует в прозрачном режиме команду SETVAR (УСТПЕРЕМ) для изменения типа точки:

```
Command: POINT
Point: 'SETVAR>>Variable name or ?: PDMODE
>>New value for PDMODE <0>: 34
Resuming POINT command.
Point:
```

```
Команда: ТОЧКА
Точка: 'УСТПЕРЕМ>>Имя переменной или ?: PDMODE
>>Новое значение PDMODE <0>: 34
Возобновляю команду ТОЧКА.
Точка:
```

Аналогично изменяется размер знаков отображения точек (в меню это пункт Pdsize (Тчкразм).

Можно использовать точки как вспомогательное средство для сложных геометрических построений. Удобно размещать их на отдельном слое и назначать отличный от других объектов цвет - такую методику можно применять при работе с командами DIVIDE (ПОДЕЛИ) и MEASURE (РАЗМЕТЬ). Закончив работу с точками как со вспомогательными примитивами, необязательно стирать все точки: можно просто отключить соответствующий слой или назначить ему режим отображения 1.

3.1.2. Отрезок

Отрезок прямой является одним из наиболее простых и часто используемых примитивов. Кроме общих свойств отрезок характеризуется двумя точками, задавать которые можно любым из известных в Автокаде способов:

- вводом координат (абсолютных или относительных) точек с клавиатуры:
  - 10,10 абсолютные декартовы координаты
  - @10,10 относительные декартовы координаты (относительно предыдущей точки)
  - @10<10 относительные полярные координаты (длина и направление нового сегмента в полярной системе координат, начало которой находится в предыдущей точке)
- указанием точек прямо в поле чертежа (графической зоне) графическим курсором с помощью устройства указания (можно использовать режимы SNAP (ШАГ), ORTHO (ОРТО), фильтры и объектные привязки)
- любыми комбинациями этих способов

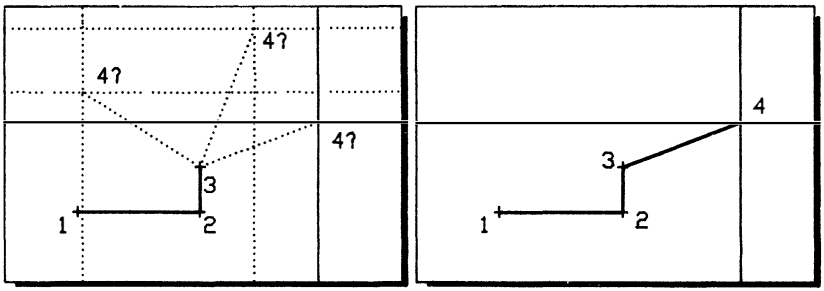
Приведем пример использования команды LINE (ОТРЕЗОК) для отрисовки повернутого на 45 градусов квадрата. Применим три возможных способа задания координат точек с клавиатуры: первая колонка иллюстрирует ввод абсолютных координат, вторая - относительных декартовых, а третья - относительных полярных координат. В четвертой колонке используются все три способа:

Command: **LINE**			
From point: **30,10**	**30,10**	**30,10**	**30,10**
To point: **10,30**	**@-20,20**	**@28.2<135**	**@28.2<135**
To point: **30,50**	**@20,20**	**@28.2<45**	**@20,20**
To point: **50,30**	**@20,-20**	**@28.2<-45**	**50,30**
To point: **Close**	**Close**	**Close**	**30,10**
To point: **ПРОБЕЛ** или **RETURN**			

Команда: **ОТРЕЗОК**			
От точки: **30,10**	**30,10**	**30,10**	**30,10**
К точке: **10,30**	**@-20,20**	**@28.2<135**	**@28.2<135**
К точке: **30,50**	**@20,20**	**@28.2<45**	**@20,20**
К точке: **50,30**	**@20,-20**	**@28.2<-45**	**50,30**
К точке: **Замкни**	**Замкни**	**Замкни**	**30,10**
К точке: **ПРОБЕЛ** или **RETURN**			

Обратите внимание на то, что команда LINE (ОТРЕЗОК) не завершается после указания второй точки, а продолжает запрашивать одну точку за другой, позволяя строить непрерывную ломаную линию. Для того чтобы вам построить два отдельных отрезка, нужно завершить команду, нажав клавишу ПРОБЕЛ или RETURN, а затем снова вызвать команду LINE (ОТРЕЗОК). Кстати, еще раз вызвать предыдущую команду можно нажатием клавиши RETURN в ответ на запрос "Command:" ("Команда:"). Если, вызвав повторно команду LINE (ОТРЕЗОК), не вводить первую точку, а еще раз нажать клавишу RETURN, то в качестве первой точки берется последняя точка предыдущего отрезка (или дуги). Тем самым мы вызываем опцию Continue (Продолжи), которую можно также вызвать явно.

При отрисовке отрезков при перемещении по полю чертежа перекрестия за ним тянется т.н. "резиновая линия", показывающая положение будущего сегмента. Сегмент фиксируется вводом координат последней точки с клавиатуры или указанием точки на экране, для чего нужно нажать левую клавишу "мыши".



Последний отрисованный сегмент ломаной может быть отменен в процессе отрисовки указанием на опцию Undo (Отмени) в экранном меню команды LINE (ОТРЕЗОК). Опцию отмены можно ввести также с клавиатуры. Отменяя последовательно сегменты, можно отменить всю ломаную.

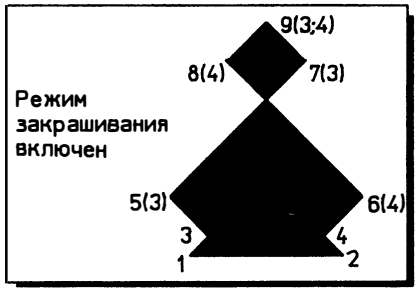
Для удаления нарисованных отрезков используйте команды ERASE (СОТРИ) и UNDO (ОТМЕНИ) (см. главу 5 "Простое редактирование").

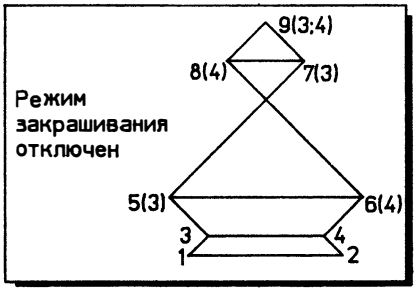
В последующих главах вы убедитесь в том, какие богатые возможности существуют для изменения (редактирования) даже такого простого объекта, как отрезок (отрезок можно удлинять, укорачивать, обрывать, размыкать, сопрягать, поворачивать, строить перпендикулярные и параллельные отрезки и пр.).

3.1.3. Фигура

Автокад не умеет "заливать" замкнутую область цветом, как это делает, например, графический редактор Paintbrush: это связано с ограничениями, налагаемыми общей "идеологией" системы. Для построения закрашенной области в Автокаде существует специальный примитив - фигура. Фигура - это область, ограниченная четырехугольником (или треугольником, если две вершины совпадают). Эта область может быть закрашенной или незакрашенной.

Фигура строится командой SOLID (ФИГУРА), которой требуется указать четыре точки: обратите внимание на последовательность задания вершин:





Command: **SOLID**
First point: **10,0**
Second point: **90,0**
Third point: **20,10**
Fourth point: **80,10**
Third point: **0,30**
Fourth point: **100,30**
Third point: **70,100**
Fourth point: **30,100**
Third point: **50,120**
Fourth point: **RETURN**(взять третью точку как четвертую)
Third point: **RETURN**(отмена команды)

Команда: **ФИГУРА**
Первая точка: **10,0**
Вторая точка: **90,0**
Третья точка: **20,10**
Четвертая точка: **80,10**
Третья точка: **0,30**
Четвертая точка: **100,30**
Третья точка: **70,100**
Четвертая точка: **30,100**
Третья точка: **50,120**
Четвертая точка: **RETURN**(взять третью точку как четвертую)
Третья точка: **RETURN**(отмена команды)

Закрашиваться могут и другие примитивы, имеющие ширину, - полоса и полилиния (и ее производные, например кольцо). Закраска не свойство примитива, а режим отображения объекта на экране, который управляется значением системной переменной FILLMODE (1 - закраска включена, 0 - отключена). Хотя вы можете включать и выключать режим закрашивания несколько раз в процессе построения фигур (получая, таким образом, на одном экране закрашенные и незакрашенные фигуры), после регенерации рисунка для всех закрашиваемых примитивов будет отработано последнее состояние режима закраски.

3.1.4. Полоса

Полоса является объектом, производным от фигуры. Полоса - это фигура постоянной ширины, с помощью которой легко отрисовать сплошную ломаную линию:

Полоса отрисовывается командой TRACE (ПОЛОСА):

```
Command: TRACE
Trace width <значение по умолчанию>: 0.5
From point:
To point:
To point:
```

```
Команда: ПОЛОСА
Ширина полосы <значение по умолчанию>: 0.5
От точки:
К точке:
К точке:
```

Хотя на самом деле Автокад хранит в памяти четыре вершины полосы, при отрисовке задается ширина полосы и воображаемые конечные точки, лежащие на осевой линии. Ширина полосы запрашивается один раз при входе в команду (ее можно ввести не только с клавиатуры, но и указанием двух точек на экране); поэтому смежные сегменты полосы при отрисовке не могут быть сделаны разной ширины (хотя средствами редактирования или Автолиспа полосу легко изменить до неузнаваемости).

Автокад автоматически "подрезает" концы смежных сегментов, поэтому отрисовка каждого сегмента производится только после того, как будет задана конечная точка следующего сегмента или не будет завершена команда. Торцевые концы полосы всегда строятся перпендикулярно осевой линии.

Начальное значение ширины полосы хранится в системной переменной TRACEWID, которая по окончании сеанса редактирования записывается в файл чертежа.

У команды TRACE (ПОЛОСА) нет никаких опций, это очень простой объект. Мы рекомендуем вместо полос использовать полилинии, которые предоставляют гораздо больше возможностей при редактировании чертежа.

3.1.5. Дуга

Дуги окружностей строятся командой ARC (ДУГА). Способов построения дуги в Автокаде насчитывается более двенадцати. И хотя не все способы являются существенно различными (некоторые отличаются лишь порядком задания числовых значений параметров), богатство предоставляемых возможностей, безусловно, полезно в процессе построения сложных геометрических чертежей.

Оговоримся сразу, что в некоторых случаях трудно мысленно предсказать результат работы команды, поэтому мы рекомендуем при работе с дугами всегда включать режим слежения (см. следующую главу).

Приведем пример вызова команды ARC (ДУГА). Обратите внимание на то, как задается способ отрисовки дуги:

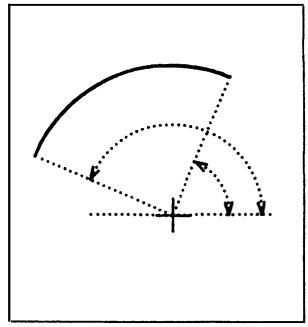
```
Command: ARC
Center/<Start point>: C
Center: 100,100
Start point: 200,200
Angle/Length of chord/<End point>: A
Included angle: -30
```

```
Команда: ДУГА
Центр/<Начальная точка>: Ц
Центр: 100,100
Начальная точка: 200,200
Угол/Хорда/<Конечная точка>: У
Центральный угол: -30
```

Как видим, последовательно предлагается несколько способов построения дуги. В ответ на каждый запрос нужно либо задать точку (любым из известных способов), либо ввести буквенный код, означающий выбор опции.

Способы построения дуги выбираются по буквенным кодам. Буквенные коды для английской и русской версий приведены в таблице:

S	Starting point	Н	Начальная точка
E	End point	К	Конечная точка
R	Radius	Р	Радиус
A	Angle	У	центральный Угол
L	chord Length	Х	длина Хорды
D	starting Direction	На	начальное Направление

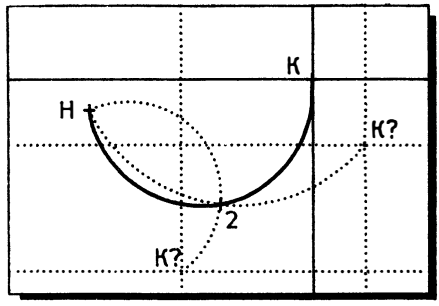


Дуга математически определяется четырьмя параметрами - центром, радиусом и двумя центральными углами (см. рисунок справа). При построении дуги на чертеже удобно, однако, использовать другой набор параметров (например, центр и две точки или две точки и длина хорды). При этом число параметров уменьшается до трех, поскольку, например, заданием конечной точки определяется сразу и радиус, и соответствующий центральный угол.

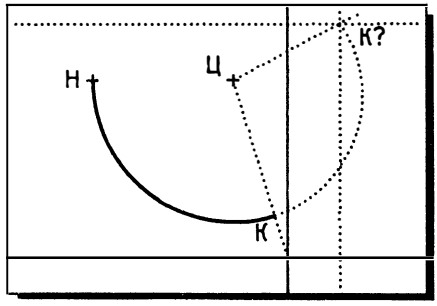
Следует оговориться, что в Автокаде принят ряд соглашений, цель которых - снять возможные неоднозначности, упростить и унифицировать построения. Так, дуги строятся из начальной точки по направлению отсчета углов (против часовой стрелки, если не установлено обратное путем изменения системной переменной ANGDIR). Напомним, что осью отсчета углов считается ось OX текущей системы координат. Если можно провести две дуги, то Автокад выбирает ту, которая имеет наименьшую длину.

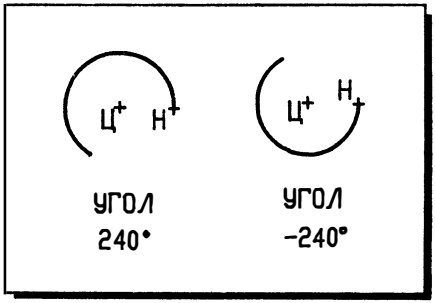
Ниже рассматриваются различные способы построения дуг.

3-point (3 точки)
Построение дуги по трем точкам: начальной, второй и конечной



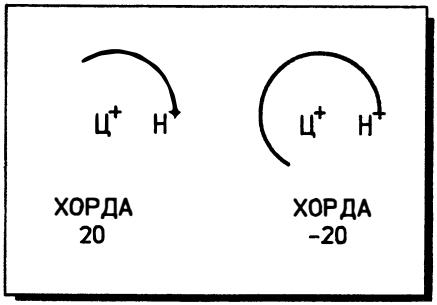
SCE (НЦК)
Построение дуги заданием начальной точки, затем центра и конечной точки





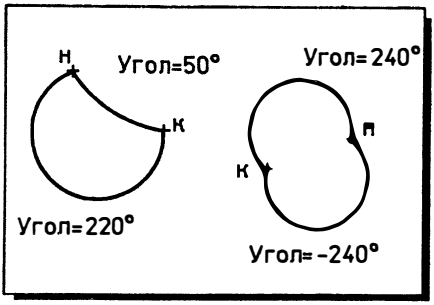
SCA (НЦУ)

Построение дуги заданием начальной точки, затем центра и угла



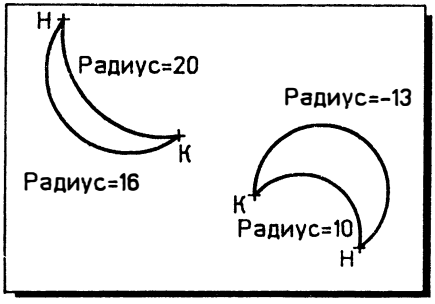
SCL (НЦХ)

Построение дуги заданием начальной точки, центра и хорды



SEA (НКУ)

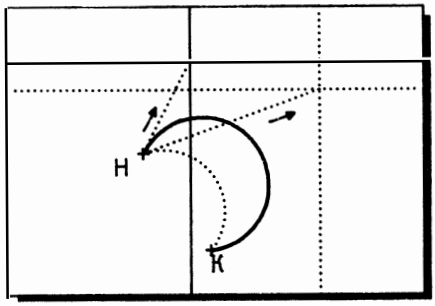
Построение дуги заданием начальной точки, конечной точки и угла



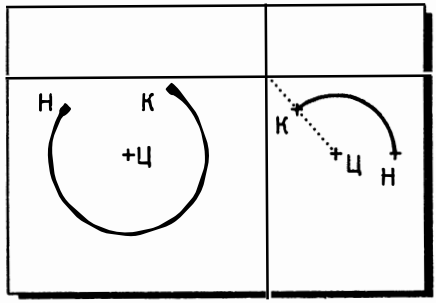
SER (НКР)

Построение дуги заданием начальной точки, затем конечной точки и радиуса

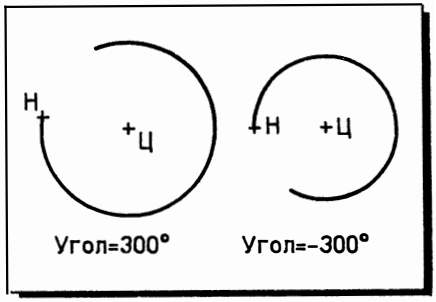
SED (НКНа)
Построение дуги заданием двух точек и направлением касательной



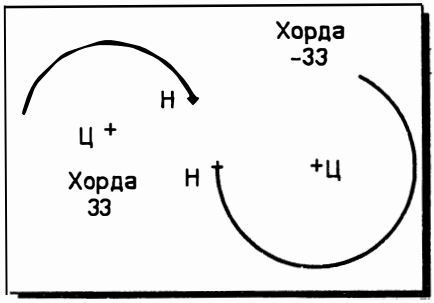
CSE (ЦНК)
Построение дуги заданием центра, начальной точки и конечной точки

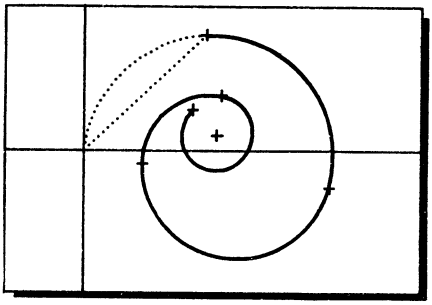


CSA (ЦНУ)
Построение дуги заданием центра, начальной точки и угла

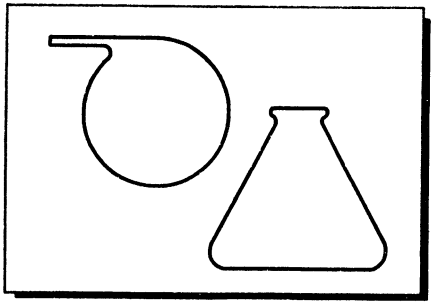


CSL (НКД)
Построение дуги заданием центра, затем начальной точки и длины хорды





Contin (Продолжи)
Эта опция позволяет в качестве первой точки взять конечную точку предыдущей дуги или отрезка



Примеры использования для дуг и отрезков опции Contin (Продолжи)

3.1.6. Круг

Команда CIRCLE (КРУГ) позволяет отрисовать круг пятью способами. Все возможности отрисовки круга реализованы в подменю КРУГ Главного Меню. Перечислим пункты подменю КРУГ

CEN,RAD: (ЦЕН,РАД:) - построение окружности по центру и радиусу (или точке, принадлежащей окружности)
CEN,DIA: (ЦЕН,ДИА:) - окружность строится по центру и диаметру
2POINT: (2 ТОЧКИ:) - построение окружности, проходящей через две точки, определяющие ее диаметр
3POINT: (3 ТОЧКИ:) - строится окружность, проходящая через три точки, не лежащие на одной прямой
TTR: (КасКасР:) - строится окружность заданного радиуса, касательная двум прямым

Вызывая команду с клавиатуры для выбора режима отрисовки, следует, так же как это делалось при работе с дугой, вводить буквенные коды. Поскольку их комбинаций сравнительно немного, мы не будем их описывать подробно.

Вызовите команду CIRCLE (КРУГ) с клавиатуры:

```
Command: CIRCLE
3P/2P/TTR/<Center>:
```

```
Команда: КРУГ
3Т/2Т/ККР/<Центр>:
```

Если вы задали точку центра (такой ответ является ответом по умолчанию), Автокад предложит вам выбрать способ задания размера окружности - диаметром или радиусом:

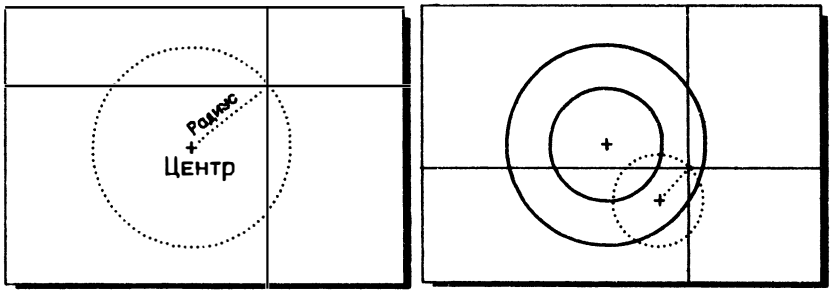
```
Command: CIRCLE
3P/2P/TTR/<Center>: 10,10
Diameter/<Radius>: D
Diameter: 10
```

```
Команда: КРУГ
3Т/2Т/ККР/<Центр>: 10,10
Диаметр/<Радиус>: Д
Диаметр: 10
```

По умолчанию введенное значение будет восприниматься как значение радиуса.

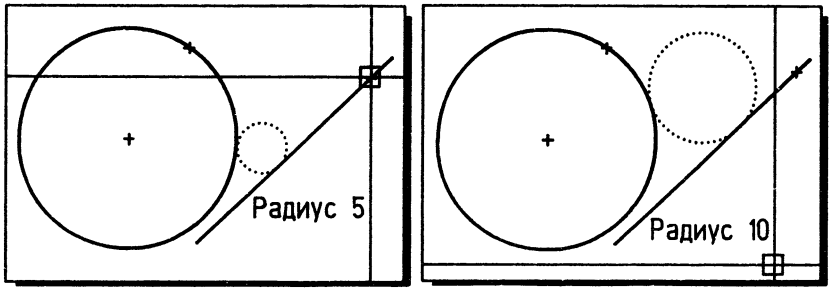
Отметим, что если вместо численного значения радиуса вы вводите координаты точки (безразлично, указываете ли вы точку "мышью" на экране или вводите координаты с клавиатуры), то Автокад отрисует окружность, проходящую через эту точку.

Если вы строите окружность по центру и диаметру и вместо значения диаметра вводите координаты точки, то построенная окружность *не будет* проходить через указанную точку. Это связано с тем, что, указывая вторую точку, вы фактически определяете не точку, а расстояние от центра окружности. Это расстояние при слежении отображается в строке состояния в формате (длина вектора, угол). Поэтому при отрисовке круга удобно пользоваться режимом слежения:



Интересная возможность предоставляется опцией TTR (ККР) команды CIRCLE (КРУГ). Указав сначала один, а затем второй примитив и введя

значение радуса, вы получите круг нужного размера, касающийся двух выбранных вами объектов:



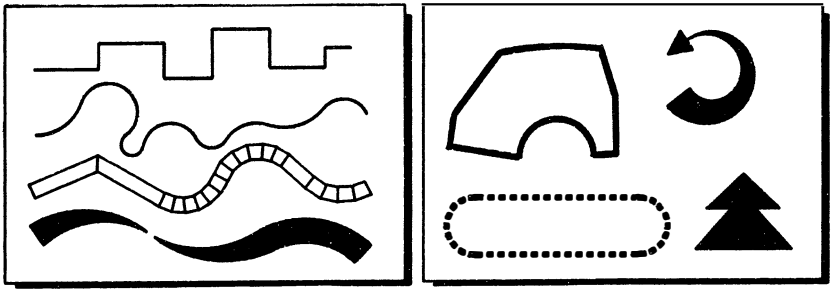
Круг можно построить касательным и к трем объектам, если при построении его по трем точкам указывать объекты с помощью объектной привязки TAN (КАС). Подробнее о привязках см. следующую главу.

3.1.7. Полилиния

Теперь вы уже знаете, как строятся простые примитивы: точки, отрезки, полосы, фигуры, дуги и окружности. В дальнейшей вашей работе с Автокадом вы будете использовать самые разнообразные комбинации этих примитивов. И все-таки мы находимся пока еще на самой поверхности Автокада. Попробуем немного углубиться в возможности этой неисчерпаемой программы.

В этом пункте мы рассмотрим, с нашей точки зрения, самый интересный примитив Автокада - полилинию. Благодаря удачной идее описания данных, принятой в Автокаде, создателям этого пакета удается ввести примитив, который, являясь очень простым с точки зрения Автокада, предлагает в то же время необычайно широкий спектр возможностей с точки зрения пользователя. Рассматривая здесь только двумерные полилинии, оговоримся, что в Автокаде существуют и трехмерные полилинии (о них говорится вкратце в п. 3.1.9).

Ниже изображено несколько двумерных полилиний:



Полилиния - объект, состоящий из списка вершин (vertex) переменной (не определенной заранее) длины, который отображается на экране как совокуп-

ность линейных и дуговых сегментов, отрисовываемых между соседними вершинами. Из этого очень простого определения вытекает ряд новых свойств этого объекта - свойств, которых не было у ранее рассмотренных нами примитивов.

Прежде всего полилиния, даже состоящая из 100 сегментов, воспринимается Автокадом как один объект (сейчас это утверждение не слишком вас впечатляет, но позже вы увидите, как это удобно).

Вторым важным свойством полилинии является то, что ее сегменты могут обладать шириной, причем начальная и конечная ширина может быть разной.

Полилиния - это примитив, на основе которого в Автокаде создаются "производные" примитивы: кольцо, многоугольник, эллипс.

С полилинией, которая является с точки зрения Автокада связным списком точек-вершин, могут быть проделаны следующие операции:

- добавление нового сегмента и удаление существующего
- изменение начальной и конечной ширины существующих сегментов
- автоматическое выполнение фасок и сопряжений указанных параметров
- сглаживание полилинии дугами окружностей или сплайнами различных типов
- "расчленение" полилинии: преобразование в более простые примитивы - дуги и отрезки, при этом часть информации, например о ширине, может утрачиваться
- преобразование отрезков и дуг в полилинии с последующим назначением им, например, толщины
- вычисление площади, ограниченной полилинией, и ее периметра

Многие преимущества, связанные с использованием полилиний, нельзя осознать вне связи с командами редактирования. Позже вы узнаете, как отредактировать полилинию. Сейчас скажем только, что такой интересный примитив имеет в Автокаде "собственную" команду редактирования: PEDIT (ПОЛРЕД).

Для отрисовки полилинии существует команда PLINE (ПЛИНИЯ), которая, как и другие команды отрисовки, находится в меню DRAW (РИСУЙ):

```
Command: PLINE
From point:
```

```
Команда: ПЛИНИЯ
От точки:
```

После введения начальной точки двумерной полилинии выдается текущее значение ее ширины и предлагается ответить на следующий запрос:

```
Current line-width is 2.89
Arc/Close/Halfwidth/Length/Undo/
Width/<Endpoint of line>:
```

```
Текущая ширина линии равна 2.89
ДУга/Замкни/Полуширина/ДЛина/ОТМени/
Ширина/<Конечная точка сегмента>:
```

Текущая ширина полилинии определяет ширину всех отрисовываемых сегментов до тех пор, пока не будет задана другая ширина.

Как видим, у команды существует большое количество опций, управляющих режимами отрисовки. По умолчанию считается, однако, что пользователь введет конечную точку следующего прямолинейного сегмента. В этом режиме команда PLINE (ПЛИНИЯ) работает внешне так же, как команда LINE (ОТРЕЗОК). Другими ответами на подсказку команды является ввод буквенных кодов возможных опций, эти коды выделены в строке подсказок прописными буквами (например, для перехода в режим отрисовки дуг в ответ на запрос следует ввести буквы "ДУ"). Рассмотрим опции этого запроса.

Arc (ДУга)

Переключает команду PEDIT в режим отрисовки дуг, подсказка и опции которого рассматриваются ниже

Close (Замкни)

Эта опция управляет очень интересным свойством полилинии. Полилиния может быть замкнутой и незамкнутой. Еще раз отметим, что с точки зрения Автокада полилиния представляет собой список вершин, между которыми отрисовываются прямолинейные или дуговые сегменты. Если полилиния замкнута, то сегмент отрисовывается также между начальной и конечной точкой полилинии. Таким образом, чтобы замкнуть контур полилинии, необязательно явно отрисовывать замыкающий сегмент - достаточно указать опцию Close (Замкни), и Автока,., е создавая ни одного нового объекта, отрисует замыкающий сегмент (если полилиния замыкается из режима отрисовки отрезков, то замыкающий сегмент будет отрезком, если замыкается из дуг - дугой). Этим свойством полилиний можно также управлять при помощи команды PEDIT (ПОЛРЕД). Опция Close (Замкни) завершает команду

Length (ДЛина)

Позволяет строить сегмент заданной длины в том же направлении, что и предыдущий. Если предыдущий сегмент является дугой, то новый линейный сегмент будет касательным к этой дуге

Undo (ОТМени)

Позволяет отменять последний сегмент. (Не путайте опцию Undo (ОТМени) с командой UNDO (ОТМЕНИ) - последняя служит для отмены команд целиком.) Отменять сегменты можно до тех пор, пока не будет отменен первый сегмент полилинии

Width (Ширина)

Позволяет задавать начальную и конечную ширину последующего сегмента. Значение ширины, равное нулю, приводит к отрисовке минимально различимой на дисплее линии. Ненулевые значения ширин позволяют создавать объекты, подобные полосам и фигурам. Введенное значение начальной ширины берется как значение по умолчанию для конечной ширины. Аналогично значение конечной ширины берется как значение по умолчанию для начальной

ширины следующего сегмента. При отрисовке широкой полилинии считается, что задаются точки на оси полилинии. Угловые стыки широких смежных сегментов полилинии обычно подрезаются. Исключением является случай, когда линейный сегмент не является касательным к смежным дуговым сегментам

Halfwidth (Полуширина)

Позволяет удобно воспользоваться устройством указания, закрепляя один конец "резиновой нити" на оси широкого линейного сегмента. В остальном действие опции аналогично описанной выше

Полилиния обрабатывается при закраске так же, как фигура и полоса. Напомним, что закраска - не свойство примитива, а режим отображения объекта на экране, который управляется значением системной переменной FILLMODE (1 - закраска включена, 0 - отключена).

Если выбрать опцию Arc (ДУга), то команда PLINE (ПЛИНИЯ) переходит в режим отрисовки дуговых сегментов. При этом вид запроса изменяется:

```
Angle/CEnter/CLose/Direction/Halfwidth/Line/Radius/
Second pt/Undo/Width/<End point of arc>:
```

```
Угол/Центр/Замкни/Направление/Полуширина/ОТРезок/Радиус/
Вторая/ОТМени/Ширина/<Конечная точка дуги>:
```

Как видим, опять по умолчанию ожидается ввод второй точки сегмента, на этот раз дугового. Наряду с уже знакомыми общими опциями полилинии, действие которых описано выше, при отрисовке дуг Автокад предлагает дополнительные возможности.

Дуговой сегмент по умолчанию проводится касательным к предыдущему отрезку (если дуга является первым сегментом полилинии, то по умолчанию предлагается направление последнего отрисованного отрезка, дуги или полилинии). Если требуется отрисовать другую дугу, то следует воспользоваться одной из опций (обратите внимание на сходство этой опции с командой ARC (ДУГА):

Angle (Угол)

Позволяет задать центральный угол дуги, после чего можно, как и при построении дуг, определить либо конечную точку, либо радиус, либо центр

Center (Центр)

Позволяет задать центр дуги, после чего можно задать либо угол, либо длину, либо конечную точку

Direction (Направление)

Позволяет задать направление касательной в начальной точке дуги (по умолчанию дуга почти всегда стрсится по касательной к предыдущему сегменту)

Radius (Радиус)

Позволяет задать радиус дуги, затем можно задать угол или конечную точку

Second pt (Вторая точка)

Запрашиваются вторая и третья точки дуги, которая строится по трем точкам

Как вы уже, наверное, видите, полилиния - весьма богатый объект, и мы не ошибемся, если скажем, что это основной примитив графического редактора Автокада.

На этом месте мы рекомендуем вам прервать чтение нашей книги и попробовать в качестве упражнения повторить полилинией все объекты, приводимые в иллюстрациях к командам LINE (ОТРЕЗОК), SOLID (ФИГУРА), TRACE (ПОЛОСА) и ARC (ДУГА).

3.1.8. Многоугольник, эллипс и кольцо

Как вы, наверное, уже убедились, с помощью полилинии можно строить самые разнообразные объекты. На основе полилинии в Автокаде реализовано автоматизированное построение некоторых правильных геометрических объектов - многоугольника, эллипса и кольца.

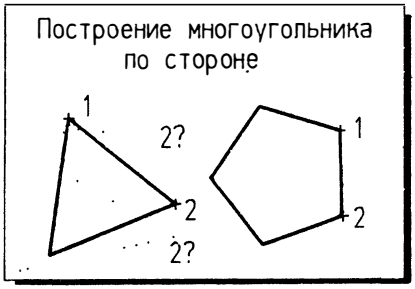
Эти примитивы отрисовываются соответственно командами POLYGON (МНОГОУГОЛ), ELLIPS (ЭЛЛИПС) и DONUT (КОЛЬЦО).

Мы не будем здесь рассматривать указанные примитивы слишком подробно - попробуйте исследовать эти простые объекты сами. Перечислим только основные возможности команд.

Многоугольник

Команда POLYGON (МНОГОУГОЛЬНИК) реализует три способа построения правильного многоугольника. Задав число сторон многоугольника, вы можете построить его:

- по центру и радиусу описанной окружности
- по центру и радиусу вписанной окружности
- по стороне



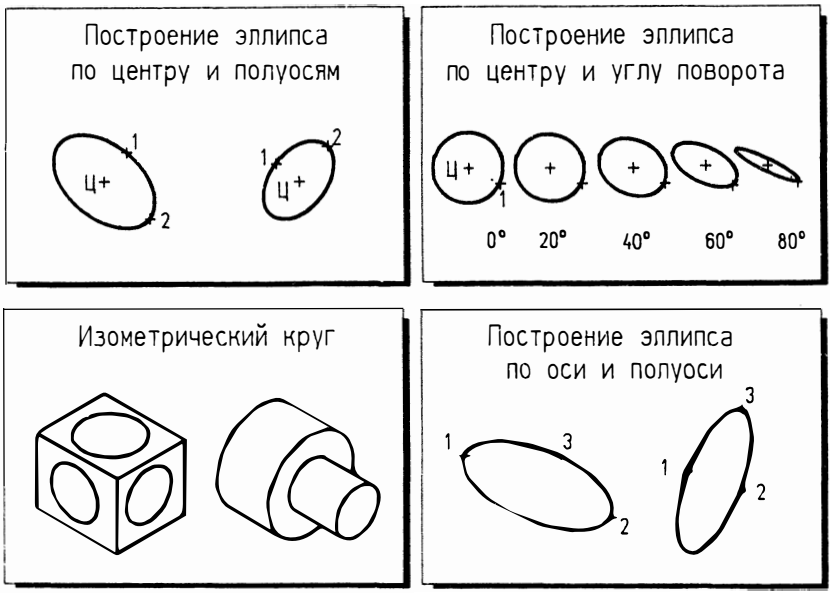
При построении многоугольника по стороне считается, что центр правильного многоугольника лежит слева по направлению движения от первой задаваемой точки ко второй. При построении многоугольника создается полилиния нулевой

ширины, вершины которой "пронумерованы" (по порядку создания) против часовой стрелки.

Эллипс

Команда ELLIPSE (ЭЛЛИПС) строит эллипс одним из четырех способов:

- по оси и полуоси
- по оси и углу поворота воображаемого круга относительно плоскости построений (как известно, в проекции круга получается эллипс)
- по центру и двум полуосям
- по центру, длине полуоси и углу поворота воображаемого круга относительно плоскости построений



Эллипс, так же как и многоугольник, отрисовывается полилинией, т.е. дугами окружностей. В действительности, таким образом, Автокад отрисовывает не эллипс, а двенадцатицентровый овал.

Такая высокая точность построений иногда мешает - например, если вы хотите при помощи объектной привязки построить касательную к эллипсу, то нужно примерно правильно указывать предполагаемую точку касания, иначе вы рискуете указать на другую дугу, на которой точка касания не будет найдена (привязка не воспринимает полилинию как единый объект).

Эллипс используется и для изображения окружностей в изометрии. Команда ELLIPSE (ЭЛЛИПС) при включенном режиме изометрии выдает дополнитель-

ный запрос "Isocircle:" ("Изокруг:"), по выборе которого в соответствии с текущей изометрической плоскостью автоматически отрисуется изометрический круг.

Кольцо

Кольцо в Автокаде отрисовывается командой DONUT (Кольцо). Эта команда отрисовывает замкнутую полилинию ненулевой ширины, состоящую из одного дугового сегмента, представляющего собой полуокружность. Команда запрашивает сначала внутренний диаметр, затем внешний диаметр, после чего просит указать центр кольца. Задав нулевую ширину внутреннего кольца, этой командой можно отрисовывать закрашенные круги.

3.1.9. Трехмерные объекты

10-я версия Автокада отличается развитой трехмерной графикой. Известно, какой большой объем вычислений требуется для работы с трехмерной графикой; и Автокад выделяется среди других пакетов САПР, в частности, тем, что успешно реализует эту сложную задачу на микроЭВМ.

Подобно тому как на плоскости любой чертеж Автокада состоит из ограниченного числа типов примитивов, трехмерные объекты также могут в Автокаде состоять только из трех типов примитивов - трехмерной полилинии, трехмерной грани и трехмерной сети.

Несмотря на такую, казалось бы, ограниченную постановку задачи, создатели Автокада сумели на основе этой "идеологии" достичь очень многого: Автокад позволяет строить сложные трехмерные поверхности - поверхности вращения, поверхности, определенные двумя кривыми, поверхности сдвига и пр. Двумерная графика ранних версий Автокада обобщается теперь в трехмерную графику за счет введения такого понятия, как текущая трехмерная система координат.

Разбирая команды создания трехмерных примитивов, вы убедитесь в том, что они чрезвычайно просты. Оговоримся, что, несмотря на простоту команд построения трехмерных примитивов, техника работы с трехмерными изображениями в Автокаде достаточно сложна и описание ее требует большого объема, и из-за недостатка места мы не будем сейчас углубляться в этот вопрос, а лишь рассмотрим вкратце команды создания трехмерных примитивов.

Трехмерная полилиния

Трехмерная полилиния создается с помощью команды 3DPOLY (3-ПОЛИ), находящейся на второй странице стандартного экранного меню.

С помощью команды создаются трехмерные полилинии, состоящие только из линейных сегментов. Дуговые сегменты в трехмерную полилинию не введены, по-видимому, из-за сложности обработки пространственных дуг. После ввода команды Автокад запрашивает начальную точку 3М полилинии:

```
Command: 3DPOLY
From point:
```

```
Команда: 3-ПОЛИ
От точки:
```

Трехмерную точку можно задать любым из возможных способов. После ввода точки Автокад выдает запрос:

```
Close/Undo/<Endpoint of line>:
```

```
Замкни/Отмени/<Конец сегмента>:
```

Ответом по умолчанию является точка. Можно вводить сколько угодно новых точек.

Трехмерную полилинию можно также отредактировать командой PEDIT (ПОЛРЕД).

Трехмерная грань

Трехмерная грань аналогична фигуре с той разницей, что вершины грани могут иметь различные координаты по оси Z. Однако основное свойство граней - это способность Автокада "воспринимать" грани как непрозрачные пространственные объекты и удалять скрытые линии.

Трехмерные грани создаются командой 3DFACE (3-ГРАНЬ). Грань, как и фигура, может состоять только из четырех точек, однако в отличие от команды SOLID (ФИГУРА) порядок задания точек в команде 3DFACE (3-ГРАНЬ) более естественный - точки задаются обходом контура грани по или против часовой стрелки. Все точки трехмерной грани следует задавать лежащими в одной плоскости, иначе трехмерная грань становится прозрачной для команды СКРОЙ и ее использование теряет смысл.

В команде 3DFACE (3-ГРАНЬ) есть средство управления видимостью каждого края грани, это позволяет моделировать отверстия и многоугольные сегменты плоскостей. Чтобы какой-нибудь край трехмерной грани сделать невидимым, первую точку этого края необходимо ввести с предшествующим признаком I(Н) - Invisible (Невидимый) вне зависимости от способа ввода точки.

Отображением невидимых краев трехмерных граней управляет системная переменная SPLFRAME. Если ей присвоить нулевое значение, то все невидимые края граней станут видимыми и их можно будет редактировать как видимые.

Трехмерные сети

Многоугольная сеть играет в трехмерной графике такую же роль, как полилиния в двумерной графике. Мы имеем в виду тот факт, что, с одной стороны, сеть может быть, как и полилиния, расчленена на более простые объекты (трехмерные грани), а, с другой стороны, многоугольная сеть может порождать различные трехмерные объекты (это позволяет автоматизировать отрисовку часто встречающихся форм). Многоугольная сеть - это двумерный (MxN) массив, аппроксимирующий криволинейные поверхности. Точностью аппроксимации криволинейных поверхностей можно управлять, задавая плотность сети. Плотность сети по M и N можно изменять, устанавливая соответственно значения системных переменных TABSURF1 и TABSURF2.

Оговоримся, что сеть - единственный способ представления трехмерных поверхностей в Автокаде.

Многоугольная сеть имеет и другие сходства с полилинией: может быть замкнутой или разомкнутой, многоугольная сеть является единым объектом. Многоугольная сеть может быть отредактирована большинством команд редактирования.

Ниже перечислены команды, позволяющие создавать многоугольные сети, применяя различные способы их определения:

EDGESURF (П-СОЕД)
строит многоугольную сеть, аппроксимирующую поверхность соединения между двумя выбранными объектами

REVSURF (П-ВРАЩ)
строит многоугольную сеть, аппроксимирующую поверхность вращения, поворачивая выбранный объект вокруг заданной оси

RULESURF (П-СДВИГ)
строит многоугольную сеть, аппроксимирующую поверхность сдвига, передвигая заданный вектор вдоль выбранной определяющей кривой

TABSURF (П-КРАЙ)
строит многоугольную сеть, аппроксимирующую участок поверхности Кунса, ограниченный четырьмя выбранными кривыми

Из падающего меню DRAW (РИСУЙ) можно вызвать графическое меню трехмерных объектов, которые создаются из трехмерных сетей средствами Автолиспа. Реализованы такие простые формы, как куб, конус, сфера, тор и т.п.

Трехмерная графика Автокада, повторимся, техническая, т.е. имеет все преимущества и недостатки, вообще присущие подходу к трехмерному миру как к набору простых геометрических объектов. Если на плоскости действительно

можно любой чертеж представить как совокупность отрезков и дуг, то в пространстве даже в технике используются поверхности более сложные, чем шар. К тому же строить нерегулярные поверхности вручную в Автокаде непросто, это может подтвердить любой, кто пытался на Автокаде сделать чертеж, скажем, штампа для простенькой пластмассовой детали. Использование же Автолиспа не всегда возможно и удобно. Поэтому хочется предостеречь от эйфории, возникающей от чтения рекламных проспектов, и призвать ставить перед собой те задачи, решение которых на Автокаде значительно ускоряется и, следовательно, даст ощутимый эффект - это в основном задачи двумерного черчения.

В то же время есть ряд областей (например, архитектура, некоторые задачи трехмерного моделирования), в которых ставятся трехмерные задачи, решаемые средствами Автокада достаточно просто и изящно.

3.1.10. Форма

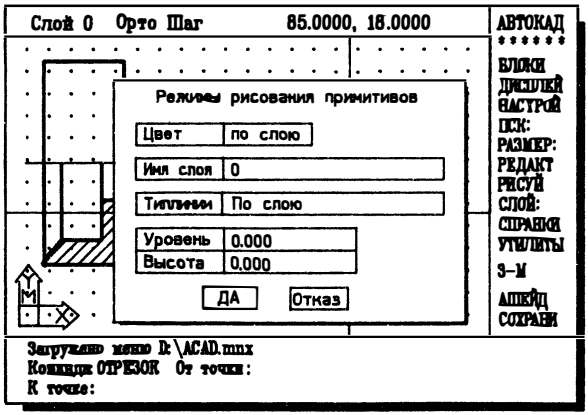
Формы представляют собой специальные графические примитивы, составленные из отрезков, дуг и окружностей. Формы хранятся не так, как чертежи. Формы можно рассматривать как специальный способ задания правил отрисовки графического объекта. Форма создается вручную вне Автокада в любом текстовом редакторе. По определенным правилам пользователь описывает процесс отрисовки простого объекта с тем, чтобы потом его легко можно было вставлять в разные части рисунка. Это описание хранится в специальном файле с расширением .shp, который должен быть обработан (скомпилирован) затем Автокадом (пункт 6 Главного Меню). В результате компиляции файла определений форм получается файл с тем же именем, что исходный, но с расширением .shx. Затем пользователь входит в Автокад и командой LOAD (ЗАГРУЗИ) загружает файл определений форм. После того (и только после того), как файл форм загружен в рисунок, можно осуществлять вставку формы в рисунок командой SHAPE (ФОРМА), которая запросит сначала имя формы, а потом ее начальную точку, высоту и угол поворота. Если вы забыли имена форм, то можно просмотреть список форм, введя в ответ на первый запрос команды ФОРМА знак вопроса.

Формы очень удобны для определения простых объектов, какими, например, являются литеры шрифтов, математические и картографические знаки и т.д. Вообще шрифты в Автокаде определяются при помощи форм - файл шрифтов представляет собой разновидность файла форм. Формы - специальное средство Автокада. Создание форм - непростое дело (мы сейчас не будем детально описывать этот процесс). В Автокаде существует гораздо более удобное и простое средство создания составных примитивов - блоки (см. п. 3.6).

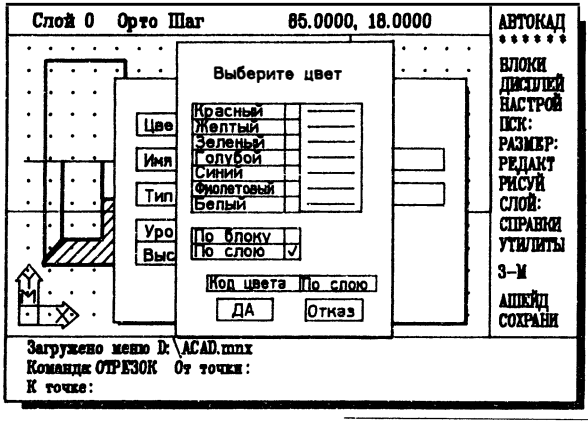
3.2. Общие свойства примитивов

Все примитивы обладают рядом общих свойств. Очевидными свойствами, присущими любому примитиву, являются цвет и тип линии. В то же время в Автокаде вводятся новые для традиционной графики свойства - принадлежность слою, уровень и высота. Все эти свойства подробно рассмотрены ниже.

При создании новых примитивов им назначаются текущие общие свойства - текущий цвет, текущий тип линии, текущие слой, уровень и высота. Текущие значения свойств могут быть назначены командами, управляющими этими свойствами: COLOR (ЦВЕТ), LINETYPE Set (ТИПЛИН Установи), LAYER Set (СЛОЙ Установи), ELEVATION (УРОВЕНЬ). Для установки текущих значений удобно пользоваться диалоговым окном (команда 'DDEMODES ('ДИАЛПРИМ), вызываемым из падающего меню Settings (Режимы), пункт Entity creation (Создание примитивов):



Новые значения уровня и высоты вводятся с клавиатуры, а цвет, тип линии и новый текущий слой выбираются указанием на кнопку против нужной строки списка, появляющегося во вложенном диалоговом окне. Поскольку многие команды считывают системные переменные сразу же после вызова (до того, как выдаются запросы), эффект новых установок, сделанных в прозрачном режиме, может не отразиться на выполнении текущей команды.



3.2.1. Цвет

Каждый примитив характеризуется цветом.

Текущий цвет (цвет, который назначается вновь создаваемому примитиву) может быть изменен командой COLOR (ЦВЕТ):

```
Command: COLOR
New color <1 (red)>: yellow
```

```
Команда: ЦВЕТ
Новый цвет <1 (красный)>: желтый
```

В ответ на этот запрос может быть введен как код цвета, так и его наименование (только для семи основных цветов - см. таблицу ниже). Возможны также ответы "bylayer" ("ПОСлою") и "byblock" ("ПОБлоку").

Команда COLOR (ЦВЕТ) не изменяет цвета уже отрисованных примитивов. Для изменения цвета уже созданных примитивов следует использовать команду CHANGE Color (ИЗМЕНИ Цвет).

Цвета в Автокаде могут быть постоянные и переменные. Понятия "постоянный цвет" и "переменный цвет" объясняются ниже.

Постоянный цвет

Постоянный цвет - это номер (код цвета), по которому Автокад определяет, каким цветом следует отобразить тот или иной примитив. Если у примитива постоянный цвет, например, синий, то этот примитив будет всегда отображаться синим цветом независимо от того, на каком слое или в каком блоке он находится.

Постоянные цвета в Автокаде кодируются целыми числами от 1 до 256 (для всех типов мониторов). Для того чтобы проверить, какие цвета возможны в вашей системе, выберите из экранного подменю SETTINGS (НАСТРОЙ) пункт COLOR (ЦВЕТ), из которого, в свою очередь, выберите пункт Chroma (256Цвета). При этом на вашем дисплее появится слайд возможных на вашем дисплее цветов.

Все цвета, которые не поддерживаются данным типом монитора, заменяются при отображении белым цветом. Следует отметить, что вне зависимости от типа монитора в базе данных рисунка в любом случае хранится код цвета примитива. Отсюда следует, что, например, при работе *с монохромным монитором* можно получать цветные чертежные документы, а при установке монитора с возможностью отображения 256 цветов ранее трехцветный рисунок может заиграть всеми цветами радуги.

Если вы работаете *с монитором EGA* (Enchanced Graphics Adapter), то ваш дисплей может отображать самое большее 16 цветов - 8 основных и 8 дополнительных. Из этих 16 цветов Автокад резервирует один цвет для служебных целей (цвет фона, код 0) и запрещает вводить его код, а все 15 остальных цветов доступны для рисования.

Хотя в диалоговом окне и в экранном меню доступны только 7 цветов, вы можете вводить и другие коды цветов в соответствии с таблицей, приводимой ниже (кстати, вы можете сами дополнить меню неосновными цветами). Обратите внимание на то, что некоторые коды соответствуют разным цветам в зависимости от цвета фона (графический редактор может быть настроен как рисующий черным по белому и белым по черному).

Коды цветов Автокада для белого (черного) цвета фона

Основные цвета		*Дополнительные цвета*	
нет	серый цвет (черный)	8	темно-серый
1	красный	9	темно-красный
2	желтый	10	коричневый
3	зеленый	11	темно-зеленый
4	голубой	12	темно-голубой
5	синий	13	темно-синий
6	фиолетовый	14	темно-фиолетовый
7	черный (белый)	15	белый (серый)

Переменный цвет

В Автокаде понятие цвета как свойства примитива расширено по сравнению с обыденным представлением - введены дополнительные "цвета" - цвет "bylayer" ("ПОСлою") и цвет "byblock" ("ПОБлоку"). Цвета по слою и по блоку присваиваются примитивам так же, как и постоянные цвета. Отличие заключается в том, что примитив, например, с цветом "bylayer" ("ПОСлою") не имеет "собственного" постоянного цвета, а принимает тот цвет, который назначен слою, на котором находится примитив. При изменении цвета слоя цвет такого примитива также изменяется. Переменный цвет "byblock" ("ПОБлоку") позволяет с каждой вставкой блока в чертеж связывать свой собственный цвет.

3.2.2. Тип линии

Все основные двумерные примитивы (отрезок, полилиния, дуга, круг) характеризуются типом линии.

Тип линии - это шаблон (последовательность чередущихся линейных сегментов, пробелов и точек), по которому отрисовываются линии Автокада. В отличие от цветов шаблоны типов линий различаются только по именам. Имя типа линии может состоять из букв, цифр и некоторых специальных знаков, таких, как "**$**" (знак денежной единицы), "**-**" (дефис), "**_**" (знак подчеркивания), в любых сочетаниях общей длиной не более 31 знака.

Шаблоны типов линий не "зашиты" в Автокад жестко, а определяются в специальных текстовых файлах (с расширением .lin), создаваемых в соответствии с установленными в Автокаде соглашениями. Стандартная библиотека типов линий

хранится в файле acad.lin. Вы можете модифицировать этот файл (изменяя стандартные типы линий Автокада) или создать в дополнение к стандартной свою собственную библиотеку типов линий, хранящуюся в отдельном файле. Число возможных типов линий (шаблонов) не ограничено.

Поскольку описание типа линии находится в специальной библиотеке, оно должно быть загружено в чертеж. Загрузка происходит либо автоматически (в процессе выполнения команд LAYER Ltype (СЛОЙ Типлин), LINETYPE Set (ТИПЛИН Установи) или из рисунка-прототипа), либо командой LINETYPE Load (ТИПЛИН Загрузи). Загружать шаблон нужно только один раз, в момент начала работы с новым типом линии. При загрузке определения типа линии шаблон отрисовки записывается в базу данных текущего чертежа и по окончании сеанса редактирования записывается в файл чертежа. В следующем сеансе редактирования, таким образом, загружать тип линии не нужно.

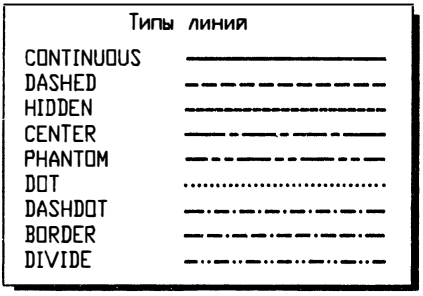
Если вы используете рисунок-прототип, то можете вообще исключить операцию загрузки типа линии. Дело в том, что при создании нового чертежа загруженные в прототип типы линий автоматически переносятся в новый файл чертежа. Загрузив один раз в прототип нужные вам типы линий, вы избавите себя от необходимости загружать их в новый чертеж вручную. В момент загрузки имя загружаемого типа линии заносится в *список определенных в рисунке типов линий*, который можно просмотреть, вызвав, например, команду 'DDEMODES ('ДИАЛПРИМ) и попытавшись изменить установленный тип линии. Для работы с типами линий (загрузки типа линий в чертеж, установки текущего типа линий, создания новых типов линий) предназначена команда LTYPE (ТИПЛИН). Изменение типа линии примитива осуществляется командой CHANGE LType (ИЗМЕНИ Типлин). Масштабом отрисовки шаблона типа линии управляет команда LTSCALE (ЛМАСШТАБ).

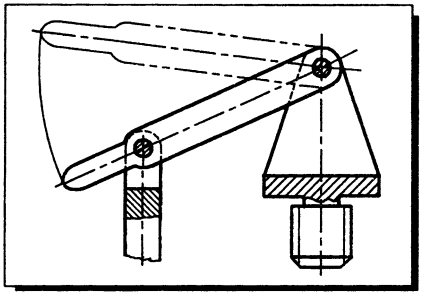
Постоянные и переменные типы линий

Понятие типа линии в Автокаде, аналогично цвету, расширено введением дополнительных "переменных" типов линий: "bylayer" ("ПОСлою") и "byblock" ("ПОБлоку") ("постоянным" типом линии будем называть любой тип линии из списка определенных в рисунке типов линий). Переменные типы линий присваиваются примитивам точно так же, как и постоянные. Как и для цветов, отличие заключается в том, что примитив, например, с типом линии "ПОСлою" не имеет "собственного" постоянного типа линии, а отображается тем типом линии слоя, на котором он находится. При изменении типа линии слоя тип линии такого примитива также изменяется. Переменный тип линии "ПОБлоку" позволяет с каждой вставкой блока в чертеж связывать свой собственный тип линии.

Загрузка типа линии из библиотеки

Загрузка типа линии осуществляется из файла, содержащего описание типов линий. Такой файл должен иметь расширение .lin. Стандартная библиотека типов линий Автокада хранится в файле acad.lin.





Для загрузки типа линии в чертеж нужно вызвать команду LINETYPE Load (ТИПЛИН Установи):

```
Command: LINETYPE
?/Create/Load/Set: L
Linetype(s) to load:
```

```
Команда: ТИПЛИН
?/Создай/Загрузи/Установи: З
Загрузить тип линии:
```

В ответ на этот запрос можно вводить имена типов линий, разделенные запятыми, причем допускается употребление символов поиска ДОС (т.н. "wildcards" - "*", "?"). После указания имени типа линии Автокад запрашивает имя файла-библиотеки типов линий; ответом по умолчанию является acad:

```
File to search <ACAD>:
```

```
Искать в файле <ACAD>:
```

Продемонстрируем использование символов "*" и "?" на примере загрузки всех типов линий, начинающихся с букв "DASH":

```
Command: LINETYPE
?/Create/Load/Set: L
Linetype(s) to load: dash*
File to search <ACAD>:
Linetype DASHED loaded.
Linetype DASHDOT loaded.
```

```
Команда: ТИПЛИН
?/Создай/Загрузи/Установи: З
Загрузить тип линии: dash*
Искать в файле <ACAD>:
Тип линии DASHED загружен.
Тип линии DASHDOT загружен.
```

Если тип линии уже загружен, то Автокад просит подтвердить повторную загрузку. При положительном ответе на вопрос (по умолчанию - подтверждение повторной загрузки) Автокад загружает указанный тип линии еще раз:

```
Linetype DASHED is already loaded. Reload it? <Y>:
Linetype DASHED reloaded.
```

```
Данный тип линии уже был загружен. Перезагрузить? <Д>:
Тип линии DASHED перезагружен.
```

Установка текущего типа линии

Для установки типа линии в качестве текущего (текущий тип линии присваивается всем вновь создаваемым примитивам) служит команда LINETYPE Set (ТИПЛИН Установи):

```
Command: LINETYPE
?/Create/Load/Set: S
New entity linetype (or ?) <текущий тип линии>:
```

```
Команда: ТИПЛИН
?/Создай/Загрузи/Установи: У
Новый тип линии (или ?) <текущий тип линии>:
```

В ответ на запрос можно указывать как постоянные типы линий (имена типов линий из библиотеки описаний acad.lin), так и переменные типы линий - "bylayer" ("ПОСлою") и "byblock" ("ПОБлоку"). Если устанавливаемый тип линии не загружен, то Автокад попытается загрузить его из стандартного файла описаний типов линий acad.lin. Если описание типа линии с указанным именем не будет найдено, Автокад выдаст сообщение:

```
Linetype ИМЯ not found. Use the LOAD option to load it.
```

```
Тип линии ИМЯ не найден. Используйте опцию ЗАГРУЗИ для загрузки.
```

Для просмотра библиотеки описаний типов линий введите знак вопроса. При этом Автокад попросит ввести имя файла библиотеки типов линий и выведет на экран список определенных в этом файле типов линий.

Создание нового типа линии

Одним из неоспоримых преимуществ Автокада является возможность создания своих типов линий. Несмотря на то что в стандартной поставке Автокада типы линий не соответствуют ЕСКД, соответствующие ГОСТу типы линий могут быть созданы за несколько минут.

Поскольку библиотека описаний типов линий (файлы с расширением .lin) - это обычные текстовые файлы, то библиотеку описаний типов линий можно дополнить и вне Автокада, воспользовавшись текстовым редактором. Отметим, что используемый редактор должен работать в режиме "программист", т.е. не добавлять в текстовый файл никаких своих служебных символов. Библиотека описаний типов линий может содержать неограниченное число определений типов линий; каждое определение состоит из двух строк - заглавной строки и собственно строки определения:

```
*имя[,описание]
выравнивание,штрих1,штрих2,штрих3,...
```

Приведем в качестве примера описание стандартного типа линии DASHDOT:

```
*DASHDOT,__ . __ . __ . __ . __ . __ .
A,0.5,-0.25,0,-0.25
```

В заглавной строке, как видим, помещается имя типа линии, по которому с этим типом будет работать Автокад, и его текстовое описание, которое будет выводиться на экран по команде LINETYPE ? (ТИПЛИН ?). Подчеркнем, что описание не определяет тип линии, а является комментарием и может быть, например, такой строкой: "Штрихпунктирный тип линии". Описание может быть опущено; в этом случае запятая после имени не ставится. Собственно, определением типа линии является вторая строка, которая использует следующие соглашения:

- Определение типа линии записывается в одну строку, в начале которой помещается буквенный код выравнивания, а затем идут разделенные запятыми действительные числа, задающие длину отрисовываемых штрихов
- Если число больше нуля, то отрисовывается штрих, длина которого равна заданному числу в условных единицах Автокада (режим "перо опущено")
- Если число меньше нуля, то делается пропуск, длина которого равна указанному числу в условных единицах Автокада (режим "перо поднято"). При выравнивании типа А первый штрих в определении шаблона не может быть пропуском
- Если число равно нулю, то отрисовывается точка

В настоящее время Автокад реализует только один тип выравнивания (способ отрисовки шаблона вблизи конечных точек отрезков, кругов и дуг), поэтому в начале строки всегда следует помещать латинскую букву "А". При выравнивании типа А Автокад следит за тем, чтобы конечными точками начала и конца линий были штрихи, а не пропуски. Если отрисовываемый отрезок меньше шаблона, то рисуется непрерывная линия.

При создании нового типа линии при помощи команды LINETYPE Create (ТИПЛИН Создай) Автокад сам следит за соблюдением формата файла описаний типа линий, запрашивая только те значения, которые допускается определять пользователю (выравнивание, например, всегда устанавливается типа А ав-

томатически). Ниже приведен пример создания типа линии, соответствующего штрихпунктирной линии (ГОСТ 2.303-68):

```
Command: LINETYPE
?/Create/Load/Set: C
Name of linetype to create: CENTER_GOST
File for storage of linetype <ACAD>:

Wait, checking if linetype already defined...

Descriptive text: ____ _ ____ _ ____ _ ____ _ ____ _ ____ _
Enter pattern (on next line):
  A,0.5,-0.1,0.1,-0.1

New definition written to file.
?/Create/Load/Set: CTRL C
Command:
```

```
Команда: ТИПЛИН
?/Создай/Загрузи/Установи: С
Введите имя типа линии, который будете создавать: ШТРИХПУНКТИР
Файл для хранения типа линии <ACAD>:

Ждите, идет проверка того, не определен ли уже данный тип...

Описывающий текст: ____ _ ____ _ ____ _ ____ _ ____ _ ____ _
Введите образец (на следующей строке):
  A,0.5,-0.1,0.1,-0.1

Новое определение записано в файл.
?/Создай/Загрузи/Установи: CTRL C
Команда:
```

Если в текущем файле описаний типов линий определение с таким именем уже имеется, то Автокад выводит его на дисплей и спрашивает, нужно ли заменить старое описание новым. При положительном ответе на этот вопрос новый тип линии определяется так же, как в примере выше.

3.2.3. Слой

Понятие слоя, незнакомое ранее конструкторам и чертежникам, является естественным для автоматизированного проектирования. Слои можно уподобить прозрачным калькам, из которых может состоять чертеж. На разных листах "кальки" удобно располагать объекты, которые можно объединить по какому-то одному признаку, например на одном листе - осевые линии, на другом - вспомогательные построения, на третьем - обводка. Как наложенные друг на друга кальки, слои совмещены в трехмерном пространстве Автокада, т.е. во всех слоях действительна одна и та же система координат и все связанные с ней параметры (например, лимиты).

Слой в Автокаде можно рассматривать просто как часть чертежа, на которой располагаются объекты. Каждый примитив принадлежит к какому-либо слою. По мере создания чертежа вы можете вводить новые слои (число слоев не ограничено), удалять (кроме нулевого слоя, свойства которого рассматриваются ниже), переименовывать и менять свойства существующих. В каждый момент времени новые объекты создаются на одном слое, который называется текущим.

В качестве примера использования слоев представим себе чертеж какого-либо устройства, изображения деталей которого содержатся на отдельных слоях. Для каждой детали, в свою очередь, также можно на разных слоях размещать различные варианты исполнения, объединенные по тем или иным признакам группы элементов, размеры отдельных деталей и сборочных единиц и т.п. Управляя видимостью слоев, можно формировать на экране дисплея и на бумажном носителе всю группу графических документов, начиная с проектной документации (например, чертеж общего вида) и кончая комплектом рабочей конструкторской документации (например, чертежи деталей) или документации технологической подготовки производства.

Команды работы со слоями

Ниже будут рассмотрены все команды, имеющие отношение к слоям. Основной командой является команда LAYER (СЛОЙ):

```
Command: LAYER
?/Make/Set/New/ON/OFF/Color/Ltype/Freeze/Thaw:
```

```
Команда: СЛОЙ
?/Создай/Установи/Новый/Вкл/Откл/Цвет/Типлинии/Заморозь/
Разморозь:
```

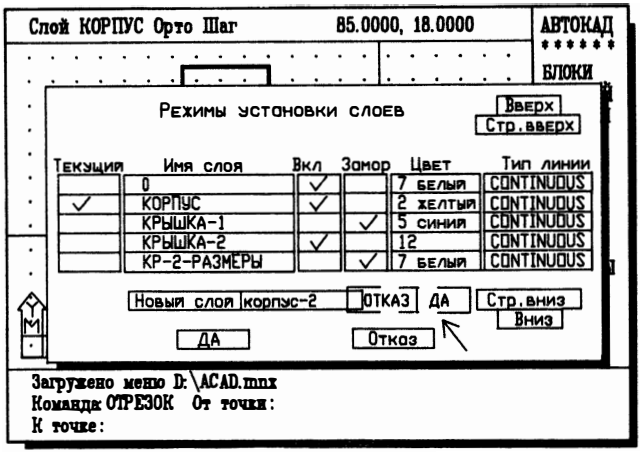
Ее опции позволяют управлять почти всеми свойствами слоев и будут рассматриваться постепенно по мере изложения. После выполнения выбранной опции команда не завершается, а вновь возвращается на основной запрос. Для завершения команды следует в ответ на основной запрос нажать на клавишу RETURN или ПРОБЕЛ (можно CTRL C; при этом завершенные подкоманды не отменяются). Если внесенные в чертеж изменения требуют регенерации изображения, то она будет произведена после завершения команды LAYER (СЛОЙ).

В этом пункте и далее мы для краткости не будем, если в этом нет необходимости, каждый раз приводить основной запрос команды, а будем сразу писать имя нужной опции, например LAYER Set (СЛОЙ Установи). Последовательный выбор команды и опции в дальнейшем будем называть подкомандой.

Для переименования слоя используется подкоманда RENAME Layer (НОВОЕИМЯ Слой).

Возможности этих двух команд объединяет в себе команда работы с диалоговым окном слоев, 'DDLAYER ('ДИАЛСЛОЙ), которую мы рекомендуем предпочитать основным командам. Диалоговое окно работы со слоями можно

вызвать из падающего меню Settings (Режимы), пункт Modify Layer... (Установки Слоев). Это диалоговое окно предоставляет пользователю удобную среду для просмотра списка определенных в рисунке слоев и их свойств, создания новых слоев, изменения имен и свойств существующих слоев:



Поскольку основные понятия и приемы работы с диалоговыми окнами рассмотрены в главе 2, мы не будем подробно описывать это диалоговое окно. Заметим только, что, хотя команду 'DDLAYER ('ДИАЛСЛОЙ) можно вызвать в прозрачном режиме, внесенные изменения не будут иметь видимого эффекта без регенерации изображения, если это необходимо (например, разморозка слоя).

Принадлежность объекта слою может быть изменена подкомандой CHANGE Layer (ИЗМЕНИ Слой).

Удалить слой можно только в том случае, если на слое нет ни одного объекта, и только в самом начале сеанса редактирования. Для удаления слоя следует пользоваться подкомандой PURGE Layer (УДАЛИ Слой).

В стандартной поставке Автокада есть также программа на Автолиспе (dellayer.lsp), которая реализует функцию удаления всех объектов, находящихся на указываемом слое. Работа с Автолиспом рассматривается в главе 8.

Просмотр списка слоев

Хотя самым удобным средством для просмотра списка слоев является диалоговое окно, список слоев можно также выводить на экран командой LAYER ? (СЛОЙ ?), при вызове которой на экран выводится запрос:

```
Layer name (s) for listing <*>:
```

```
Список имен слоев <*>:
```

Единственное преимущество этой подкоманды перед диалоговым окном заключается в том, что при указании имени слоя можно использовать символы поиска ("wildcards") ДОС - "*" и "?", использование которых уже рассматривалось нами (см. по предметному указателю).

Значением по умолчанию в этой подкоманде является символ "*", поэтому при нажатии в ответ на запрос клавиши RETURN или ПРОБЕЛ на экран выводится список всех определенных в рисунке слоев.

Создание слоя и выбор текущего слоя

При создании слоя вы должны присвоить ему *имя*, по которому будете отличать один слой от другого. Имя может состоять из букв, цифр и некоторых специальных знаков, таких, как "$" (знак денежной единицы), "-" (дефис) и "_" (знак подчеркивания), в любых сочетаниях общей длиной не более 31 знака. В качестве имени слоя можно использовать наименование изделия (например: НАСОС, РЕДУКТОР, КОРПУС), наименование изображения (например: ПЛАН, РАЗВЕРТКА, ФРОНТАЛЬНЫЙ), наименование акцентированной информации, содержащейся в слое (например: ШЕРОХОВАТОСТЬ, ТЕХНИЧЕСКИЕ ТРЕБОВАНИЯ, РАЗМЕРЫ) и т.п. Имена существующих в рисунке слоев хранятся в специальном списке, который удобно просматривать в диалоговом окне команды 'DDLMODES ('ДИАЛСЛОЙ), рассмотренной выше.

Для создания слоя пользуйтесь диалоговым окном, для чего следует открыть графу "Новый слой" и ввести имя нового слоя. При помощи "кнопки" Current (Текущий) диалогового окна можно также назначить нужный слой текущим (текущим может быть только один слой).

Создать новый слой можно также подкомандой LAYER Make (СЛОЙ Создай), выдающей запрос:

```
New current layer <значение по умолчанию>:
```

```
Новый текущий слой <значение по умолчанию>:
```

При создании нового слоя ему автоматически присваивается 7-й (белый) цвет и CONTINUOUS (непрерывный) тип линии. Слой создается включенным и автоматически становится текущим слоем чертежа. Если слой с таким именем существовал, то он делается текущим и включается. Если слой с указанным именем существовал, но был заморожен, то при попытке сделать его текущим Автокад выдаст предупреждающее сообщение, поскольку замороженный слой не может быть текущим.

Установить слой текущим можно также при помощи подкоманды LAYER Set (СЛОЙ Установи).

Если не нужно делать новый слой текущим, то для создания слоя следует воспользоваться подкомандой LAYER New (СЛОЙ Новый). В остальном эта подкоманда работает так же, как подкоманда LAYER Make (СЛОЙ Создай).

При создании слоев по понятным причинам не допускается использование символов поиска ДОС. Нажатие на клавишу RETURN или ПРОБЕЛ вместо ввода имени слоя воспринимается как отказ от подкоманды.

Цвет и тип линии слоя

С каждым слоем связан цвет и тип линии. Эти свойства слоя являются как бы значениями цветов и типов линий всех примитивов с переменными свойствами, находящихся на данном слое (о понятиях "переменный цвет" и "переменный тип линии" говорилось в предыдущих пунктах). Так, например, предположим, что слою TEST назначен красный цвет. Тогда красным цветом будут отрисовываться те примитивы, которые имеют переменный цвет "bylayer" ("ПОСлою"). Примитивы с постоянными цветами (голубым, белым и т.п.) будут отрисовываться своими цветами (все сказанное в равной степени относится и к типам линий примитивов).

Для переназначения типа линии и цвета существующего слоя используются соответственно подкоманды LAYER Ltype (СЛОЙ Типлинии) LAYER Color (СЛОЙ Цвет). При вызове этих подкоманд сначала Автокад запрашивает имя типа линии (код или название цвета), а затем имя слоя. Имя слоя можно указывать с применением символов поиска ДОС. Значением по умолчанию в этих подкомандах является имя текущего слоя. Подкоманда LAYER Ltype (СЛОЙ Типлинии) выдает запрос:

```
Line type (or ?) <текущий тип линии>:
```

```
Типлинии (или ?) <текущий тип линии>:
```

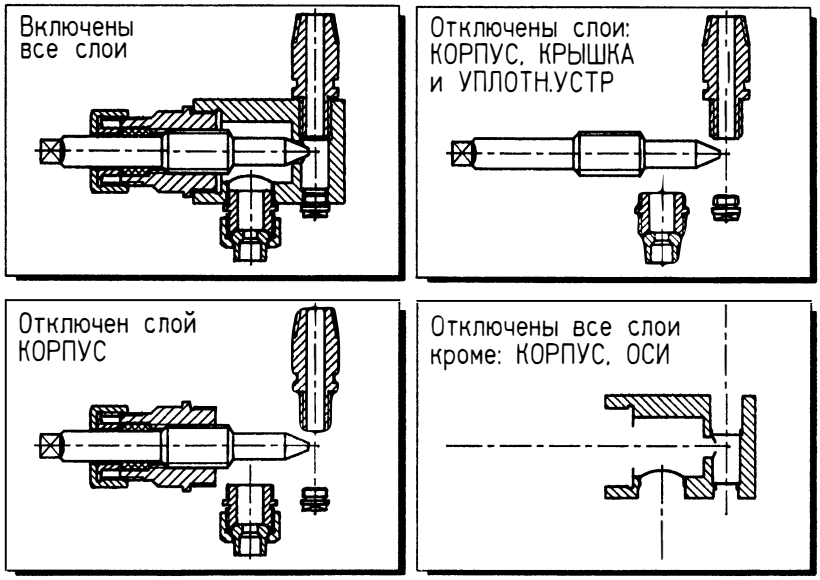
В ответ вы можете ввести знак вопроса, при этом на экран будет выведен список загруженных в чертеж типов линий.

Выключение и заморозка

Каждый слой может быть включен (On) или отключен (Off). Объекты, находящиеся на включенном слое, видимы; отключение слоя делает невидимыми все находящиеся на нем объекты.

Включение и отключение слоев - удобное средство создания и редактирования чертежа. Если, например, в ходе создания чертежа требуется проводить сложные геометрические построения для одной из деталей, то можно создать временный слой, на котором будут располагаться все вспомогательные линии (модульная сетка, трафареты, шаблоны и т.п.), и отключить слои, содержащие другие детали. По окончании работы временный слой можно отключить. Ниже

приводится пример того, как на разных слоях можно располагать отдельные детали, принадлежащие одной сборочной единице.



Слой также может быть замороженный и размороженный (Frozen и Thawed). Объекты, находящиеся на замороженном слое, так же не отображаются на экране (невидимы), как и на отключенном. Для того чтобы понять разницу между отключенным и замороженным слоем, нужно сначала разобраться в том, как Автокад строит изображение на дисплее.

В графической базе данных Автокада информация о чертеже хранится в виде чисел с плавающей точкой, что позволяет поддерживать точность представления данных до 16 знаков после запятой, а соотношение между размерами наименьшего и наибольшего объектов рисунка может достигать десяти триллионов к одному. Однако математические операции с действительными числами требуют большого объема вычислений, и даже при наличии сопроцессора время обработки изображений возрастает до неприемлемых величин. Вместе с тем экран графического монитора отображает сравнительно небольшое количество точек, и с координатами графических объектов на экране можно работать как с целыми числами. Поэтому при построении изображения Автокад осуществляет преобразование типов данных из формата с плавающей точкой в целые числа. Для того чтобы не выполнять такое преобразование каждый раз при изменении вида, в Автокаде используется *виртуальный экран*. Виртуальный экран - это область памяти, в которой хранится оригинал изображения, выводимого на монитор. Виртуальный экран имеет размеры 32768 x 32768 пикселов; координаты объек-

тов хранятся на виртуальном экране как целые числа. При построении изображения сначала осуществляется построение изображения на виртуальном экране (этот процесс называется *регенерацией*), а затем оно переносится с виртуального экрана на дисплей (происходит *перерисовка* изображения). Команды управления изображением следят за степенью соответствия изображения на виртуальном экране истинному изображению и по возможности перерисовывают вид, т.е. без регенерации переносят изображение с виртуального экрана на дисплей.

Если слой отключен, то изображение с виртуального экрана не переносится на монитор - объекты, находящиеся на отключенном слое, становятся невидимыми. Однако объекты, находящиеся на отключенном слое, регенерируются, и работа с изображением из-за этого замедляется. Замораживание слоя накладывает запрет на регенерацию объектов, находящихся на этом слое, в результате чего Автокад работает быстрее. Замораживание слоя требуется только при работе с очень сложными чертежами. В большинстве случаев следует просто отключать слой.

Для включения и отключения, заморозки и разморозки слоев пользуйтесь либо диалоговым окном, либо соответственно подкомандами LAYER OFF (СЛОЙ Откл), LAYER ON (СЛОЙ Вкл), LAYER Freeze (СЛОЙ Заморозь), Layer Thaw (СЛОЙ Разморозь). Во всех этих подкомандах разрешается при указании имен слоев использовать символы поиска ДОС.

Свойства нулевого слоя

В любом чертеже Автокада всегда существует по крайней мере один слой с именем "0". Этот слой имеет специальные свойства. Он автоматически формируется при создании рисунка, и ему всегда присваивается 7-й (белый) цвет и CONTINUOUS (непрерывный) тип линии.

Слой 0 предназначен для работы с блоками. Приведем правило:

- Если входящий в блок примитив был создан на слое 0, то при вставке блока он помещается на текущий слой.

Поскольку мы уже знакомы с понятиями переменного цвета и типа линии, то можем сформулировать следующее утверждение, вытекающее из предыдущего:

- Если входящий в блок примитив был создан на слое 0 и имел цвет и(или) тип линии "bylayer" ("ПОСлою"), то при вставке блока он будет отрисован цветом и типом линии текущего слоя.

Во всех других случаях при вставке блока цвет и тип линии входящих в блок примитивов не будут зависеть от свойств текущего слоя.

Слой 0 не может быть удален или переименован.

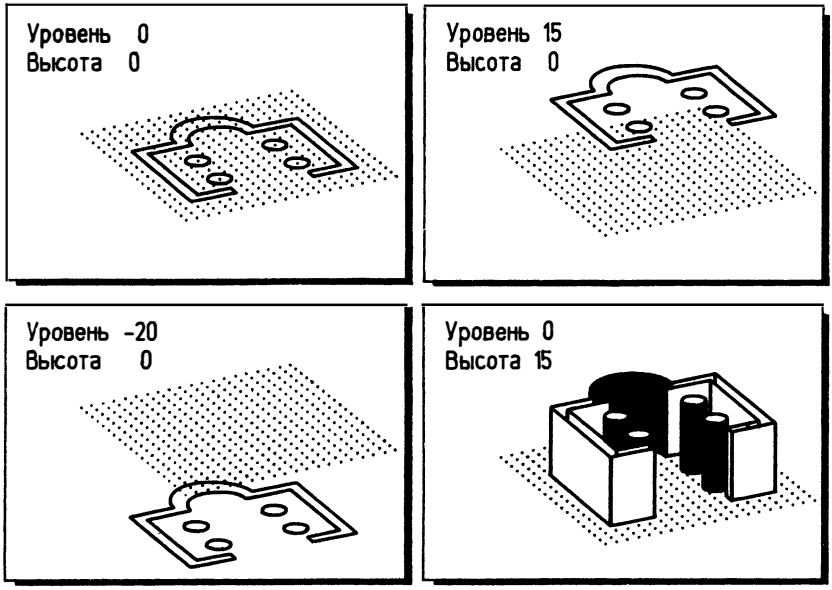
3.2.4. Уровень и высота

Создаваемые в текущей ПСК объекты обычно имеют координату Z, равную нулю. Однако в Автокаде есть средство назначения по умолчанию координаты Z.

Дело в том, что при создании объекта координата Z по умолчанию в действительности берется равной значению переменной, называемой *уровнем*. Уровень можно рассматривать как дополнительное средство задания плоскости отрисовки двумерных объектов - эта плоскость отстоит от плоскости X0Y текущей системы координат на расстояние (Z), равное значению уровня.

Плоские примитивы могут иметь *высоту*. Высоту можно рассматривать как толщину примитива по координате Z. Если высота, например отрезка, равняется нулю, то Автокад отрисовывает обычный двумерный чертеж. При ненулевой высоте точка будет отрисовываться как перпендикулярный плоскости X0Y отрезок с длиной, равной значению высоты, отрезок будет отрисовываться как поставленная на ребро грань, круг - как цилиндр, полилиния с шириной - брусками и т.д. Таким образом, задавая высоту примитивов, мы можем отрисовывать трехмерные объекты, причем Автокад будет правильно удалять скрытые линии.

Ниже приведены примеры использования этих свойств примитивов.



Уровень и высоту вновь создаваемых примитивов можно задавать в диалоговом окне 'DDEMODES ('ДИАЛПРИМ), которое приведено на стр. 66. Для установки текущего уровня и высоты можно также использовать команду ELEV (УРОВЕНЬ), формат которой настолько прост, что мы его не приводим. Изменять уровень и высоту уже созданных примитивов можно с помощью подкоманд CHANGE Elev (ИЗМЕНИ Уровень) и CHANGE Thickness (ИЗМЕНИ Высота).

Имеется существенная разница между созданием "трехмерных" объектов при

помощи высоты и применением действительно трехмерного примитива - трехмерной грани. За недостатком места этот вопрос не может быть рассмотрен более подробно.

3.3. Штриховка

При создании чертежей часто приходится выполнять разрезы и сечения, в которых используется графическое обозначение материала деталей - штриховка.

Штриховка в Автокаде выполняется автоматически. Для выполнения штриховки пользователь должен указать Автокаду *контур штрихования*. Контур штрихования заполняется линиями штриховки автоматически с использованием *образцов штриховок* (стандартные образцы штриховок находятся в файле acad.pat; существует возможность определения своих собственных образцов) или простыми линиями, определив непосредственно в процессе штрихования некоторые параметры (угол наклона, расстояние между линиями и т.п.).

Для выполнения штриховки предназначена команда HATCH (ШТРИХ). При штриховании с помощью образца (выбор по умолчанию) надо ввести имя образца штриховки. После указания имени (Автокад сразу ищет его в файле acad.pat или в файле имя.pat), вам надо определить масштаб и угол поворота шаблона штриховки, затем следует указать объекты, определяющие контур штрихования:

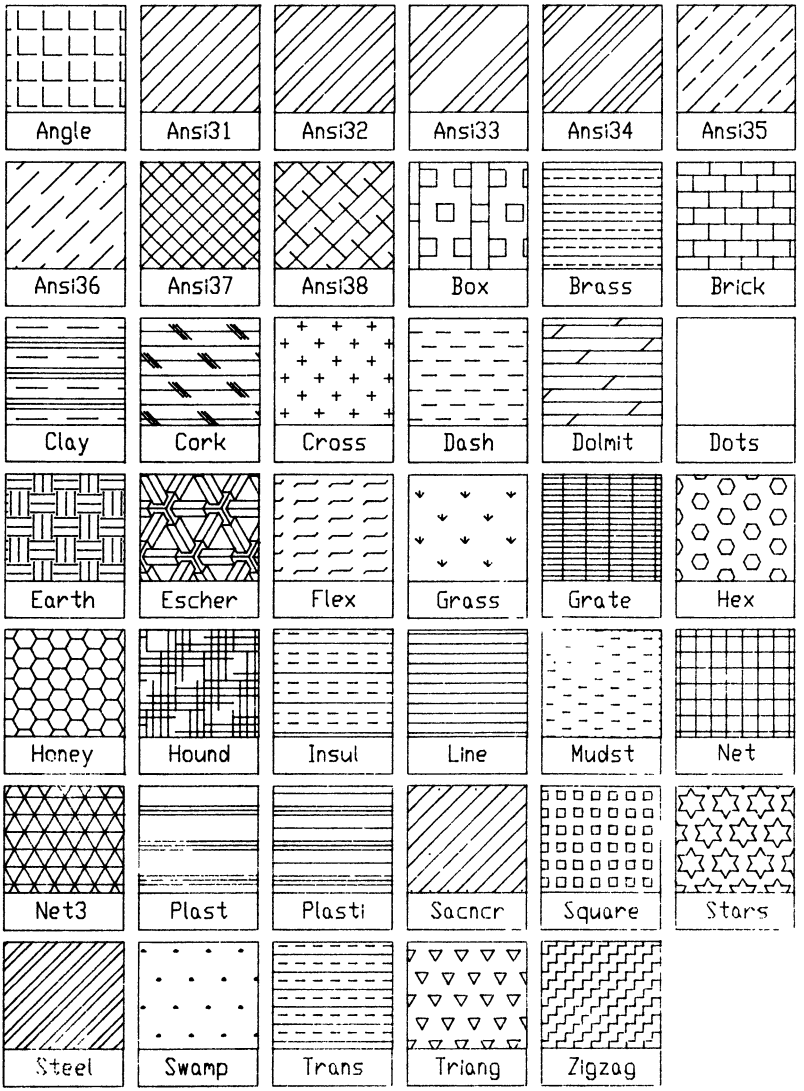
```
Command: HATCH
Pattern (? or name/U, style) <по умолчанию>: ansi31
Scale for pattern <1.000>:
Angle for pattern <0>:
Select objects:
```

```
Команда: ШТРИХ
Образец (? или имя/С, стиль) <по умолчанию>: ansi31
Масштаб штриховки <1.000>:
Наклон штриховки <0>:
Выберите объекты:
```

Если вы не помните имена шаблонов штриховок Автокада, то можете вывести на экран имена всех определенных в стандартной библиотеке шаблонов штриховок. Для этого следует вместо имени образца ввести знак вопроса. Если вы не хотите пользоваться шаблонами штриховок, то можете определить штриховку прямо в процессе штрихования. Для этого следует вместо имени шаблона штриховки ввести букву "U" ("С"), после чего Автокад предложит определить простые параметры штрихования:

```
Command: HATCH
Pattern (? or name/U, style) <по умолчанию>: U
Angle for crosshatch lines <0>: 45
Spacing between lines <1.0000>: 3
Double hatch area? <N>:
Select objects:
```

Штриховые узоры Автокада



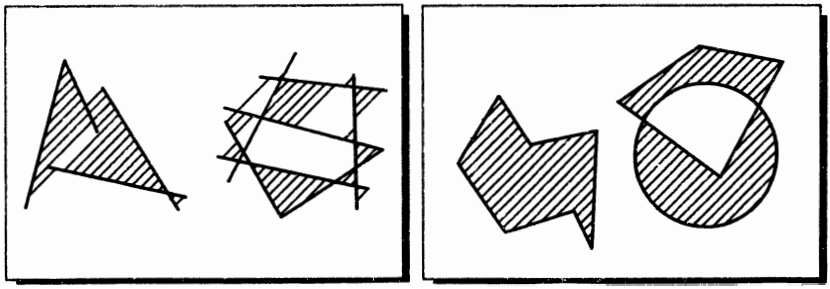
```
Команда: ШТРИХ
Образец (? или имя/С, стиль) <по умолчанию>: С
Угол наклона штриховки <0>: 45
Расстояние между штриховыми линиями <1.0000>: 3
Штриховать крест-накрест? <Н>:
Выберите объекты:
```

Выбор определяющих контур объектов производится так же, как в других командах: можно выбирать объекты рамкой, удалять объекты из набора, пополнять набор. После завершения выбора объектов Автокад выполняет штриховку.

К контуру штрихования в Автокаде предъявляются особые требования, без выполнения которых правильное штрихование не гарантируется. Существует также несколько ограничений, связанных со свойствами штриховки как объекта Автокада. Во-первых, Автокад не анализирует пересекающиеся примитивы на предмет того, образуют ли они замкнутую область, поэтому

- контур штрихования должен быть замкнут и состоять из примитивов (отрезков, дуг, окружностей, полилиний или граней), имеющих *общие конечные точки*.

Ниже приведены примеры выполнения штриховки при неправильном (слева) и правильном указании контуров.



При построении чертежа лучше всего заранее учитывать эту особенность штриховки и отрисовывать будущие контуры штрихования одной командой LINE (ОТРЕЗОК) или PLINE (ПЛИНИЯ) с использованием опции Close (Замкни).

При штриховке контура, ограниченного полилинией ненулевой ширины или полосой, контур штриховки определяется осевой линией этих примитивов.

Штриховка распознает внутреннюю структуру блока - штрихование производится так, как если бы мы указали на отдельные объекты рамкой.

Штриховка - это обычные отрезки Автокада, отрисованные в соответствии с шаблоном и объединенные во внутренний блок. Отсюда следует несколько свойств штриховки:

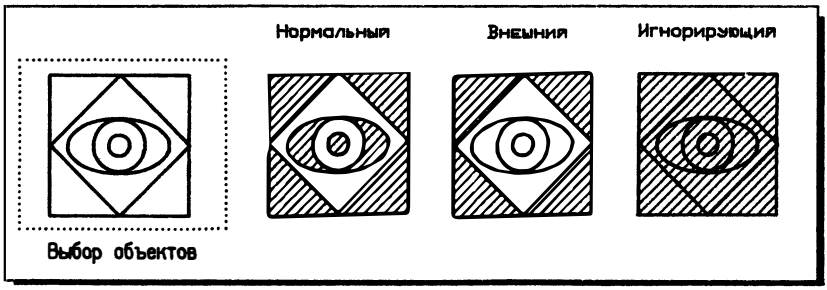
- Штриховка - это один объект, а не много отрезков. Поэтому для удаления штриховки достаточно указать на любую из ее линий. В то же время это

иногда затрудняет редактирование, поскольку некоторые из команд редактирования не работают с блоками (например, STRETCH (РАСТЯНИ)

- При отрисовке штриховки действительны все текущие общие свойства примитивов (тип линии, цвет, уровень, высота). Поэтому перед выполнением штриховки рекомендуется устанавливать CONTINUOUS (непрерывный) тип линии
- Штриховка всегда создается в текущей ПСК, даже если указанный контур располагается вне плоскости ПСК

3.3.1. Стили штрихования

При выполнении штриховки можно выбирать не только ее образец, но и стиль. В Автокаде предусмотрены три стиля штриховок: Normal (Нормальный), Outer (Внешний) и Ignoring (Игнорирующий). Рассмотрим подробнее, что означает каждый из указанных стилей штриховки (считаем, что объекты выбираются рамкой). Сначала следует отметить, что стиль действует только в том случае, когда внутри контура штрихования содержатся какие-то объекты. Если пространство внутри области штриховки пусто, то оно заполняется выбранным образцом. Ниже приведен пример использования стилей штрихования:



Нормальный стиль штриховки устанавливается по умолчанию. Штрихование начинается с внешнего контура, и при нахождении внутреннего контура штрихование прекращается до тех пор, пока не будет найден еще один вложенный контур. Таким образом, все нечетные области будут заштрихованы, а четные нет. *Внешний* стиль штриховки штрихует только внешний контур. При нахождении вложенного контура штрихование прекращается (ни один вложенный контур не будет заштрихован). *Игнорирующий* стиль отключает режим поиска и анализа вложенных контуров штриховки. Заштриховывается вся внутренняя область.

Для использования стиля, отличного от задаваемого по умолчанию, надо после имени стиля или опции через запятую ввести первую букву названия стиля.

3.3.2. Создание новых образцов штриховок

Хотя описание шаблона штриховок сходно с описанием типа линии, создание новых образцов штриховок более сложный процесс. Шаблоны штриховок хра-

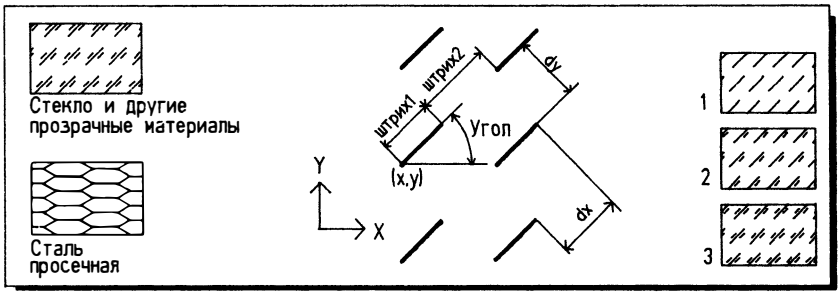
нятся в файлах с расширением .pat (стандартная библиотека описаний шаблонов штриховок хранится в файле acad.pat). Новый образец штриховки может быть записан как в файл acad.pat, так и в отдельный файл. Если при вызове команды HATCH (ШТРИХ) мы укажем имя нового образца штриховки (скажем, NEW), Автокад сначала будет искать образец с таким именем в файле acad.pat, а затем, если его там нет, в файле new.pat.

Описание образца штриховки состоит из заглавной строки и одной или нескольких строк определения:

```
*имя[,описание]
угол,X,Y,DX,DY,штрих1,штрих2,...
...
угол,X,Y,DX,DY,штрих1,штрих2,...
```

Здесь, как и в файлах описания типов линий, строка **описание** - просто текстовый литерал, выводимый на экран подкомандой HATCH ? (ШТРИХ ?).

Любая штриховка Автокада состоит из одной или нескольких штриховых линий, каждая из которых определяется на отдельной строке. Параметры **штрих1** и **штрих2** играют ту же роль, что и в файлах описаний типов линий (см. выше): задают шаблон типа линии (правда, в отличие от типа линии этот шаблон отрисовывается отдельными отрезками). Первые пять параметров определяют расположение линий штриховки на плоскости и показаны на рисунке. Ниже приводится описание шаблона штриховки "стекло и другие прозрачные материалы" (ГОСТ 2.306-68):




```
*glass
45, 0,0, 1,0.7, 1,-0.7
45, 0.48,0.28, 1,0.7, 0.3,-1.4
45, 0.07,0.28, 1,0.7, 0.3,-1.4
```

Попробуйте поэтапно создать штриховку "стекло и другие прозрачные материалы". Создайте файл glass.pat и введите в описание шаблона штриховки главную линию (первую строку). Попробуйте заштриховать полученным образцом какую-нибудь область, а затем постепенно дополняйте шаблон (см. кадры 1, 2, 3 на рисунке выше).

После отладки шаблона попробуйте выполнить такой чертеж:

Приведем еще один пример создания нового образца штриховки - "сталь просечная" (ГОСТ 2.306-68):

```
*holesteel
0, 0,0, 4.5,0.86603, 3.0,-6.0
30, -1.5,0.86603, 0.86603,1.5, 1.73205,-3.4641
-30, -1.5,0.86603, -0.86603,1.5, 1.73205,-3.4641
```

3.4. Текст

Текст - это примитив Автокада, имеющий такие специальные свойства, как точку вставки текстовой строки, гарнитуру, масштабные коэффициенты отображения и, конечно, значение текстовой строки (собственно текст). Текст в Автокаде может иметь любые размеры. Его можно растягивать, сжимать, поворачивать, наклонять буквы, зеркально отражать, подчеркивать, надчеркивать и пр.

Для того чтобы разобраться с тем, как в Автокаде осуществляется отрисовка текста, введем понятия *шрифт* и *гарнитура.*

Шрифт - это модель отрисовки символов. В Автокаде используются графические векторные шрифты, т.е. любая буква отрисовывается последовательными векторами, относительные длины и углы наклона которых заданы в файле определения шрифта. Этот файл представляет собой не что иное, как файл форм Автокада, что позволяет в принципе создавать свои собственные шрифты (этот процесс настолько громоздок, что мы его здесь рассматривать не будем).

Для того чтобы не указывать каждый раз при вставке текста редко меняющиеся параметры (например, степень сжатия/растяжения, угол наклона букв и т.п.), введено понятие гарнитуры.

Гарнитура - это характеристика отрисовки шрифта. Гарнитур может быть много; каждая гарнитура имеет имя. Гарнитура включает в себя имя файла

Шрифты Автокада

Имя шрифта	Примеры начертания	
	Русская клавиатура	Латинская клавиатура
Cyrillc	Автокад 10	AutoCAD 10
Greekc	±?????? ø	АυτοΚΑΔ 10
Greeks	±?????? ø	АυτοΚΑΔ 10
Gothice	±?????? ø	𝔄𝔲𝔱𝔬𝔈𝔄𝔇 10
Gothicg	±?????? ø	𝔄𝔲𝔱𝔬𝔈𝔄𝔇 10
Gothici	±?????? ø	𝔄𝔲𝔱𝔬𝔈𝔄𝔇 10
Italicc	±????? ø	*AutoCAD 10*
Italict	±????? ø	***AutoCAD 10***
Monotxt	Автокад 10	AutoCAD 10
Romanc	±?????? ø	AutoCAD 10
Romand	Автокад 10	AutoCAD 10
Romans	Автокад 10	AutoCAD 10
Romant	±?????? ø	**AutoCAD 10**
Russ	Автокад 10	AutoCAD 10
Scriptc	±?????? ø	*AutoCAD 10*
Txt	Автокад 10	AutoCAD 10

шрифта, высоту текста, степень сжатия/растяжения, угол наклона, направление отрисовки текста (слева направо или справа налево), ориентацию текста (вертикальный или горизонтальный) и другие параметры.

С примитивом Автокада "текст" связан не файл шрифта, а гарнитура. Введение понятия гарнитуры позволяет, таким образом, создавать несколько видов одного и того же шрифта. Изменяя в определении гарнитуры имя файла шрифта, можно менять шрифт уже выполненных надписей.

Для выполнения надписей на чертежах в Автокаде существуют команды TEXT (ТЕКСТ), DTEXT (ДТЕКСТ). Для работы с гарнитурами и шрифтами - команды STYLE (СТИЛЬ) и RENAME Style (НОВОЕИМЯ Гарнитура).

При использовании команды TEXT (ТЕКСТ), после того, как вы определите размер букв (если в гарнитуре указана не фиксированная высота текста, а введено значение 0) и угол поворота строки, Автокад ждет ввода текстовой строки. Текст появляется на экране только после завершения команды.

Единственная разница между командами TEXT (ТЕКСТ) и DTEXT (ДТЕКСТ) заключается в том, что команда DTEXT (ДТЕКСТ) отображает текст на экране по мере его ввода и позволяет вводить несколько строк текста за один раз, причем точку вставки новых строк можно прямо в процессе выполнения надписи указывать "мышью" на экране.

Если вместо указания начальной точки нажать на клавишу RETURN или ПРОБЕЛ, то начало нового текста будет располагаться на строке ниже введенного предыдущего текста. При этом высота и угол поворота берутся равными их предыдущему значению, вне зависимости от того, выполнялись ли между вызовами команды DTEXT (ДТЕКСТ) другие команды.

```
Command: DTEXT
Start point or Align/Center/Fit/Middle/Right/
Style:
```

```
Команда: ДТЕКСТ
Начальная точка или ВПИсанный/Центр/ВЫравненный/Середина/ВПРаво/
Гарнитура:
```

В ответ на запрос необходимо выбрать один из возможных типов размещения текста или выбрать текущую гарнитуру.

3.4.1. Способы размещения текста

Start point (Начальная точка)

Является опцией по умолчанию и определяет точку вставки строки. После указания точки вставки надо ввести высоту букв и угол поворота строки

Aligned (ВПИсанный)

Эта опция требует указания двух точек, определяющих длину всех вводимых строк. Размер текста определяется числом символов на строке; буквы масштабируются без изменения пропорций так, чтобы заполнить всю строку

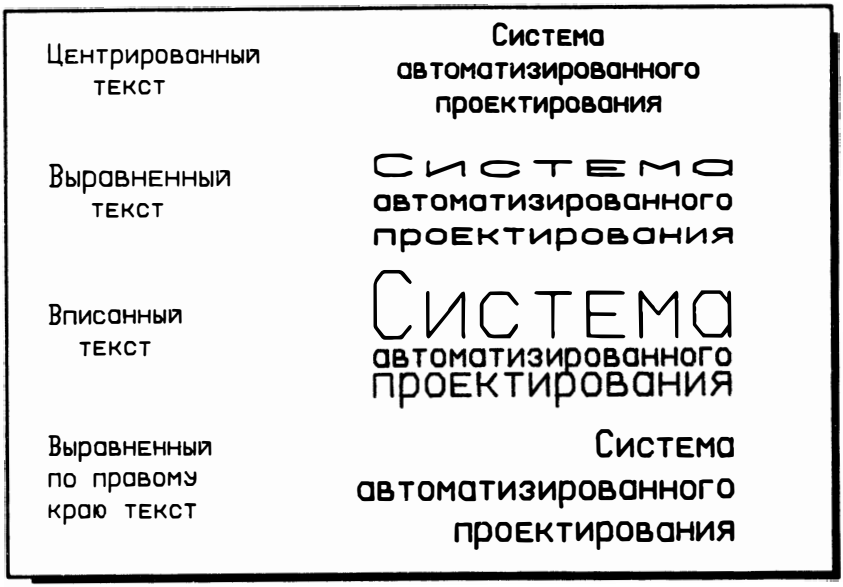
Center (Центр)
Задается точка, относительно которой текстовая строка центрируется по горизонтали

Fit (ВЫравненный)
Текст вписывается в строку заданной длины и выравнивается по высоте. Пропорции букв при этом меняются

Middle (Середина)
Задается точка, относительно которой текстовая строка центрируется как по горизонтали, так и по вертикали

Right (ВПРаво)
Все строки текста выравниваются по правому краю, которым становится указанная точка

Style (Гарнитура)
Эта опция позволяет выбрать текущую гарнитуру, после чего возвращается к основному запросу команды



3.4.2. Ввод специальных символов

На чертежах часто используются специальные символы, которые удобно вводить вместе с текстом (знаки диаметра, градуса, плюс/минус и т.п.). В файлах определения шрифтов (правда, не во всех) кроме букв также помещены

определения таких специальных символов. Однако если букву мы "вызываем" нажатием на соответствующую клавишу, то для того, чтобы поместить в строке специальный символ, используются т.н. управляющие коды. Так, например, если мы в текстовую строку включим последовательность "%%c", то Автокад отрисует после регенерации изображения знак диаметра (если, конечно, он определен в текущем шрифте). Кроме вставки специальных символов при помощи управляющих кодов можно также подчеркнуть или надчеркнуть текст. Ниже приведены допустимые управляющие последовательности:

%%o - Переключение режима надчеркивания (Вкл/Откл)
%%u - Переключение режима подчеркивания (Вкл/Откл)
%%d - Специальный символ "градус" (°)
%%p - Специальный символ "допуск" (±)
%%c - Специальный символ "диаметр" (Ø)
%%% - Вывод одиночного символа "процент"
%%nnn - Специальный символ с кодом ASCII "nnn"

3.4.3. Определение гарнитуры

Гарнитуру можно создать или изменить командой STYLE (СТИЛЬ), которая связывает с именем гарнитуры некоторые постоянные для разных текстовых строк параметры использования шрифта. Еще раз напомним, что для одного шрифта может быть определено несколько гарнитур с разными именами и созданной гарнитуре можно поставить в соответствие любой имеющийся шрифт. Вы можете, например, для одного и того же шрифта определить две гарнитуры, скажем, с именами ВЫСОКАЯ и ШИРОКАЯ, что позволит в дальнейшем при выполнении надписей на чертеже быстро менять настройки параметров шрифта:

```
Command: STYLE
Text style name (or ?) <txt>: HIGH
New style.
Font file <txt>: romans
Height <0.000>:
Width factor <1.00>: 0.5
Obliquing angle <0>:
Backwards? <N>:
Upside-down? <N>:
HIGH is now the current text style.
```

```
Команда: СТИЛЬ
Имя гарнитуры шрифта (или ?) <СТАНДАРТ>: ВЫСОКАЯ
Новая гарнитура.
Файл шрифта <TXT>: romans
Высота <0.00>:
Степень сжатия/растяжения <1.00>: 0.5
Угол наклона <0.00>:
Справа налево? <Н>:
Перевернутый? <Н>:
Вертикальный? <N>:
ВЫСОКАЯ - Новая гарнитура шрифта
```

Аналогично определяется гарнитура с именем ШИРОКАЯ. После этого, пользуясь опцией Style (Гарнитура) команд TEXT (ТЕКСТ) и DTEXT (ДТЕКСТ), можно быстро менять одну гарнитуру на другую.

Для изменения имени существующей гарнитуры используйте подкоманду RENAME Style (НОВОЕИМЯ Гарнитура). После назначения гарнитуре нового имени все вхождения этой гарнитуры в существующие текстовые строки будут переопределены.

3.4.4. Контурный текст

По мере насыщения чертежа на перерисовку и регенерацию изображения уходит все больше и больше времени. Чтобы избежать этого, можно воспользоваться одним из следующих способов:

- До последнего момента используйте простейший шрифт, TXT. По окончании работы над чертежом вы можете, переопределив гарнитуру, быстро заменить шрифт. Таким образом вы экономите время при редактировании чертежа. При замене шрифта остальные параметры гарнитуры следует оставить без изменений. Если вы измените высоту или угол наклона букв, может оказаться, что текст с новым шрифтом не вписывается в отведенное для него место. Следует иметь в виду, что шрифты отличаются по ширине, так что в некоторых случаях вам придется изменить ширину букв.
- Пользуйтесь командой QTEXT (КТЕКСТ), которая подавляет отрисовку букв, показывая лишь габаритные очертания текстовой строки. Эта команда находится в экранном меню SETTINGS (НАСТРОЙ). При очередной регенерации текст заменяется прямоугольником. Перед выводом на плоттер не забудьте, еще раз воспользовавшись командой QTEXT (КТЕКСТ), выключить контурный текст, иначе плоттер нарисует вам вместо текста прямоугольники.

3.5. Размеры

Проставление размеров - один из самых трудоемких этапов оформления чертежа. Этот процесс в Автокаде в значительной степени автоматизирован введением специального примитива - размера.

Несмотря на то что размер состоит из многих элементов (выносные линии, размерная линия, текст, линия выноски), для конструктора он является единым целым, поскольку все элементы, из которых состоит размер, тесно связаны между собой (размерный текст, например, зависит от расстояния между размерными линиями и текущих единиц измерения, положение его определяется расположением размерных линий и т.п.). Эта идеология реализуется с помощью специального примитива Автокада - размера. При создании размера все входящие в него примитивы записываются во внутренний блок (объединяются в один составной примитив), что придает размеру как примитиву новые свойства. Поскольку

размер в принципе можно проставить без использования специальных средств образмеривания Автокада (нарисовать командой LINE (ОТРЕЗОК) выносные и размерные линии, командой TEXT (ТЕКСТ) написать размерный текст и т.п.), то размеры, создаваемые специальными командами Автокада, принято называть *ассоциативными размерами.* Здесь и далее под словом "размер" мы будем понимать именно ассоциативный размер как примитив Автокада.

Перечислим основные свойства размера:

- Размер является составным примитивом (непоименованным блоком специального вида), поэтому команды редактирования (за исключением STRETCH (РАСТЯНИ) работают с размером как с единым целым
- Размерный текст может включать в себя, кроме значения размера, текст, введенный пользователем при проставлении размера. Если при вводе размерного текста вы не подавляете измеренное Автокадом значение, то оно остается связанным с базой размера (не фиксировано, а измеряется в процессе регенерации). Поэтому при изменении базы такого размера (например, в процессе редактирования чертежа) значение размера автоматически корректируется
- Поскольку размер является блоком, он может быть расчленен на составные примитивы. При этом размер теряет все свои характерные свойства и с точки зрения Автокада перестает быть размером
- Генерация ассоциативных размеров подавляется отключением системной переменной DIMASO (РЗМАССО)

Для проставления размеров в Автокаде предусмотрены две команды - DIM (РАЗМЕР) и DIM1 (РАЗМЕР1), которые находятся в экранном меню DIM: (РАЗМЕР:). При выборе этого пункта вызывается команда DIM (РАЗМЕР), которая переводит систему в специальный режим образмеривания:

```
Command: DIM
Dim:
```

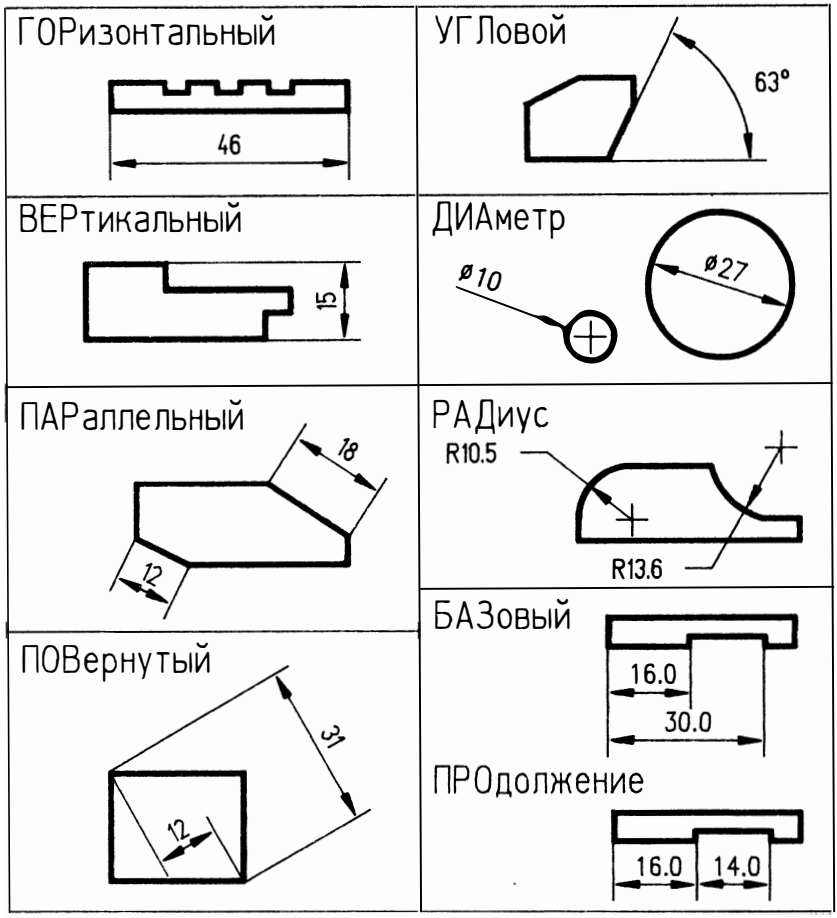
```
Команда: РАЗМЕР
Размер:
```

В этом режиме набор команд Автокада заменяется на набор подкоманд образмеривания и команды Автокада становятся недоступными (поэтому в набор команд образмеривания включена команда REDRAW (ОСВЕЖИ). Выход из команды DIM (РАЗМЕР) происходит по команде EXIT (ВЫХод) или при отмене команды DIM (РАЗМЕР).

Если надо проставить только один размер, используйте команду DIM1 (РАЗМЕР1). После ее выполнения автоматически происходит выход из режима проставления размеров и в строке подсказок появляется слово "Команда:".

Подкоманды образмеривания позволяют проставлять размеры линейные, угловые, диаметра и радиуса. Кроме того, ряд подкоманд позволяет управлять раз-

личными режимами. Ниже приведен перечень всех подкоманд проставления размеров:



3.5.2. Нанесение линейных размеров

В Автокаде выделяется четыре типа линейных размеров - HORizontal (ГОРизонтальный), VERtical (ВЕРтикальный), ALIgned (ПАРаллельный) и ROTated (ПОВернутый) (см. рисунок выше). Единственное различие между ними - угол наклона размерной линии (в подкоманде ROTated (ПОВернутый) этот угол задает пользователь).

Выносные линии линейных размеров можно проставлять как вручную, так и автоматически. При построении выносных линий вручную нужно ввести координаты начала первой выносной линии:

```
Dim: hor
First extension line origin or RETURN to select: 100,100
Second extension line origin: 200,200
```

```
Размер: гор
Начало первой выносной линии или RETURN - для выбора: 100,100
Начало второй выносной линии: 200,200
```

Если вместо указания начала первой выносной линии ответить нажатием RETURN, то система попросит указать образмериваемый примитив и по окончании ввода проставит выносные линии автоматически:

```
Dim: hor
First extension line origin or RETURN to select:
Select line, arc, or circle:
```

```
Размер: гор
Начало первой выносной линии или RETURN - для выбора:
Выберите отрезок, дугу или круг:
```

При автоматическом проставлении размеров Автокад находит конечные точки отрезков и дуг и располагает в них начальные точки выносных линий.

При проставлении размеров сегменты полилинии рассматриваются как отдельные примитивы.

После определения выносных линий следует определить положение размерной линии и размерный текст:

```
Dimension line location: 300,300
Dimension text <измеренное значение>: 2 отв. M20x1
```

```
Место положения размерной линии: 300,300
Размерный текст <измеренное значение>: 2 отв. M20x1
```

Размерный текст может состоять из введенного пользователем текста и размерного значения, измеренного Автокадом. Можно выделить четыре случая:

- Размерная строка набирается с клавиатуры. Измеренное Автокадом значение не используется; при редактировании размерный текст не меняется
- Размерная строка не вводится (пользователь нажимает RETURN). В размерной строке проставляется измеренное Автокадом значение. В этом случае при изменении базы размера в процессе редактирования Автокад изменит значение размера

- Поскольку в размерную строку могут входить пробелы, пробел здесь не дублирует функцию клавиши RETURN. Размерная строка, таким образом, может состоять из одного пробела (пустой текст). При этом размер будет без надписей
- Включение измеренного значения в набранный с клавиатуры размерный текст осуществляется специальным символом "<>". Например, если измеренное значение было равно 14.6 и пользователь ввел строку "Длина равняется <> по горизонтали", то будет получена размерная строка "Длина равняется 14.6 по горизонтали". Такой размер связан с базой размера - при изменении базы размера в процессе редактирования Автокад изменит значение размера

К измеренному значению можно автоматически добавлять текстовую строку (например, чтобы по умолчанию Автокад писал 14.6 мм). Для этого следует записать нужную строку в размерную переменную DIMPOST (РЗМСУФ) (для альтернативных единиц предусмотрена переменная DIMAPOST (РЗМАСУФ). По умолчанию эта переменная пуста (для того чтобы сделать ее пустой, нужно записать в нее точку "."). Если в процессе построения чертежа заменить один суффикс на другой, то уже отрисованные размеры останутся без изменений.

Автокад позволяет проставлять размеры с общей базой и размерные цепи - для этого существуют специальные подкоманды:

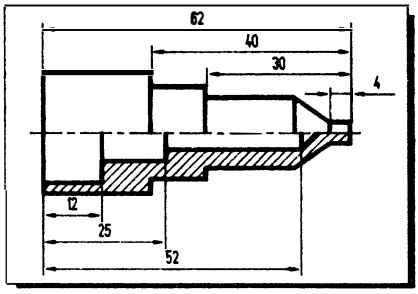
BASeline (БАЗовый)

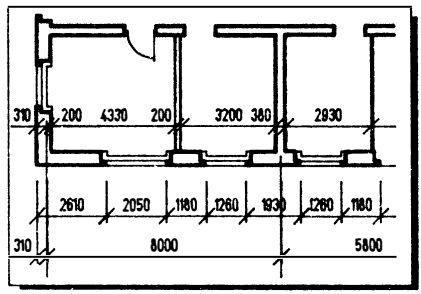
Наносит линейные размеры от первой выносной линии последнего размера, которая принимается за базу. При этом каждая новая размерная линия, чтобы не сливаться с предыдущей, смещается на заданную размерной переменной DIMDLI (РЗМОРЛ) величину

CONtinue (ПРОдолжение)

Наносит размеры цепью

По умолчанию соблюдается стандарт ISO, по которому размерный текст должен располагаться в разрыве размерной линии. (Вопрос соблюдения стандартов ЕСКД рассматривается в разделе 3.5.5.)





3.5.3. Нанесение угловых размеров

Для нанесения угловых размеров используется подкоманда ANGular (УГЛовой). После указания двух отрезков, определяющих образмериваемый угол, Автокад просит указать положение размерной дуги, а затем ввести размерный текст и указать его положение:

```
DIM: ang
Select first line:
Second line:
Enter dimension arc location:
Dimension text <измеренное значение>:
Dimension text location:
```

```
Размер: угл
Выберите первую линию:
Вторая линия:
Укажите положение размерной дуги:
Размерный текст <измеренное значение>:
Укажите место текста:
```

Если положение размерной дуги указано так, что размерная дуга не пересекает указанные объекты, то автоматически будут добавлены выносные линии.

Размерный текст может быть введен так же, как для линейных размеров. Единственное отличие заключается в том, что в угловых размерах нельзя вводить пустую строку (пробел), т.к. в этом случае Автокад разорвет размерную дугу. Если вы хотите сделать пустую размерную строку без разрыва дуги, то следует поместить размерный текст рядом с дугой, а затем воспользоваться средствами редактирования размеров. По умолчанию размерный текст помещается в разрыве дуги. Предоставляется, однако, возможность указывать другое положение размерного текста.

Для устранения неоднозначности при проставлении угловых размеров введено ограничение, что образмериваемый угол не может быть больше 180^{O}.

3.5.4. Радиальные размеры для дуг и окружностей

В размерах, проставляемых для дуг и окружностей, нет выносных линий и размерная линия проводится через центр, поэтому они проще, чем другие размеры. Для проставления размеров дуг и окружностей используются подкоманды RADius (РАДиус) и DIAmeter (ДИАметр):

```
DIM: dia
Select arc or circle:
Dimension text <измеренное значение>:
```

```
Размер: диа
Выберите дугу или круг:
Размерный текст <измеренное значение>:
```

При выборе объекта вы указываете одновременно и точку, через которую будет проведена размерная линия.

По умолчанию размерный текст также располагается в разрыве размерной линии. В разделе 3.5.5 рассказывается, как можно приблизить проставление размеров Автокадом к требованиям ЕСКД.

Если текст не помещается в разрыве размерной линии, выдается сообщение:

```
DIM: dia
Select arc or circle:
Dimension text <измеренное значение>:
Text does not fit.
Enter leader length for text: 10
```

```
Размер: диа
Выберите дугу или круг:
Размерный текст <измеренное значение>:
Текст не помещается.
Введите длину выноски для текста: 10
```

Выноска текста является продолжением размерной линии. Текст располагается всегда горизонтально.

Аналогично наносятся радиальные размеры для дуг и окружностей (подкоманда RADius (РАДиус).

3.5.5. Средства настройки

Проставление размеров осуществляется в соответствии со значениями т.н. *размерных переменных*. Таких переменных более 30. Некоторые из них являются простыми переключателями типа Вкл/Откл, другие служат для хранения числовых значений и других данных.

Чтобы задать новое значение размерной переменной, нужно из меню команды DIM: (РАЗМЕР:) выбрать опцию DimVars (РзмПерем), а затем из появившегося перечня переменных выбрать нужную (или просто полностью ввести имя переменной с клавиатуры). Появится подсказка:

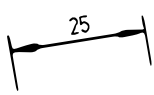
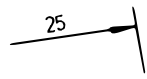
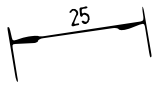
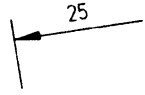
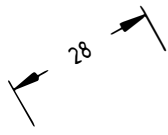
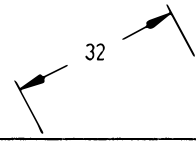
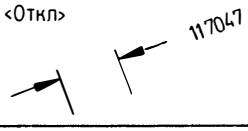
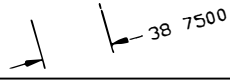
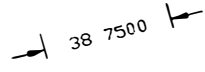
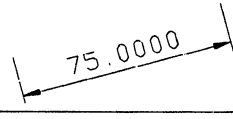
```
Current value <текущее значение> New value:
```

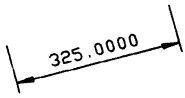
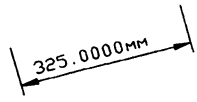
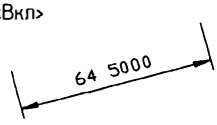
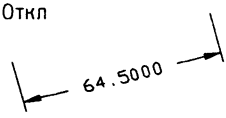
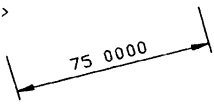
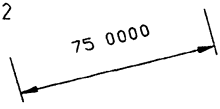
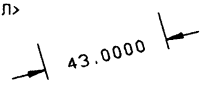
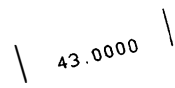
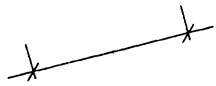
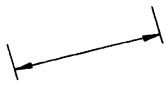
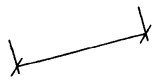
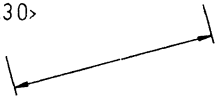
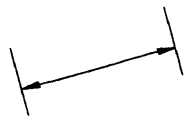
```
Текущее значение <текущее значение> Новое значение:
```

В этот момент можно ввести новое значение или подтвердить текущее, нажав RETURN.

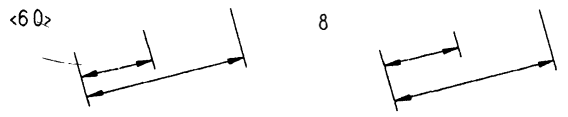
Новые значения размерных переменных можно устанавливать в процессе образмеривания (за исключением запроса "Dimention text" ("Размерный текст"). При этом новые значения размерных переменных начинают действовать при проставлении последующих размеров; в то же время уже проставленные размеры не изменяются.

Ниже приводится список размерных переменных Автокада:

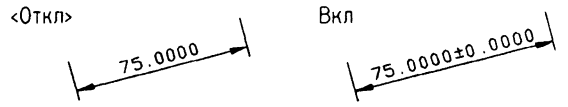
Имя и комментарии	*Первоначальное значение*	*Измененное значение*
DIMSE1 (РЗМПДВЛ1) Подавление первой выносной линии	<Откл> 	Вкл 
DIMSE2 (РЗМПДВЛ2) Подавление второй выносной линии	<Откл> 	Вкл 
DIMTIH (РЗМТМЕЖГ) Текст между размерными линиями горизонтален	<Откл> 	Вкл 
DIMTOH (РЗМТВНЕГ) Текст вне размерных линий горизонтален	<Откл> 	Вкл 
DIMTOFL (РЗМРЛМВ) Текст вне выносных линий, размерная линия внутри	<Откл> 	Вкл 
DIMTIX (РЗМТМВ) Текст между выносными линиями	<Откл> 	Вкл 
DIMTXT (РЗМТЕКСТ) Высота текста (только в том случае, если в текущей гарнитуре высота равняется нулю)	<35> 	5 

DIMPOST (РЗМСУФ) Содержит надпись, которая ставится после измеренного значения размера (в <>). Чтобы отменить суффикс, следует ввести точку (".")	<>	мм
		
DIMTAD (РЗМТНРЛ) Поместить текст над размерной линией	<Вкл>	Откл
		
DIMTVP (РЗМВПТ) Вертикальное положение текста. Работает только в случае, если DIMTAD (РЗМТНРЛ) "откл."	<0>	2
		
DIMSOXD (РЗМРЛЗВ) Размерная линия за выносными	<Откл>	Вкл
		
DIMDLE (РЗМУРЛ) Удлинение размерной линии для засечек (РЗМВЛЗАС отлична от нуля)	<0.0>	5
		
DIMTSZ (РЗМВЛЗАС) Размер засечек	<0.0>	3
		
DIMASZ (РЗМВЛСТ) Размер стрелок (РЗМВЛЗАС = 0)	<3.0>	5
		
DIMEXO (РЗМОВЛ) Отступ выносной линии	<0.0>	3
		
DIMEXE (РЗМПВЛ) Продолжение выносной линии	<3.0>	6
		

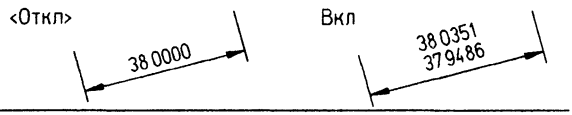
DIMDLI (РЗМОРЛ)
Отступ размерной линии



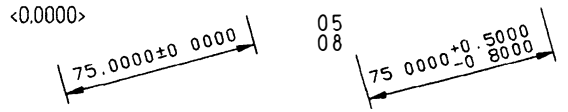
DIMTOL (РЗМДОП)
Генерация размерных допусков



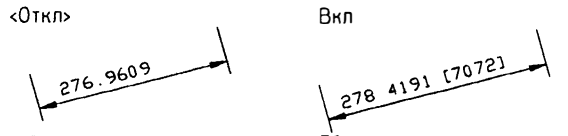
DIMLIM (РЗМПРД)
Генерация размерных пределов



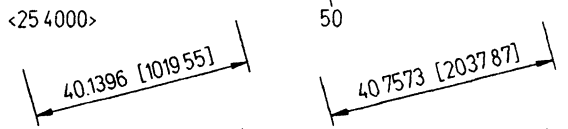
DIMTP (РЗМДПЛ)
DIMTM (РЗМДМИН)
Плюс/минус допуск



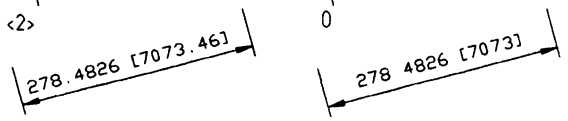
DIMALT (РЗМАЛЬТ)
Выбор альтернативных единиц.



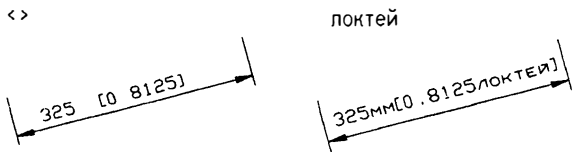
DIMALTF (РЗМАЛЬТФ)
Масштабный фактор для получения размеров в альтернативных единицах.



DIMALTD (РЗМАЛЬТД)
Количество знаков после запятой для альтернативных единиц



DIMAPOST (РЗМАСУФ)
То же, что DIMPOST (РЗМСУФ), но для альтернативных единиц



DIMASO (РЗМАССО) — Создание ассоциативных размеров. Если эта переменная равна нулю, то размер автоматически расчленяется после отрисовки

DIMSHO (РЗМСЛЕЖ) — Переопределение размеров при слежении. Если эта переменная равна 1, то размеры будут пересчитываться в процессе слежения (рекомендуется отключать)

DIMCEN (РЗМЦЕНТ) Задает размер маркера центра или осевые линии	<0.00>	2	-18
DIMBLK (РЗМБЛК) Служит для переопределения блоков стрелок	< >		DOT
DIMSAH (РЗМЗБС) DIMBLK1 (РЗМБЛК1) DIMBLK2 (РЗМБЛК2) Служит для задания разных блоков стрелок одной размерной линии	< >		< > DOT
DIMRND (РЗМОКР) Точность округления размеров	<0>	37.6710	0.5 37.5000
DIMSCALE (РЗММАСШТ) Глобальный масштабный коэффициент. Не действует на допуски, измеренные длины и углы	<1.0>	57.5000	2 57.5000
DIMLFAC (РЗМДЛФ) Масштабный фактор длины	<1.0>	64.0000	5 320.0000
DIMZIN (РЗМДФН)	Нуль в формате футы/дюймы		

Настройка в соответствии с ЕСКД

Стандартная поставка ориентирована на международные стандарты ISO, значительно отличающиеся от советских. И все же Автокад настолько гибок, что проставление размеров можно настроить достаточно близко к требованиям ЕСКД.

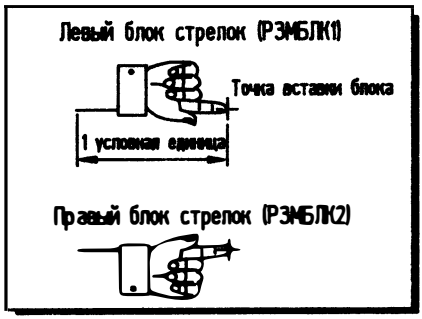
Для осуществления такой настройки необходимо выполнить определенные действия.

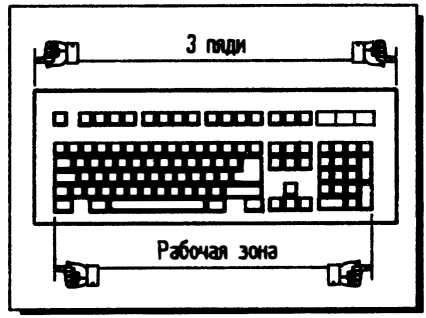
- Для того чтобы текст располагался над размерной линией как в линейных размерах, так и в размере "диаметр", надо отключить DIMTAD (РЗМТНРЛ) и занести в переменную DIMTVP (РЗМВПТ) расстояние от основания текстовой строки до размерной линии (оно должно быть больше 1)
- Количество знаков, проставляемых в размерах после запятой, определяется точностью, задаваемой командой UNITS (ЕДИНИЦЫ)
- Отступ размерной линии от объекта определяется переменной DIMEXO (РЗМОВЛ)
- Для того чтобы размерный текст был параллелен размерной линии при его размещении как между выносными линиями, так и вне их, нужно отключить переменные DIMTIH (РЗМТМЕЖГ) и DIMTOH (РЗМТВНЕГ)

Определение стрелок пользователя

В стандартной поставке на концах размерной линии может рисоваться либо стрелка, либо засечка. Существует, однако, возможность создавать свои собственные "стрелки", которые будут отрисовываться автоматически при проставлении линейных размеров (к сожалению, на проставление других размеров определение блоков стрелок не влияет).

Для того чтобы вместо стрелок в линейных размерах рисовались ваши блоки стрелок, нужно сначала определить сами блоки так, как показано на рисунке:





Создав свои блоки стрелок, нужно отключить переменную DIMTSZ (РЗМВЛЗАС) и включить переменную DIMSAH (РЗМБС). После этого следует имя левого блока стрелки занести в переменную DIMBLK1 (РЗМБЛК1), а имя правого блока - в переменную DIMBLK2 (РЗМБЛК2). Если блок стрелок симметричен относительно размерной линии (может использоваться как для левой, так и для правой стрелки), то следует отключить переменную DIMSAH (РЗМБС) и имя блока занести в переменную DIMBLK (РЗМБЛК).

Если требуется отменить установку переменных DIMBLK (РЗМБЛК), DIMBLK1 (РЗМБЛК1) или DIMBLK2 (РЗМБЛК2), то надо занести в них точку.

3.5.6. Редактирование размеров

В большинстве случаев выносные линии начинаются от концов указанного примитива. Однако при проставлении размеров база размера определяется не концами выносных линий, а т.н. *определяющими точками*, которые помещаются в характерных точках образмериваемых примитивов (характерные точки совпадают с концами выносных линий только в том случае, если отступ выносных линий от образмериваемого примитива, определяемый переменной DIMEXO (РЗМОВЛ), равен нулю). Для диаметров и радиусов определяющие точки помещаются на концах размерной линии. Кроме "размерных" определяющих точек создается также точка, определяющая положение размерного текста (это позволяет перемещать текст в процессе редактирования).

Определяющие точки помещаются на слой DEFPOINTS, но при отключении слоя они остаются видимыми. Отключение слоя DEFPOINTS, однако, запрещает вывод определяющих точек на принтер или плоттер, и поэтому может быть полезным.

При редактировании размеров (например, при использовании команды STRETCH (РАСТЯНИ) следует учитывать существование определяющих точек. Если определяющие точки не попали в набор объектов, выбранных для редактирования, то размер изменен не будет.

3.6. Блоки и атрибуты

Примитивы могут быть простыми, сложными и составными. Составные примитивы в Автокаде называются блоками. Перечислим преимущества использования блоков:

- Блоки хранятся отдельно от остального чертежа, и один и тот же блок может использоваться в чертеже многократно, что позволяет сократить время создания чертежа, упростить редактирование и сэкономить память на диске
- Блок можно вставлять в разрабатываемый чертеж в любом масштабе и под любым углом. Это позволяет быстро и легко создавать изображения из простых символов
- Если вы постоянно работаете с одной предметной областью, то вы можете, один раз создав свою "библиотеку" стандартных и типовых элементов, использовать ее в своей работе, что значительно ускоряет процесс проектирования. Такие библиотеки, состоящие из блоков, вместе со специализированными меню позволяют создавать на базе Автокада законченные прикладные системы для автоматизации проектирования
- Блоки отличаются от других примитивов также тем, что при создании блока пользователь присваивает ему имя, по которому он впоследствии

отличается от других блоков. Поэтому, определив новый блок с тем же именем, что и старый (при этом старый блок будет удален), мы заменим блок во всем чертеже

Все команды создания блоков, изучаемые в этой главе, собраны в подменю экранного меню - меню BLOCKS (БЛОКИ).

Работать с блоками очень просто. Команда BLOCK (БЛОК) позволяет создать блок, команда INSERT (ВСТАВЬ) - разместить их, а WBLOCK (ПБЛОК) - записать блок на диск.

3.6.1. Создание блока

Для создания блока из фрагментов рисунка нужно вызвать команду BLOCK (БЛОК) из подменю BLOCKS (БЛОКИ) экранного меню.

Сначала Автокад запрашивает имя блока:

```
Command: BLOCK
Block name (or ?):
Insertion base point:
```

```
Команда: БЛОК
Имя блока (или ?):
Базовая точка вставки:
```

Наряду с именем надо определить т. н. базовую точку вставки блока. Эту точку можно рассматривать как "ручку", за которую определенная блоком группа объектов будет вставляться в чертеж. При вставке блока вы сможете вращать блок вокруг точки вставки, а также изменять размеры блока отдельно по X и Y с центром гомотетии в точке вставки. Точка вставки является началом локальной системы координат, связанной с блоком, оси которой параллельны осям текущей системы координат в момент определения блока. После вставки блока в рисунок эта система координат поворачивается так, чтобы ее оси были параллельны осям текущей системы координат рисунка в момент вставки блока.

После этого нужно заполнить набор объектов, из которого будет создано определение блока:

```
Select objects:
```

```
Выберите объекты:
```

В момент создания блока объекты удаляются из чертежа. При желании их можно восстановить, использовав команду OOPS (ОЙ).

Блоки, создаваемые с использованием команды BLOCK (БЛОК), хранятся только в рисунке, и для того, чтобы блок можно было включить в другие рисунки, его определение нужно записывать в отдельный файл, используя команду WBLOCK (ПБЛОК).

3.6.2. Вставка блока

Вставка ранее определенного блока в рисунок осуществляется командой INSERT (ВСТАВЬ). Для вставки блока в рисунок нужно указать имя блока:

```
Command:INSERT
Block name (or ?):
```

```
Команда:ВСТАВЬ
Имя блока (или ?):
```

Эта команда позволяет также вывести на текстовый экран список определенных в рисунке блоков. Для этого нужно в ответ на запрос "Имя блока (или ?):" ввести знак вопроса:

```
Block name (or ?):?
```

```
Имя блока (или ?):?
```

После этого Автокад переключит экран в текстовый режим и выведет список всех определенных в рисунке блоков.

При вставке блока в чертеж вы должны указать точку вставки. После этого нужно задать масштабные коэффициенты и угол поворота блока при вставке:

```
Scale factor <1>/Corner/XYZ:
Scale factor <default x>:
Rotation angle <0>:
```

```
Масштаб по оси X <1>/Угол/XYZ:
Масштаб по оси Y <по умолчанию = X>:
Угол поворота <0>:
```

Если вы не хотите изменять масштаб по осям X и Y, то нужно принять значение по умолчанию, которое равно 1. Угол поворота по умолчанию равен 0. Для принятия значений по умолчанию следует нажимать клавишу RETURN или правую кнопку "мыши".

Интересно, что если указать отрицательный масштабный коэффициент по какой-то координате (или по обеим координатам), то при вставке блока будет осуществлено зеркальное отображение по той из осей, по которой задан отрицательный масштабный коэффициент (или по обеим осям).

Масштабный прямоугольник

Масштаб по X и Y можно указывать одновременно. Для этого служит опция Angle (Угол) команды INSERT (ВСТАВЬ). В тот момент, когда Автокад просит вас ввести масштаб по оси X, можно не вводить с клавиатуры масштабный коэффициент, а указать вторую точку. При этом Автокад вычислит масштабные

коэффициенты по осям X и Y в соответствии со сторонами прямоугольника, одним углом которого является точка вставки, а другим - указанная вами точка (вот почему эта опция называется "Угол").

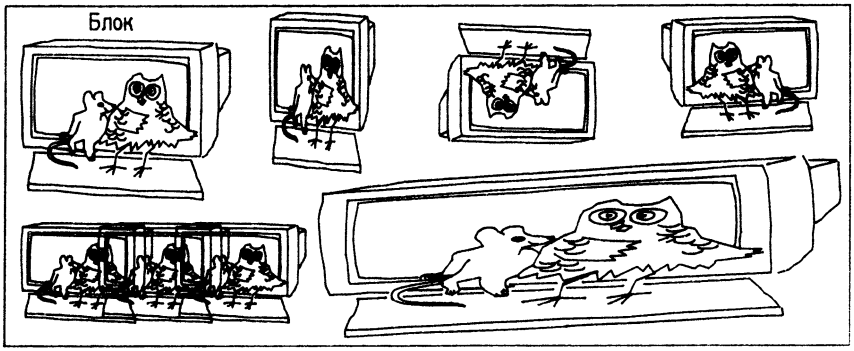
Эту опцию не надо выбирать из меню или набирать с клавиатуры - она включается автоматически. Включите режим слежения, вызовите команду INSERT (ВСТАВЬ) и укажите точку вставки. Система выдаст запрос:

```
Scale factor <1>/Corner/XYZ:
```

```
Масштаб по оси X <1>/Угол/XYZ:
```

Не указывайте масштаб, а подвигайте курсор. Вы увидите, как Автокад масштабирует вставляемый блок, причем если перекрестие курсора находится левее или ниже точки вставки, то блок зеркально отображается по одной (или обеим) осям координат.

Последним шагом при вставке блока является задание угла поворота вокруг оси Z. Можно оставить угол поворота равным значению по умолчанию "0" - не поворачивать блок.



Вставка блока с предварительным
заданием масштабных коэффициентов и угла поворота

Обычно команда INSERT (ВСТАВЬ) сначала просит указать точку вставки, а затем масштабы по X и Y и угол поворота. Это не всегда удобно - масштаб (размеры) блока может быть жестко фиксирован, а изменять можно только его точку вставки.

В этом случае можно сначала указать масштаб и угол поворота блока, а затем, используя режим слежения, наилучшим образом разместить в чертеже блок фиксированных размеров. Для предварительного задания параметров вставки следует в тот момент, когда команда INSERT (ВСТАВЬ) просит задать точку вставки, указать одну из опций, описанных ниже. Отметим сразу, что все опции

масштаба имеют своих "двойников" с буквой "Р" ("П") (от слова "Pre-" ("Проба") в начале. Эти опции масштабируют блок при слежении, но вставляется немасштабированный блок. Сделано это для того, чтобы вы могли попробовать, как будет выглядеть блок при том или ином масштабе. Поскольку в режиме пробы масштабный коэффициент в действительности не устанавливается, то после указания точки вставки команда INSERT (ВСТАВЬ) запросит параметры вставки обычным образом.

Scale (Масштаб) и PScale (ПМасштаб)
Запрашивают масштабный коэффициент. Коэффициент можно указать заданием двух точек, при этом за масштаб берется расстояние между этими точками в принятых единицах. Отслеживание прерывается до тех пор, пока не будет задан масштаб; после указания масштаба отслеживание продолжается, но уже в заданном масштабе. Этот масштаб общий по всем осям координат

Xscale (Хмасштаб) и PXScale (ПХмасштаб)
То же, что и Масштаб, но только по оси X

Yscale (Умасштаб) и YScale (ПУмасштаб)
То же, что и Масштаб, но только по оси Y

ZScale (Zмасштаб) и PZScale (ПZмасштаб)
То же, что и Масштаб, но только по оси Z

Rotate (Поворот) и PRotate (ППоворот)
Запрашивают угол поворота. Угол может быть указан заданием вектора. Отслеживание прерывается до тех пор, пока не будет задан масштаб; после указания масштаба блок продолжает отслеживаться, но уже в заданном масштабе

Эти опции можно указывать многократно и в любой последовательности. Ясно, что если вы, выбрав нужный масштаб при помощи, скажем, опции PZScale (ПМасштаб) укажете опцию Scale (Масштаб), то тем самым установите масштаб вставки блока и в дальнейшем масштаб запрашиваться не будет.

Свойства блока, расчленение при вставке

При определении блока все входящие в него примитивы образуют единый составной примитив, который заносится во внутреннюю библиотеку блоков текущего чертежа. При вставке блока командой INSERT (ВСТАВЬ) собственно в чертеж в него в действительности вставляются не примитивы, образующие блок, а сам блок, т.е. определяются точка вставки, имя блока, числовые коэффициенты вставки блока и другие свойства. При редактировании такого блока он будет рассматриваться как один объект (например, командой MOVE (ПЕРЕНЕСИ) или ERASE (СОТРИ). В действительности редактироваться будет не определение блока, а характеристики вставки блока (координаты точки вставки, масштаб - этот тип данных называется в Автокаде Block Reference, "Ссылка на блок"). Команда MOVE (ПЕРЕНЕСИ), например, модифицирует точку вставки блока; команда ROTATE (ПОВЕРНИ) - угол поворота блока и т.п.

Можно перенести в рисунок не блок, а собственно примитивы из определения блока (это может понадобиться при редактировании). Здесь есть два способа. Первый заключается в том, что после вставки блока командой INSERT (ВСТАВЬ) используется команда EXPLODE (РАСЧЛЕНИ), в результате выполнения которой блок удаляется из чертежа, а на его место вставляются примитивы из определения блока, причем они при этом модифицируются в соответствии с параметрами вставки блока. Следует отметить, что нельзя расчленить блок, который вставлен с неравными масштабами по осям X и Y.

Второй способ состоит в том, что блок расчленяется в момент вставки, что может сэкономить наше время. Для этого следует на запрос команды CHANGE (ИЗМЕНИ) перед именем блока ввести звездочку:

```
Command: BLOCK
Block name (or ?): *ИМЯ
```

```
Команда: БЛОК
Имя блока (или ?): *ИМЯ
```

При этом в чертеж будут вставлены примитивы из определения блока, а не сам блок. По понятным причинам после этого вас попросят задать не два масштаба, а только один, общий для X и Y.

Есть несколько тонкостей, связанных с установкой свойств примитивов при определении блока. Дело в том, что видимость примитива и то, на каком слое он находится, взаимодействуют сложным образом.

- Если примитивы, из которых состоит блок, имели постоянные свойства (цвет и тип линии не **по блоку** и не **по слою**) и при вставке в блок находились не на слое "0", то при вставке блока они будут отрисованы не на текущем слое, а на том слое, на котором были определены, и тем цветом, который они имели при записи в блок. Более того, командой CHANGE (ИЗМЕНИ) нельзя будет изменить их цвет или тип линии
- Если примитивы, из которых состоит блок, имели переменные свойства (цвет и тип линии **по блоку** или **по слою**) и при вставке в блок находились не на слое "0", то при вставке блока они будут также отрисованы не на текущем слое, а на том слое, на котором были определены, и тем тем цветом, который они имели при записи в блок. Однако цвет их можно будет менять изменением цвета слоя (в случае цвета **по слою**) или командой CHANGE (ИЗМЕНИ) (в случае цвета **по блоку**)
- Если примитивы, из которых состоит блок, имели переменные свойства (цвет и тип линии **по блоку** или **по слою**) и при вставке в блок находились на слое "0", то при вставке блока они будут отрисованы на текущем слое с цветом текущего слоя и цвет их можно будет менять изменением цвета текущего слоя (в случае цвета **по слою**) или командой CHANGE (ИЗМЕНИ) (в случае цвета **по блоку**)

3.6.3. Вставка блока массивом

Команда MINSERT (МВСТАВЬ) на первый взгляд эквивалентна последовательности команд INSERT (ВСТАВЬ) и ARRAY (МАССИВ), поскольку формирует прямоугольный массив блоков, запрашивая при вставке шаг по X и по Y. Формат ее мы здесь не приводим, поскольку там нет никаких тонкостей. Однако объект, который получается в результате работы этой команды, несколько необычен с точки зрения пользователя это блок, но он почему-то не расчленяется командой EXPLODE (РАСЧЛЕНИ) (и поэтому его нельзя расчленить при вставке использованием знака "*", как в команде INSERT (ВСТАВЬ). На самом деле получается объект того же типа, что и при работе команды INSERT (ВСТАВЬ), - ссылка на определение блока ("block reference"). Воспользуйтесь командой MINSERT (МВСТАВЬ), а затем определите тип получившегося объекта командой LIST (СПИСОК). Вы увидите, что получившийся объект отличается от того, что пользователь называет блоком, только ненулевым шагом по горизонтали и вертикали. Запрет на расчленение такого объекта наложен, вероятно, из желания защититься от возможного переполнения памяти.

Команда MINSERT (МВСТАВЬ) в некоторых случаях может весьма значительно сэкономить память - ведь даже если вы создали командой MINSERT (МВСТАВЬ) массив размером 1000x1000, в графическом файле на самом деле хранится всего один объект! Если вам хочется редактировать некоторые элементы массива и в то же время сэкономить память, следует все же воспользоваться сначала командой INSERT (ВСТАВЬ), а затем ARRAY (МАССИВ) (и для редактируемых примитивов EXPLODE (РАСЧЛЕНИ).

3.6.4. Запись блока на диск

Эта команда служит для записи всего рисунка или его фрагментов в отдельный файл и имеет формат вида:

```
Command: WBLOCK
File name:
Block name:
```

```
Команда: ПБЛОК
Имя файла:
Имя блока:
```

Здесь **имя файла** - имя вновь создаваемого файла, в котором будет содержаться записываемый фрагмент. Имя файла следует указывать без расширения - оно полагается .dwg по умолчанию. Далее нужно указать **имя блока**. Возможны следующие ответы на этот запрос:

ИМЯ В файл записывается блок ИМЯ (если он определен в текущем рисунке)

= В файл записывается блок с именем, которое было вами указано как имя файла (если он определен в текущем рисунке)

* Записывается весь рисунок, неиспользуемые блоки не записываются

ПРОБЕЛ При таком ответе Автокад просит указать базовую точку вставки и выбрать объекты, как в команде INSERT (ВСТАВЬ). Выбранные объекты записываются в отдельный файл и удаляются из текущего рисунка. Восстановить их можно командой OOPS (ОЙ)

3.6.5. Целые рисунки как блоки

Целые рисунки можно рассматривать как блоки с точкой вставки, расположенной в начале координат рисунка. Поэтому команда INSERT (ВСТАВЬ) имеет важную особенность: если заданный блок не найден в текущем рисунке, то на диске ищется рисунок с этим именем, который можно затем вставить в чертеж. При вставке сначала определяется новый внутренний блок текущего рисунка с именем, совпадающим с именем файла рисунка, затем предлагается вставить вновь определенный блок. Таким образом рисунок-блок "загружается" в файл.

При загрузке рисунка в блок можно автоматически переопределить существующий блок. Для этого нужно в команде INSERT (ВСТАВЬ) указать имя старого блока, "приравняв" его новому. Предположим, например, что в нашем чертеже имеется блок KNOB, который мы хотим заменить на блок HANDLE, находящийся в отдельном файле handle.dwg. Это можно сделать следующим образом:

```
Command: INSERT
Block name (or ?): KNOB=HANDLE
```

```
Команда: ВСТАВЬ
Имя блока (или ?): KNOB=HANDLE
```

При этом старый блок KNOB будет заменен на новый блок HANDLE. Дальнейшие запросы команды INSERT (ВСТАВЬ) можно отменить. Если имя нового блока не указывать (ввести KNOB=), то Автокад будет искать на диске файл блока с тем же именем.

Точку вставки рисунка, как блок, можно переопределить. Для этого надо использовать команду БАЗА. Формат ее мы не приводим - он очевиден. Базовая точка вставки хранится в системной переменной INBASE, которую можно изменить также средствами Автолиспа или командой SETVAR (УСТПЕРЕМ). Заметим, что командой BASE (БАЗА) можно определить ненулевую координату Z.

Отметим, что если, скажем, в рисунке test определены блоки A, B, C и т.п., то при вставке рисунка test в новый чертеж не только будет создан новый блок test, но в новый чертеж будут перенесены блоки A, B, C и т.п. - все блоки, определенные в блоке test. Это позволяет систематизировать библиотеки блоков - не хранить на диске огромное количество самостоятельных файлов-блоков.

Сказанное относится не только к блокам, но и ко всем поименованным объектам файла-чертежа, вставляемого как блок (слои, типы линий и гарнитуры шрифтов), за исключением поименованных видов. Если вставленный и текущий рисунки содержат объекты с совпадающими именами, то приоритет имеет определение в *текущем* рисунке.

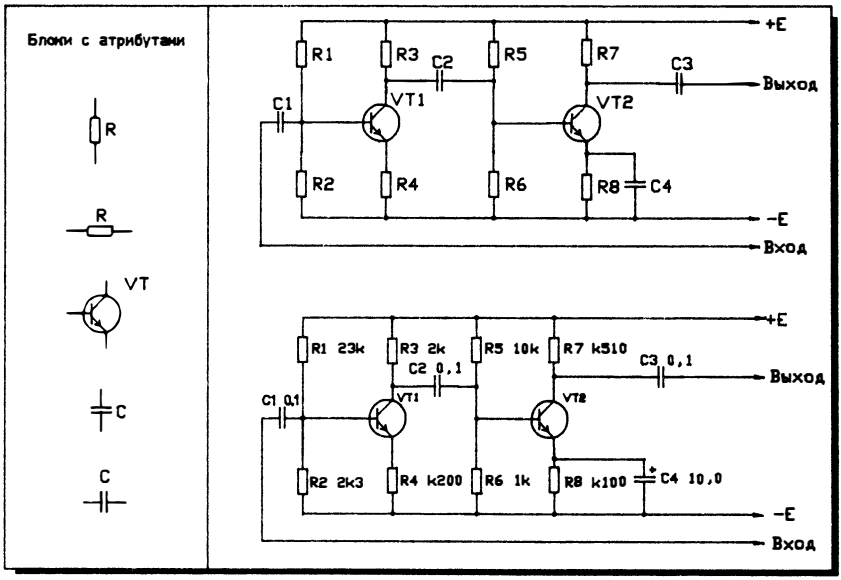
Неиспользуемые в рисунке блоки можно удалить из чертежа подкомандой PURGE Block (УДАЛИ Блок). Отметим, что удалять блоки можно только в начале сеанса редактирования, поэтому если вы хотите удалить блок, то нужно выйти из графического редактора, опять войти в него, вызвав тот же рисунок, и только после этого можно использовать команду ERASE (УДАЛИ). Блоки можно переименовывать подкомандой RENAME Block (НОВОЕИМЯ Блок).

3.6.6. Переопределение блоков

Вы можете осуществить глобальную замену блоков в вашем графическом файле - это позволит вам на практике применить методологию проектирования сверху вниз. Сложный чертеж можно создать сначала с использованием простых блоков, а затем переопределить их, заменив на более сложные. Так работать очень удобно: во-первых, не загромождается чертеж, что позволяет сосредоточиться на главном; во-вторых, простой чертеж быстрее регенерируется - ускоряется работа над чертежом; в-третьих, вы всегда можете усовершенствовать систему, быстро и просто заменив старый вариант на новый.

3.6.7. Атрибуты

Атрибут - специальный примитив Автокада, предназначенный для работы с блоками. Будучи записан в блок, он служит в качестве текстовой переменной - "ячейки", в которую при вставке блока можно записать некоторую строку. Это позволяет, однажды создав блок, в который входит атрибут, с каждой вставкой блока в чертеж связывать новую текстовую строку (гарнитура и параметры отрисовки строки определяются атрибутом и едины для всех вхождений блока).



Атрибуты, как это хорошо видно из рисунка, особенно удобны при создании различных схем, в которых один и тот же графический символ отрисовывается несколько раз с разными обозначениями:

Средствами Автолиспа можно извлекать из чертежа значения атрибутов и передавать эти значения во внешнюю программу (например, в базу данных) или, наоборот, конструировать схему на основе внешней информации.

Создание атрибутов

Для того чтобы включить атрибут в блок, сначала нужно создать атрибут командой ATTDEF (АТОПР). После создания атрибута он отображается на экране как текстовая строка с именем атрибута (все характеристики отображения текста сохраняются).

Уже созданный атрибут можно записать в блок. Очевидно, в блок может быть записано и несколько атрибутов. Удобно, например, один раз отрисовать рамку для основной надписи чертежа, графы которой оформить в виде атрибутов, и записать ее вместе с атрибутами в блок. Это позволит, во-первых, не отрисовывать каждый раз рамку для основной надписи и, во-вторых, удобно заполнять графы основной надписи, не заботясь об оформлении текста. Если в группе чертежей основные надписи выполнены таким образом, то можно автоматически создавать спецификацию, извлекая из граф-атрибутов нужную информацию.

Для создания атрибута используется команда ATTDEF (АТОПР), которая, кроме специфических свойств атрибутов, требует определения свойств отображения текстовой строки (положения, размера и гарнитуры шрифта):

```
Command: ATTDEF
Attribute modes -- Invisible: N Constant: N Verify: N Preset: N
Enter (ICVP) to change, RETURN when done:
Attribute tag: Resistor
Attribute prompt: Enter tag:
Default attribute value: R1
Start point or Align/Center/Fit/Middle/Right/
Style:
Height <0.2000>: 20
Rotation angle <0>:
```

```
Команда: АТОПР
Атрибут -- Скрытый: Н Постоянный: Н Контролируемый: Н Установленный: У
Введите (СПКУ) и нажмите RETURN:
Имя атрибута: Сопротивление
Подсказка атрибута: Введите обозначение:
Значение атрибута по умолчанию: R1
Начальная точка или ВПИсанный/Центр/ВЫравненный/Середина/ВПРаво/
Гарнитура:
Высота <0.2000>: 20
Угол поворота <0>:
```

Перечислим свойства, которые может иметь атрибут:

Invisible (Скрытый)

Атрибут невидим при вставке блока. Этот режим следует назначать тем атрибутам, которые непосредственно в чертеже не используются, а предназначены для работы со внешними программами (например, базой данных). Управление видимостью атрибутов осуществляется командой ATTDISP (АТЭКР)

Constant (Постоянный)

Фиксированное значение атрибута, одно и то же для всех вхождений блока. Редактировать такой атрибут нельзя

Verify (Контролируемый)

В момент вставки блока при присвоении такому атрибуту значений выдается дополнительный запрос, в котором предлагается проверить, то ли значение было введено (проверка правильности ввода)

Preset (Установленный)

Значение атрибута не запрашивается при вставке блока, но может быть изменено в процессе редактирования атрибута. Если для ввода атрибутов используется диалоговое окно, то режим предварительной установки игнорируется

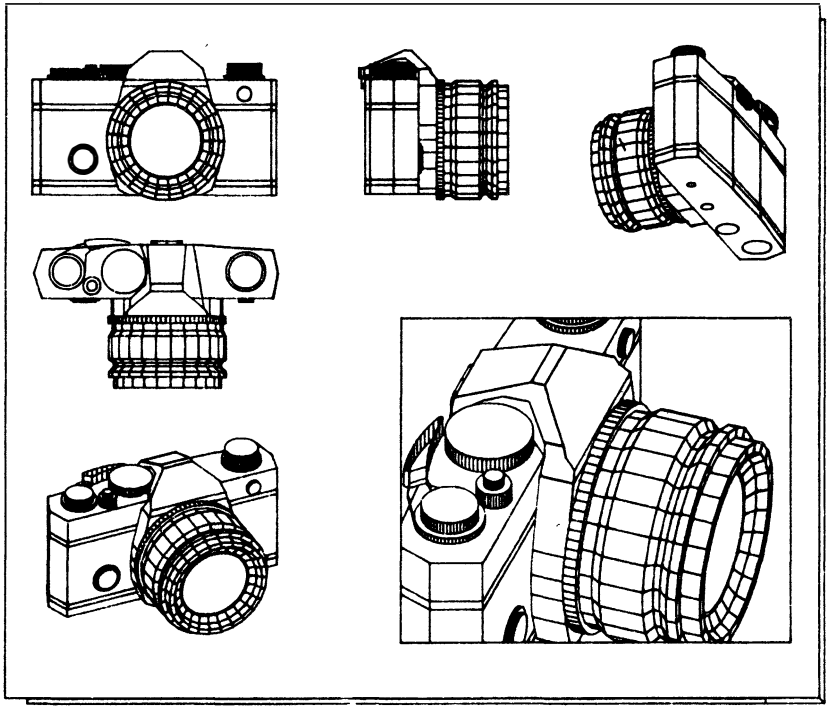
После этого следует записать атрибуты в блок. При вставке блока в чертеж Автокад будет запрашивать значения атрибутов в том случае, если системная переменная ATTREQ не равна нулю.

Существует возможность ввода значений атрибутов не в командной строке, а через диалоговое окно. Для этого следует установить системную переменную ATTDIA равной 1.

Информацию об атрибутах можно извлечь из файла командой ATTEXP (АТЭКСП). Эта команда записывает информацию об атрибуте в файл в стандартном формате СУБД типа DBASE III.

Редактирование атрибутов

Поскольку атрибут является частью блока, большинство команд редактирования не может редактировать атрибут отдельно от блока. Предусмотрена, однако, возможность редактирования значения атрибута. Для редактирования атрибутов можно воспользоваться либо командой ATTEDIT (АТРЕД), которая предоставляет широкие возможности редактирования всех атрибутов чертежа, либо командой DDATTR (ДИАЛАТР).



ГЛАВА 4

СЛУЖЕБНЫЕ СРЕДСТВА

В предыдущих главах мы подробно описали среду графического редактора, новые понятия, введенные в Автокаде, и остановились на наборе команд отрисовки примитивов.

Стремясь сохранить цельность изложения мы, однако, до сих пор ничего не говорили о служебных средствах Автокада, которые существенно облегчают и ускоряют работу конструктора в среде САПР, позволяют легко поддерживать точность геометрических построений.

Хотя вы наверняка уже самостоятельно освоили некоторые команды (например, подкоманду управления изображением ZOOM All (ПОКАЖИ Все) или ZOOM Window (ПОКАЖИ Рамка), мы надеемся, что в этой главе вы почерпнете для себя много полезного.

4.1. Настройка графического редактора

Автокад - гибкий пакет. Он и не может быть другим, потому что это пакет САПР общего применения, т.е. такой, который можно использовать везде, где используется инженерная графика. Начав практическую работу с графическим редактором, вам, скорее всего, захочется установить нужную систему единиц, размеры листа и масштаб чертежа и т.п. Но даже если вас устраивают настройки стандартной поставки, полезно уделить некоторое время изучению средств и приемов работы с графическим редактором - тем самым вы сэкономите впоследствии много времени и сил.

С одной и той же поставкой Автокада может работать несколько специалистов разного профиля, и при этом им не придется каждый раз в начале сеанса настраивать систему под свою предметную область. Дело в том, что почти все режимы работы Автокада управляются т.н. системными переменными, значения которых Автокад записывает по окончании сеанса работы не в файл конфигурации системы, а в текущий рисунок. Поскольку все настройки хранятся в файлах чертежей, начиная новый рисунок, Автокад нуждается в т.н. прототипе - чертеже, из которого Автокад переносит на новый чертеж все настройки (фактически при создании нового чертежа редактируется прототип). Поэтому если редактируется старый чертеж, то полностью восстанавливается то состояние графического редактора, которое имело место на момент окончания предыдущего сеанса. С одной стороны, такая "идеология" увеличивает размер файлов чертежей (примерно на 5 Кбайт), а с другой стороны, придает системе необычайную гибкость, поскольку разные группы пользователей могут работать с разными прототипами.

В поставку Автокада входит стандартный прототип (файл acad.dwg), в котором установлены настройки по умолчанию. Это обычный рисунок Автокада - вы

можете войти в него и изменить настройки, после чего они будут автоматически переноситься на вновь создаваемые чертежи.

Прототип не обязательно должен иметь имя acad.dwg - прототипом может быть любой чертеж. Для того чтобы взять, например, рисунок set.dwg в качестве прототипа, следует при вводе имени нового файла "приравнять" его имени прототипа:

```
Enter selection:1
Enter NAME of drawing:NEW=SET
```

```
Ваш выбор:1
Введите ИМЯ рисунка:NEW=SET
```

Отметим еще раз, что прототип полностью копируется в новый файл чертежа, поэтому если в прототипе уже были объекты, то они появятся в новом рисунке (так можно автоматизировать, например, отрисовку рамки основной надписи, хотя есть и другой, более универсальный способ).

Если вы случайно стерли файл прототипа acad.dwg, то можете создать его следующим образом:

```
Enter selection:1
Enter NAME of drawing:ACAD=
```

```
Ваш выбор:1
Введите ИМЯ рисунка:ACAD=
```

При таком способе создания файла чертежа Автокад не пользуется никаким прототипом, а создает пустой рисунок с настройками по умолчанию.

В такой развитой системе, как Автокад, вряд ли можно рекомендовать единые приемы работы - возможно, какие-то средства вы найдете удобными, а от других из-за специфики вашей работы или по другим причинам вам придется отказаться. Скорее всего, вы не будете активно заниматься модификацией меню Автокада или работать с Автолиспом, однако все же полезно знать, что настройка Автокада включает в себя не только установку удобных режимов рисования и единиц измерения. Углубляясь в настройки Автокада, можно:

- настроить режимы графического редактора (выбрать нужный масштаб, единицы измерения, тип линий и текста)
- использовать рисунки-прототипы как средство хранения настроек и стандартных элементов чертежей
- создать собственную библиотеку стандартных элементов чертежей

- использовать средства Автолиспа для создания программ автоматизации отрисовки топологически подобных объектов (параметризации чертежей типовых объектов)
- создавать свои собственные меню с учетом особенностей ваших задач

Все эти средства и возможности будут последовательно описаны в этой и следующих главах.

4.1.1. Единицы измерения

При создании чертежа на листе ватмана конструктор ограничен форматом чертежа. В Автокаде благодаря средствам отображения (в частности, масштабирования) появляется возможность создавать чертеж в реальном масштабе - 1:1, вне зависимости от того, чертит ли он здание, карту города или пишущий узел шариковой ручки. При выводе на плоттер конструктору остается только выбрать масштаб чертежа в соответствии с форматом.

Как уже говорилось в главе 1, в Автокаде все координаты измеряются в т.н. условных единицах, которые для конструктора являются реальными абсолютными размерами чертежа. Выбранные единицы измерения при этом определяют только формат представления чисел. В Автокаде имеется возможность выбирать:

- единицы измерений (формат представления условных единиц): научные, десятичные, инженерные, архитектурные
- точность отображения чисел (число знаков после запятой; при использовании архитектурных или дробных единиц задается наименьшая дробь)
- вид представления углов (в градусах с десятичными долями, в градусах/минутах/секундах/, градах (90^{O}=100 град), радианах или картографических единицах)
- точность отображения углов (число знаков после запятой) и направление оси отсчета углов (по умолчанию параллельно положительному направлению оси OX)
- направление отсчета углов (по умолчанию - по часовой стрелке)

Для установки формата представления и точности отображения единиц измерения в Автокаде существует команда UNITS (ЕДИНИЦЫ).

Следует еще раз оговориться, что в графической базе данных Автокада все координаты и углы хранятся в мировой системе координат в условных единицах и команда UNITS (ЕДИНИЦЫ) позволяет изменять только формат и точность отображения данных, а не их представление в чертеже. Отсюда следует, что ей можно пользоваться всякий раз, когда это представляется удобным (например, если вам удобнее отсчитывать углы по часовой стрелке, то вы можете легко изменить направление отсчета углов).

Выберите пункт UNITS: (ЕДИНИЦЫ:) из меню SETTINGS (НАСТРОЙ). На экране появится меню выбора формата представления чисел:

```
Command: UNITS
Systems of units          (Examples):

1. Scientific             1.55E+01
2. Decimal                15.50
3. Engineering            1'-3.50"
4. Architectural          1'-3 1/2"
5. Fractional             15 1/2

Enter choice, 1 to 5 <текущее значение>:
```

```
Команда: ЕДИНИЦЫ
Единицы измерения         (Примеры):

1. Научные                1.55E+01
2. Десятичные             15.50
3. Технические            1'-3.50"
4. Архитектурные          1'-3 1/2"
5. С дробной частью       15 1/2

Ваш выбор - от 1 до 5 <текущее значение>:
```

Если вы не хотите изменять текущее значение, показанное в скобках, нажмите на клавишу RETURN. Технические и архитектурные единицы отображаются в футах (') и дюймах (").

После того как выбран формат представления, вы можете выбрать точность отображения чисел. Если вы установили научный, десятичный или технический формат, то запрос имеет следующий вид:

```
Number of digits to right of decimal point (0 to 8) <текущее число знаков>:
```

```
Число знаков после запятой (от 1 до 8) <текущее число знаков>:
```

Следует отметить, что устанавливаемая вами точность - не точность обработки данных Автокадом, а точность отображения данных - в графической базе данных все координаты хранятся с максимально возможной точностью (при вводе значений координат точность отображения также игнорируется). Точность отображения влияет только на вывод значений координат на экран дисплея, а также на проставление измеренных значений размеров (см. предыдущую главу). Для проставления размеров вам придется часто запрещать вывод знаков после запятой (не забывайте по окончании проставления размеров восстанавливать необходимую точность, иначе "целые" числа в строке состояний могут ввести вас в заблуждение, ведь Автокад будет продолжать работать с действительными числами). Далее предлагается выбрать формат и точность представления углов:

```
Systems of angle measure:          (Examples)

1. Decimal degrees                 45.0000
2. Degrees/minutes/seconds         45d0'0"
3. Grads                           50.0000g
4. Radians                         0.7854r
5. Surveyor's units                N 45d0'0" E

Enter choice, 1 to 5 <текущее значение>:
```

```
Система измерения углов:           (Примеры)

1. Градусы в десятичном виде       45.0000
2. Градусы/минуты/секунды          45d0'0"
3. Грады                           50.0000g
4. Радианы                         0.7854r
5. Топографические единицы         N 45d0'0" E

Введите выбор от 1 до 5 <текущее значение>:
```

Для отображения различных угловых мер в Автокаде используются следующие соглашения: десятичные градусы отображаются на экране как десятичные числа без всяких добавок; к градам справа добавляется строчная буква "g", а к радианам - строчная буква "r". Градусы/минуты/секунды отображаются в виде:

```
123 d45 56'1", где
  d  - градусы
  '  - минуты
  "  - секунды
```

После выбора формата представления углов следует указать точность отображения угловых мер:

```
Number of fractional places for display of angle (0 to 8) <текущее значение>:
```

```
Длина дробной части в представлении углов (от 0 до 8)<текущее значение>:
```

При выборе формата представления градусы/минуты/секунды значения углов отображаются на экране в следующем виде:

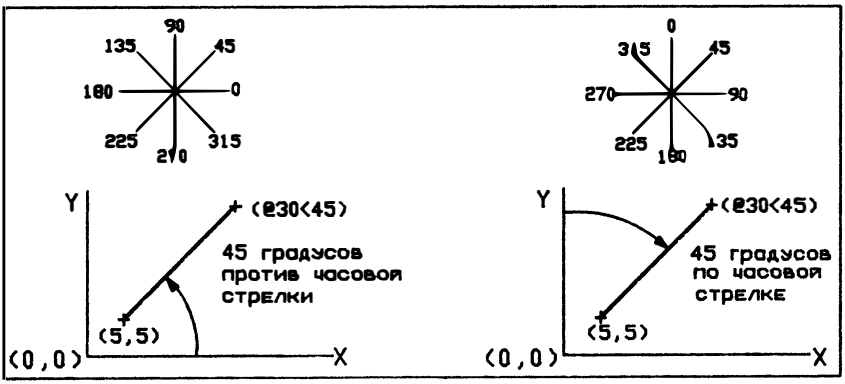
Десятичные разряды	*Что отображается*	*Пример*
0	Только градусы	159d
2	Градусы и минуты	159d10'
3-4	Градусы, минуты и секунды	159d10'12"
5-8	Градусы, минуты и доли секунды (от 1 до 4 десятичных разрядов)	159d10'12.36"

После этого Автокад предоставляет вам возможность объявить нулевое направление отсчета углов и затем положительное направление измерения углов:

```
Direction for angle 0:
     East          3 o'clock  =  0
     North         12 o'clock =  90
     West          9 o'clock  =  180
     South         6 o'clock  =  270
Enter direction for angle 0 <0>: 45
Do you want angles measured clockwise? <N>: Y
```

```
Направление угла 0:
     Восток        3 часа     =  0
     Север         12 часов   =  90
     Запад         9 часов    =  180
     Юг            6 часов    =  270
Введите направление 0 <текущее значение>: 45
Хотите ли вы измерять углы по часовой стрелке? <Н>: Д
```

Нулевое направление отсчета углов всегда устанавливается относительно оси OX текущей системы координат. Ниже приводится пример того, как повлияют установки, произведенные выше, на ввод координат, например, в команде LINE (ОТРЕЗОК):



4.1.2. Лимиты

Хотя чертеж формируется на неограниченном поле, представляется полезным ограничить рабочую зону. Это позволит нам контролировать выход за пределы чертежа в том случае, если мы заранее установили для себя формат и масштаб рисунка. Такие пределы в Автокаде называются *лимитами* и устанавливаются командой LIMITS (ЛИМИТЫ). Лимиты - это прямоугольная область плоскости

XOY мировой системы координат, задаваемая двумя точками - левой нижней и правой верхней вершинами. Ниже перечислены функции, которые в Автокаде выполняют лимиты:

- Если текущей является мировая система координат, то в пределах лимитов отображается вспомогательная координатная сетка. В пользовательских системах координат сетка отображается без соблюдения лимитов
- Если все объекты чертежа находятся в пределах лимитов, то по команде ZOOM All (ПОКАЖИ Все) осуществляется такое масштабирование отображения (вида), при котором в графическую зону экрана будет вписана зона чертежа, ограниченная лимитами
- Пользователь имеет возможность включить режим контроля выхода за лимиты. Если такой контроль включен, то при вводе координат точек за лимитами Автокад выдает сообщение "**Outside limits" ("**Вне лимитов"). Отметим, что контролируется только задание точек; если при построении, например, окружности, она выходит за пределы, сообщение не выдается (при попытке разместить вне лимитов центр этой окружности будет выдано сообщение об ошибке). Если мы работаем в пользовательской системе координат и при указании точки вышли за лимиты, Автокад выдает сообщение "**Outside WORLD limits" ("Вне МИРОВЫХ лимитов")

Формат команды следующий:

```
Command: LIMITS
ON/OFF/Lower left corner <текущее значение>:
Upper right corner <текущее значение>:
```

```
Команда: ЛИМИТЫ
Вкл/Откл/Нижний левый угол <текущее значение>:
Верхний правый угол <текущее значение>:
```

Команда LIMITS (ЛИМИТЫ), как видим, используется для задания самих лимитов, а также для включения (выключения) режима контроля за лимитами.

4.1.3. Управление поименованными объектами

По мере создания чертежа вы добавляете в него так называемые поименованные объекты - блоки, слои, типы линий и т.п. Имена присваиваются таким объектам в момент их создания и заносятся в соответствующий список (вы уже, наверное, представляете, что такое список загруженных в чертеж типов линий или список слоев).

Поскольку в процессе создания чертежа может возникнуть необходимость удалить неиспользуемые блоки (слои) или из-за конфликта имен вы захотите присвоить существующим объектам другие имена, в Автокаде предусмотрены средства работы с поименованными объектами - команды RENAME (НОВОЕИМЯ) и PURGE (УДАЛИ), формат которых очевиден.

Команда RENAME (НОВОЕИМЯ) служит для присвоения поименованному объекту (блок, слой, тип линии, гарнитура, ПСК, вид, конфигурации видовых экранов) нового имени.

Команда PURGE (УДАЛИ) используется для удаления в данный момент неиспользуемых объектов (объектов, на которые ссылки в текущем чертеже могут существовать, но в момент удаления не существуют) - блоков, слоев, типов линий, форм и гарнитур. Неиспользуемый объект - объект, на который не ссылается ни один другой объект чертежа. Например, неиспользуемый блок - это блок, который ни разу не вставлен в чертеж, а неиспользуемый слой - слой, на котором нет ни одного объекта.

Поименованные виды, ПСК и конфигурации видовых экранов не удаляются командой PURGE (УДАЛИ), поскольку на них не ссылаются никакие объекты чертежа. Они удаляются соответствующими опциями "своих" команд.

Удалять объекты можно только в самом начале сеанса редактирования. Команда PURGE (УДАЛИ) удаляет один уровень вложенности за раз. Если, например, удаляется слой, которому назначен свой, единственный в рисунке тип линии, то тип линии не удаляется - нужно после удаления слоя выйти из рисунка, снова вызвать графический редактор и удалить неиспользуемый тип линии. Если вы используете, например, блоки глубокой вложенности, то следует воспользоваться командой PURGE (УДАЛИ) несколько раз.

4.2. Режимы рисования

Задавая точки "мышью" на экране дисплея, вы на самом деле задаете их не на непрерывном координатном поле чертежа, а неявно пользуетесь сеткой, которая предопределена растром дисплея. Так, если вы пользуетесь монитором EGA, то рабочее поле чертежа при одном видовом экране составляет 572x292 пиксела, что в совокупности с условными координатами текущего вида определяет сетку, на которой (и только на которой!) вы задаете точки устройством указания.

Это чисто техническое ограничение вступает в противоречие с огромной точностью (16 знаков после запятой) представления данных в графической базе данных Автокада. Для того чтобы, пользуясь растровым устройством ввода, можно было работать с непрерывным координатным пространством, в Автокаде предусмотрены специальные средства указания точек - режимы рисования:

GRID (СЕТКА)

Формирует на экране точечное поле с заданным расстоянием между точками

SNAP (ШАГ)

Заставляет графический курсор перемещаться дискретно (по шагам), с любым задаваемым интервалом

ORTHO (ОРТО)

Включает режим рисования отрезков и перемещения примитивов только параллельно осям текущей системы координат

OSNAP (ПРИВЯЖИ)

Сокращение от "Object SNAP" ("Объектная привязка") - позволяет автоматически находить характерные точки объектов (конечные точки и середину отрезка, центр дуги, точку касания т .п.)

AXIS (ОСИ)

Отрисовывает вдоль нижней и правой сторон графической зоны экрана оси координатной разметки с заданной ценой деления

Режимы GRID (СЕТКА), ORTHO (ОРТО) и SNAP (ШАГ) могут быть включены прямо в процессе выполнения команд нажатиями клавиш F7, F8 и F9 соответственно. При этом в командной строке появляется соответствующее сообщение, а текущая команда не прерывается.

4.2.1. Сетка

Команда GRID (СЕТКА) выдает изображение координатной сетки с любым требуемым интервалом. Сетка помогает зрительно оценить относительные размеры объектов чертежа. Она не является частью чертежа, а предназначена только для визуальной координации и никогда на печать не выводится. Команда GRID (СЕТКА) вызывается из подменю SETTINGS (НАСТРОЙ) экранного меню:

```
Command: GRID
Grid spacing (x) or ON/OFF/Snap/Aspect <0.00,0.00>: 2
```

```
Команда: СЕТКА
Интервал сетки (x) или Вкл/Откл/Шаг/Аспект <0.00,0.00>: 2
```

Рассмотрим опции команды GRID (СЕТКА):

ON/OFF (Вкл/Откл)

Соответствует включению/выключению сетки. То же самое можно выполнить с помощью клавиши F7 или CTRL G

Grid Spacing (x) (Интервал сетки (x)

Устанавливает размеры ячейки сетки, причем равные по X и Y. Если вы хотите установить размер ячейки сетки кратный шагу, то необходимо к числовому значению, выдаваемому в ответ на запрос команды, дописать латинскую букву "x" (например, 5x или 0.1x означает, что размер ячейки сетки будет в 5 раз больше или в 10 раз меньше, чем размер шага)

Snap (Шаг)

Выбор этой опции обеспечивает сетку, соответствующую шагу привязки

Aspect (Аспект)

Позволяет определить сетку с различным шагом по X и Y. Эта команда включает сетку

Если для сетки задан слишком маленький относительно вида интервал, то система сетку на экране не отображает и выдает следующее сообщение:

```
Grid too dense to display
```

```
Сетка слишком плотна для отображения на мониторе
```

В этом случае для того, чтобы получить изображение сетки, следует задать больший интервал сетки.

4.2.2. Шаговая привязка

Команда SNAP (ШАГ) позволяет привязывать все указания точек к узлам воображаемой сетки с заданным шагом. Здесь можно провести аналогию с миллиметровой бумагой: почти всегда удобнее строить (например, схему) по клеточкам, чем на чистом листе бумаги. Если, например, требуется развести печатную плату, то вы можете воспользоваться командой SNAP (ШАГ) для того, чтобы задавать точки с минимальным интервалом 2,5 мм.

При указании точек "мышью" мы неявно пользуемся сеткой, определяемой условными координатами вида и числом пикселов (элементов изображения), из которых состоит графическая зона экрана. Проверим это:

```
Command: LIMITS
ON/OFF/Lower left corner <текущее значение>: 0,0
Upper right corner <текущее значение>: 572,292
Command: ZOOM
All/Center/Dynamic/Extents/Left/Previous/Window/<Scale(x)>: A
```

```
Команда: ЛИМИТЫ
Вкл/Откл/Нижний левый угол <текущее значение>: 0,0
Верхний правый угол <текущее значение>: 572,292
Команда: ПОКАЖИ
Все/Центр/Динамика/Границы/Левый/Предыдущ/Рамка/<Масштаб(x)>: А
```

Теперь каждому пикселу по горизонтали соответствует одна условная единица и мы не можем "мышью" указать точку с координатами, например (0,130.6). При произвольных координатах графической зоны экрана узлы сетки пикселов вообще попадают на иррациональные числа. Кроме того, прямоугольная область, скажем, из 10x10 пикселов является не квадратом, а прямоугольником с соотношением сторон приблизительно 1.2211 (это число называется "aspect ratio" - "аспект"), в чем вы легко можете убедиться, подвинув курсор из точки (0,0) на один пиксел вверх.

Для того чтобы обойти эти чисто технические ограничения, существует команда SNAP (ШАГ). Она корректирует все задания точек таким образом, чтобы

они попадали в узлы воображаемой сетки с заданным шагом. Другими словами, с помощью команды ШАГ можно задавать шаг перемещения курсора.

Установим шаг равным 1.5, для чего выберем команду SNAP (ШАГ) из экранного меню SETTINGS (НАСТРОЙ):

```
Command: SNAP
SNAP spacing or ON/OFF/Aspect/Rotate/Style <0.000>: 1.5
```

```
Команда: ШАГ
Шаг привязки или Вкл/Откл/Аспект/Поворот/Стиль<0.000>: 1.5
```

Теперь при включенном режиме SNAP (ШАГ) курсор будет перемещаться с шагом 1.5 (как уже говорилось, при помощи клавиши F9 этот режим можно включать и отключать в любой момент).

Подвигайте курсор - он движется по полю чертежа с установленным вами шагом и точно попадает в узлы с координатами кратными установленному вами шагу. Теперь точно задать точку с нужными координатами намного легче. Однако все имеет свои негативные стороны. Теперь для того, чтобы задать точку, которая не попадает в узел сетки, нужно отключать шаг.

Режим шаговой привязки предназначен для облегчения работы с устройством указания и не распространяется на ввод координат точек (как абсолютных, так и относительных) с клавиатуры.

Рассмотрим опции команды SNAP (ШАГ):

Snap spacing (Шаг привязки)

Устанавливает величину дискретного перемещения графического курсора. По умолчанию в угловых скобках показывается последнее значение шага

ON/OFF (Вкл/Откл)

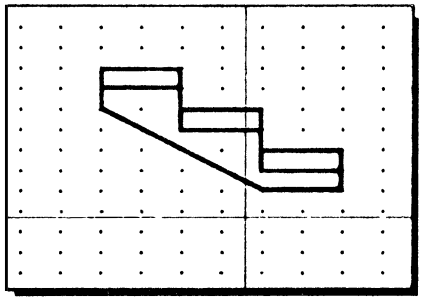
Включение/отключение шага. Аналогичную операцию можно выполнить с помощью клавиш CTRL B или F9

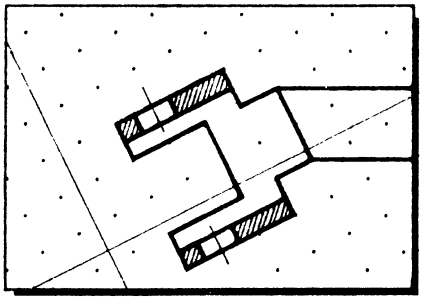
Aspect (Аспект)

Позволяет устанавливать различные интервалы по осям X и Y

Rotate (Поворот)

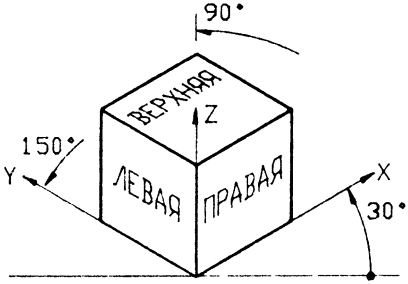
Позволяет поворачивать одновременно сетку и направление шагового перемещения графического курсора по отношению к чертежу и экрану дисплея или же устанавливать для базовой точки сетки фиксации значение, отличное от (0.0). Система запрашивает координаты базовой точки и угол поворота. На рисунке на следующей странице показан результат выполнения подкоманд SNAP Aspect (ШАГ Аспект) и SNAP Rotate (ШАГ Поворот):

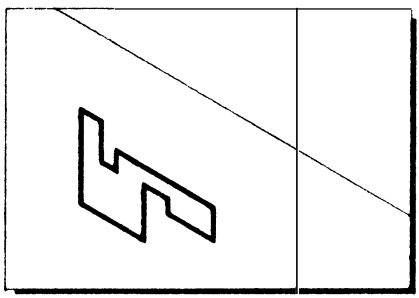




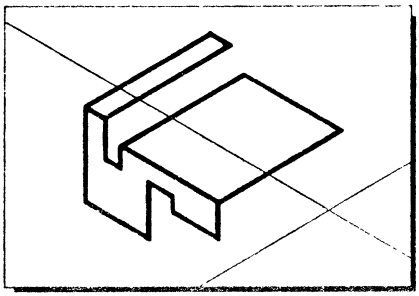
Style (Стиль)

Позволяет выбрать формат или тип сетки фиксации: "стандартный" и "изометрический". *Стандартный* соответствует сетке нормального прямоугольного типа (интервалы по X и Y могут различаться). *Изометрический* - сетке для построения изометрических проекций, в которых точки организованы так, чтобы упростить проведение линий под углами 30, 90, 150, 210, 270 и 330 градусов. Для такой сетки система запрашивает вертикальный интервал между точками. При использовании изометрического стиля режима ШАГ рисуем на левой, правой или верхней изометрической плоскости:

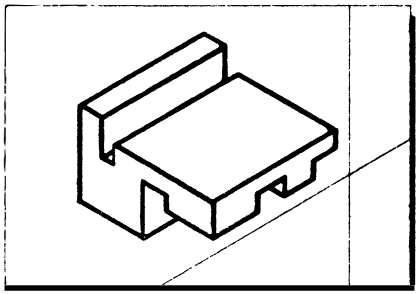




Включена Левая плоскость



Включена Верхняя плоскость



Включена Правая плоскость

В момент первого включения изометрического режима шага текущей изометрической плоскостью будет левая. Выбором необходимой плоскости изометрии и тем самым текущей пары осей управляет команда ISOPLANE (ИЗОМЕТР):

```
Command: ISOPLANE
Left/Top/Right/<Toggle>
```

```
Команда: ИЗОМЕТР
Левая/Верхняя/Правая/<Переключатель>:
```

Гораздо удобнее, однако, пользоваться клавишами CTRL E, которые позволяют переключать плоскости изометрии прямо в процессе выполнения других команд.

4.2.3. Орто

ORTHO (ОРТО) - специальный режим, в котором строить и перемещать изображение можно только параллельно линиям графического курсора. Этот режим можно уподобить использованию головки кульмана в традиционном черчении (линии графического курсора выступают как линейки головки). При этом задание угла поворота шага или ПСК равносильно повороту головки кульмана.

Режим ORTHO (ОРТО) управляется одноименной командой, однако гораздо удобнее пользоваться клавишей F8

4.2.4. Объектная привязка

Процесс конструирования неотделим от точных геометрических построений, в которых требуется восставлять перпендикуляры, проводить касательные, находить конечные точки и середины отрезков и дуг и т.п. Очевидно, что подобные задачи нельзя решить простым указанием точек на рабочем поле чертежа. Для этого в Автокаде существует специальное средство - *объектная привязка*.

Подобно тому как шаговая привязка позволяет находить точки, строго лежащие на координатной сетке, объектная привязка позволяет находить характерные точки примитивов.

Режимы объектной привязки устанавливаются командой OSNAP (ПРИВЯЖИ), имеющей большое количество опций:

CENter (ЦЕНтр)

Нахождение центра дуги или окружности

ENDpoint (КОНточка)

Нахождение конечной точки дуги, отрезка или сегмента полилинии

INSert (ТВСтавки)

Нахождение точки вставки блока или текстовой строки

INTersec (ПЕРесечение)

Нахождение точки пересечения двух объектов

MIDpoint (СЕРедина)
Нахождение середины дуги или отрезка

NEArest (БЛИжайшая)
Нахождение ближайшей точки, принадлежащей указанному объекту

NODe (УЗЕл)
Нахождение координат отдельной точки

PERpendicular (НОРмаль)
Нахождение принадлежащей объекту точки, лежащей на перпендикуляре, восставленном из предыдущей указанной точки. При восставлении перпендикуляра режим слежения не работает

TANgent (КАСательная)
Нахождение точки касания указанной дуги или окружности с отрезком, проведенным из предыдущей указанной точки. Предыдущая точка также может быть указана с режимом КАСательная, что позволяет, например, проводить касательные к двум окружностям и т.п.

QUAdrant (КВАдрант)
Нахождение точек, лежащих на пересечении указанной дуги или окружности с воображаемыми линиями, параллельными осям текущей ПСК и проходящими через центр дуги или окружности

QUICK (БЫСтрая)
Этот режим является вспомогательным и служит для ускорения работы остальных операций захвата за счет быстрого нахождения первой же точки, удовлетворяющей условию захвата

NONE (НИЧего)
Отключение объектной привязки

Объектная привязка - это специальный режим задания точек, поэтому действует только тогда, когда требуется указать точку на экране. При включенном режиме объектной привязки к перекрестью графического курсора добавляется квадратик, называемый *прицелом объектной привязки.* Для того чтобы найти, допустим, конечную точку отрезка, следует указать квадратиком на отрезок и нажать на левую кнопку "мыши". Перекрестье курсора при этом не обязательно должно находиться вблизи предполагаемой точки - достаточно, чтобы отрезок пересекал прицел объектной привязки (Автокад найдет нужную точку на указанном объекте автоматически).

Существует два режима объектной привязки - постоянный и одноразовый. Постоянный режим устанавливается командой OSNAP (ПРИВЯЖИ), которая имеет следующий формат:

```
Command: OSNAP
Object snap modes: CEN,NEA
```

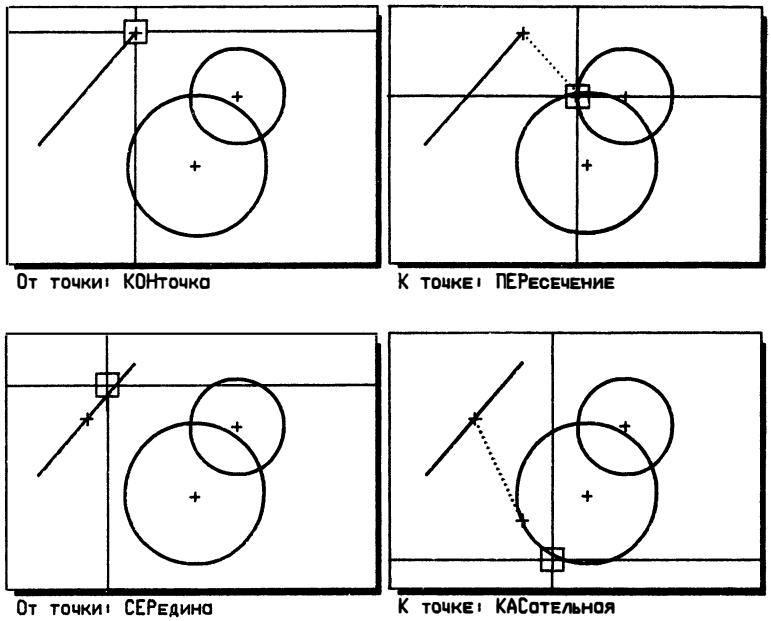
Команда: **ПРИВЯЖИ**
Режимы объектной привязки: **ЦЕН, БЛИ**

В ответ на запрос нужно указать одну или несколько опций объектной привязки, перечисленных выше. После этого в ответ на все запросы, требующие указания точек, вы должны указывать объекты, а нужные точки Автокад найдет на этих объектах автоматически по установленным вами режимам привязки.

Почти всегда на указанном объекте находится несколько точек, удовлетворяющих условиям привязки - например, у отрезка две конечные точки и одна середина, к окружности можно провести две касательных и восставить два перпендикуляра и т.д. В этом случае Автокад возвратит ту точку, которая находится ближе всего к месту указания на объекте (если, например, вы установили режимы привязки CENter, NEArest (ЦЕНтр, БЛИжайшая), то не сможете привязаться к центру окружности или дуги).

Для отмены постоянного режима объектной привязки следует снова вызвать команду OSNAP (ПРИВЯЖИ) и ответить NONE (НИЧего), OFF (ОТКлючи) или просто нажать на клавишу RETURN.

Рисунок ниже иллюстрирует применение объектной привязки:



Если рисунок насыщен и в прицел попадает много объектов, то требуется время для отыскания ближайшей к перекрестью курсора точки в списке точек,

удовлетворяющих условиям привязки. При помощи опции QUICK (БЫСтрая) можно ускорить этот процесс - если задана быстрая привязка, то берется первая попавшаяся точка, удовлетворяющая одному из условий привязки.

При постоянном режиме привязка будет действовать на все указания точек до отмены режима привязки. Постоянная привязка удобна тогда, когда постоянно идет работа с точками одного типа (например, при обводке многоугольной фигуры можно все время пользоваться точками пересечения). Чаще, однако, в одной и той же команде нужно привязываться к разным точкам. Для этого предусмотрен одноразовый режим привязки, при котором в ответ на запрос о вводе координат точек указывается сначала режим привязки, а затем объект:

```
Command: LINE
From point: CEN <указывается окружность>
To point:
```

```
Команда: ОТРЕЗОК
От точки: ЦЕН <указывается окружность>
К точке:
```

В этом примере проводится отрезок с началом в центре окружности. Привязка CENter (ЦЕНтр) действительна только на одно указание точки. Если при постоянном режиме привязки Автокад, не находя точку привязки, возвращает точку, в которой вы указывали объект, то при одноразовой привязке выдается сообщение об ошибке и возобновляется предыдущий запрос:

```
Command: LINE
From point: CEN <указывается окружность>
No Centerpoint found for specified point.
Point or option keyword required.
From point: CEN <еще раз указывается окружность>
To point:
```

```
Команда: ОТРЕЗОК
От точки: ЦЕН <указывается окружность>
Режим КОНточка не определен для заданной точки.
Требуется 2-мерная точка или ключевое слово
От точки: ЦЕН <еще раз указывается окружность>
К точке:
```

При работе с одноразовой привязкой опции объектной привязки удобно выбирать из падающего меню Tools (Средства) или из корня экранного меню, пункт ****.

В Автокаде версии 10 объектная привязка значительно усовершенствована по сравнению с более ранними версиями. Теперь объектная привязка работает в трехмерном пространстве и, в частности, может привязываться к конечным точкам (или середине любого края) двумерных объектов, имеющих высоту.

Здесь есть, однако, одна тонкость. Дело в том, что для совместимости с более ранними версиями Автокада, не работавшими с трехмерной графикой, введена переменная FLATLAND (букв. "плоская поверхность"), которая отключает некоторые новые возможности, появившиеся в десятой версии. Так, если переменная FLATLAND имеет значение 1 (это значение для совместимости устанавливается по умолчанию), то объектная привязка в трехмерном пространстве работает только с трехмерными примитивами.

Установка размера прицела объектной привязки

Размер прицела объектной привязки можно увеличить или уменьшить командой APERTUR: (АПЕРТУР:), которая находится в подменю APERTUR: (АПЕРТУР:) подменю SETTINGS (НАСТРОЙ) экранного меню:

```
Command: APERTURE
Object snap target height (1-50 pixels) <10>: 4
```

```
Команда: АПЕРТУРА
Размер прицела объектной привязки (1-50 точек) <10>: 4
```

Отметим попутно, что в том же меню находится пункт установки прицела выбора объектов.

4.2.5. Оси координат

Эта команда обеспечивает отрисовку вдоль нижней и правой сторон графической зоны экрана разметки осей координат. Отображение осей возможно только при работе с одним видовым экраном и планом мировой системы координат. Формат этой команды аналогичен формату команды GRID (СЕТКА):

```
Command: AXIS
Tick spacing (x) On/Off/Snap/Aspect<0.0000>:
```

```
Команда: ОСИ
Цена деления (x) или Вкл/Откл/Шаг/Аспект<0.0000>:
```

Если задана слишком маленькая цена деления, то засечки очень плотны и разметка осей не может быть отображена. В этом случае система выдаст следующее сообщение:

```
Axis ticks too close to display.
```

```
Разметка осей слишком плотна для отображения на экране.
```

4.2.6. Пользовательские системы координат

В главе 1 мы давали определение *пользовательской системы координат*, ПСК (User Coordinate System, UCS). В этом пункте мы рассмотрим команду UCS (ПСК), управляющую созданием пользовательских систем координат:

```
Command: UCS
Origin/ZAxis/3point/Entity/View/X/Y/Z/Prev/Restore/
Save/Del/?/<WCS>:
```

```
Команда: ПСК
Начало/ZOсь/3точки/Объект/Вид/X/Y/Z/Предыдущ/Замени/
Сохрани/Удали/?<МСК>:
```

Команда UCS (ПСК) предоставляет большое количество способов определения пользовательских систем координат, а также выполняет ряд вспомогательных функций. К сожалению, из-за недостатка места мы не в состоянии привести поясняющие примеры и описать технологию работы с пользовательскими системами координат, а вынуждены только кратко описать эти возможности:

Origin (Начало)

Новая ПСК получается из текущей ПСК плоскопараллельным переносом на вектор, началом которого является начало текущей ПСК, а концом - задаваемая пользователем трехмерная точка

ZAxis (ZOсь)

Расширение предыдущей опции. После задания нового начала координат требуется задать точку, которая будет лежать на положительной части оси Z (тем самым определяется направление оси Z), т.е. неявно задать два угла поворота новой ПСК относительно текущей

3point (3точки)

Вариант предыдущей опции. После задания нового начала координат требуется задать точку, определяющую направление оси OX, а затем заданием еще одной точки определить направление оси OY (при этом третья точка используется только для определения угла поворота новой ПСК относительно уже заданной оси OX). Положение оси Z определяется по правилу правой руки

Entity (Объект)

Определяет новую ПСК по указываемому объекту (объект, однако, не должен быть трехмерной полилинией или сетью). Плоскость OXY совмещается с плоскостью, определяемой объектом, причем в качестве начала координат берется первая характерная точка примитива (центр дуги, начальная точка полилинии и т.п.), а в качестве точки, определяющей ось OX берется вторая характерная точка примитива (вторая точка полилинии или вторая точка трехмерной грани и т.п.). Если объект не определяет плоскость (точка или отрезок), то новая ПСК берется параллельной текущей ПСК (для отрезка ось OX определяется проекцией второй точки на текущую ПСК)

View (Вид)
Новая ПСК делается параллельной виду путем соответствующего поворота вокруг начала текущей ПСК

X/Y/Z (X/Y/Z)
Новая ПСК получается из текущей путем поворота относительно одной из ее осей на указываемый угол

Prev (Предыдущ)
Восстанавливает как текущую предыдущую ПСК. Автокад хранит десять последних ПСК, восстанавливая при этом даже непоименованные ПСК

Restore (Замени)
Позволяет переименовать поименованную ПСК, делая ее затем текущей

Save (Сохрани)
Позволяет присвоить имя текущей ПСК, после чего определение текущей ПСК заносится в список определенных в рисунке систем координат

Del (Удали)
Позволяет удалить ПСК из списка определенных в рисунке систем координат. За один раз можно удалить несколько ПСК, указав их имена через запятую или используя символы поиска ДОС "*" и "?"

? Выводит на экран список имен, определенных в рисунке систем координат, с полным их определением (имя, координаты начала, направления осей). Если текущая ПСК не поименована, она выступает под фиктивным именем *NONAME*

WCS (МСК)
Устанавливает мировую систему координат текущей

При работе с системами координат очень удобно также пользоваться диалоговым окном систем координат (команда DDUCS (ДИАЛПСК) или пункт UCS Dialogue... (ПСК Диалог...) падающего меню Settings (Режимы).

За тем, какова в данный момент ориентация осей текущей ПСК, легко следить по пиктограмме системы координат, которая перерисовывается при изменении текущей ПСК.

Вид и текущая ПСК могут быть связаны и не связаны друг с другом - этим управляет системная переменная UCSFOLLOW (букв. "слежение за ПСК"). Если UCSFOLLOW имеет значение 1, то при определении новой ПСК автоматически будет показан вид в плане новой ПСК, а если UCSFOLLOW имеет значение 0, то вид при определении новой ПСК не изменится, и для того, чтобы перейти в план текущей ПСК, нужно будет воспользоваться специальной командой, PLAN (ПЛАН):

```
Command: PLAN
<Current UCS>/UCS/World:
```

```
Команда: ПЛАН
<Текущая ПСК>/ПСК/МИР:
```

Команду PLAN (ПЛАН) можно выбрать также из экранного меню DISPLAY (ПОКАЖИ), пункт PLAN: (ПЛАН:), в котором находится также пункт установки переменной UCSFOLLOW - Follow: (Следующ:). Опции команды PLAN (ПЛАН) помещены также в падающее меню Display (Дисплей), пункты Plan View (UCS) (Вид в плане (ПСК) и Plan View (World) (Вид в плане (Мир).

4.3. Управление изображением

В главе 3 мы уже рассказывали о том, как Автокад строит изображение на экране. Напомним, что изображение строится Автокадом в два этапа. Сначала в оперативной памяти ЭВМ на т.н. *виртуальном экране* (области памяти, моделирующей "дисплей" с разрешением 32768 x 32768 пикселов) создается оригинал изображения (именно этот этап, называемый *регенерацией*, наиболее долог), а затем изображение с виртуального экрана переносится на дисплей (происходит *перерисовка* изображения). Перечислим команды, которые управляют режимами перерисовки и регенерации:

REDRAW (ОСВЕЖИ)

Выполняет перерисовку экрана - очищает экран от маркеров и от графического "мусора", остающегося после выполнения команд редактирования

REGEN (РЕГЕН)

Регенерирует изображение (полностью перестраивает изображение по геометрическому описанию чертежа). После регенерации всегда автоматически выполняется перерисовка изображения на дисплее

REGENAUTO (РЕГЕНАВТО)

Включает и выключает режим автоматической регенерации изображения

4.3.1. Перерисовка и регенерация

В процессе построения и редактирования изображения графический экран "засоряется" - накапливаются маркеры (если включены), при стирании объектов исчезают совмещенные с ними части других объектов - изображение на дисплее постепенно перестает соответствовать чертежу. Для того чтобы убрать графический "мусор", нужно произвести очистку экрана. что достигается перенесением изображения с виртуального экрана на дисплей.

Восстановление изображения (перерисовка) осуществляется при помощи команды REDRAW (ОСВЕЖИ):

```
Command: REDRAW
```

```
Команда: ОСВЕЖИ
```

Команду можно выполнить набором с клавиатуры или выбором из падающего меню Display (Дисплей), но самый простой способ - включить, а затем выключить сетку, дважды нажав на клавишу F7. При переопределении гарнитуры шрифта, изменении режима закрашивания, значительном изменении вида и некоторых других операциях требуется перестроить изображение на виртуальном экране. В режиме автоматической регенерации (команда REGENAUTO (РЕГЕНАВТО) изображение на виртуальном экране перестраивается автоматически. В противном случае будет запрашиваться подтверждение регенерации:

```
Regenerate drawing? <Y>:
```

```
Регенерировать рисунок? <Д>:
```

Если следует отрицательный ответ, выполнение команды прерывается.

Регенерацию можно выполнить явно при помощи команды REGEN (РЕГЕН).

4.3.2. Изменение вида

В первой главе мы назвали видом любую часть изображения, выводимую на экран дисплея. В Автокаде существуют достаточно удобные средства управления видами - панорамирование, увеличение и уменьшение изображения, а также сохранение нужных видов.

PAN (ПАН)

Эта команда, название которой происходит от слова "панорамирование", позволяет, не изменяя масштаба отображения чертежа на экране, перемещать его относительно экрана на заданный вектор

ZOOM (ПОКАЖИ)

Эта команда позволяет управлять размером вида на экране (уменьшать и увеличивать масштаб изображения)

VIEW (ВИД)

Команда ВИД позволяет сохранять виды, присваивая им имена, а также вызывает ранее созданные виды на экран

Плоскопараллельное перемещение поля чертежа относительно графической зоны экрана осуществляется командой PAN (ПАН). Направление и величина перемещения задается парой точек, определяющих вектор:

```
Command: PAN
Displacement:
Second point:
```

```
Команда: ПАН
Смещение:
Вторая точка:
```

Изменение масштаба отображения в графической зоне экрана осуществляется командой ZOOM (ПОКАЖИ):

```
Command: ZOOM
All/Center/Dynamic/Extents/Left/Previous/Window/<Scale(x)>:
```

```
Команда: ПОКАЖИ
Все/Центр/Динамика/Границы/Левый/Предыдущий/Рамка/<Масштаб
(x)>:
```

Рассмотрим опции команды ZOOM (ПОКАЖИ):

Scale (Масштаб)
Позволяет задать числовой коэффициент изменения масштаба изображения

All (Все)
Если объекты выходят за лимиты чертежа, будут показаны экстенты; если весь чертеж находится в лимитах, будут показаны лимиты

Center (Центр)
Позволяет задать центр будущего вида и высоту графической зоны в условных единицах чертежа

Extents (Экстенты)
Показывает все объекты чертежа

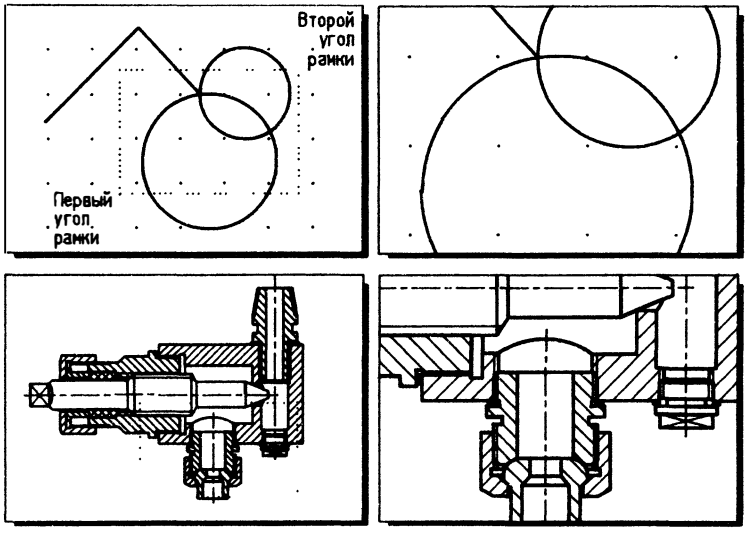
Left (Левый)
Позволяет задать левый угол будущего вида и высоту графической зоны в условных единицах чертежа

Previous (Предыдущий)
Восстанавливает предыдущий вид

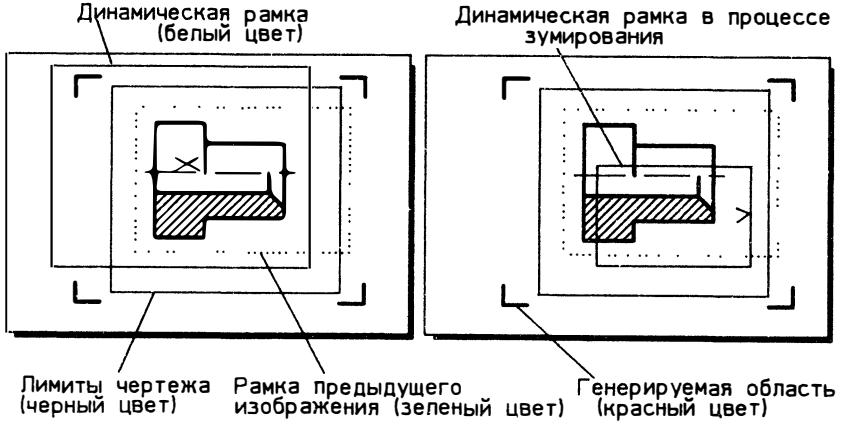
Window (Рамка)
Позволяет задать будущий вид г ем определения прямоугольной зоны, которая будет вписана в графическую зону экрана

В простейшем варианте команда ZOOM (ПОКАЖИ) позволяет определить новый вид заданием коэффициента изменения изображения относительно первого вида в сеансе редактирования. Коэффициент изменения 1 не изменяет вид; числа от 0 до 1 приводят к уменьшению изображения, а числа большие 1 приводят к увеличению изображения. Если вводится число, за которым следует латинская буква "x", то вид масштабируется относительно текущего вида (например, ввод "0.5x" приводит к уменьшению изображения объекта в два раза).

Пользуясь режимом Window (Рамка), при необходимости можно увеличить часть изображения, приблизительно задав рамкой интересующую нас область чертежа. Изображение увеличится так, чтобы указанная рамкой область была вписана в графическую зону экрана. Пример выполнения команды ZOOM Window (ПОКАЖИ Рамка) показан ниже:



Автокад хранит полный протокол работы с момента начала сеанса редактирования (этот вопрос подробно рассматривается в следующей главе, п. 5.10) и может восстановить последовательно по одному все виды, которыми вы пользовались в текущем сеансе. Вернуться к предыдущему виду можно посредством команды ZOOM Previous (ПОКАЖИ Предыдущий), а многократное повторение этой команды приведет к возврату к первоначальному виду Дополнительные возможности управления видами предоставляются подкомандой ZOOM Dynamic (ПОКАЖИ Динамика), при вызове которой на экране дисплея отображаются четыре зоны чертежа:



Попытаемся понять, как следует управлять координатами и размерами окна динамического увеличения.

При перемещении устройства указания область, обозначающая будущий вид, передвигается по полю чертежа без изменения размеров. Нажмите левую клавишу "мыши". У левого края динамического окна появится стрелка, которая сообщает нам о том, что левый край зафиксирован. Теперь перемещение устройства указания приводит к изменению размеров будущего вида. Еще раз нажав на левую клавишу "мыши", мы опять зафиксируем размер окна и получим возможность перемещать его по полю чертежа и т.п. Нажав на правую клавишу "мыши", вы зафиксируете выбранный вид.

При выходе графического курсора за зону виртуального экрана в левом нижнем углу графической зоны появляется пиктограмма песочных часов. Таким способом Автокад сообщает вам, что при выборе этого вида потребуется регенерация изображения.

Настройка нужного вида занимает немного времени, однако если требуется чертить в двух местах параллельно и при этом еще пользоваться каким-то общим видом, то применять последовательностью команд ZOOM All - ZOOM Window (ПОКАЖИ Все - ПОКАЖИ Рамка) не очень удобно. В Автокаде предусмотрена возможность сохранения часто используемых видов - для этого следует пользоваться командой VIEW (ВИД):

```
Command: VIEW
?/Delete/Restore/Save/Window: S
VIEW name to save:
```

```
Команда: ВИД
?/Удали/Восстанови/Сохрани/Рамка: С
Имя вида:
```

Перечислим опции команды VIEW (ВИД):

? Позволяет просмотреть список поименованных видов чертежа

Save (Сохрани)
Присваивает имя текущему виду, что позволяет сохранить вид

Restore (Восстанови)
Вызывает поименованный вид на экран

Delete (Удали)
Удаляет поименованный вид из чертежа

Window (Рамка)
Сохраняет задаваемую рамкой прямоугольную область экрана как поименованный вид

Кроме того, с помощью команды RENAME View (НОВОЕИМЯ Вид) вы можете переименовать определенный вами ранее вид

В заключение отметим, что команды ПАН, ПОКАЖИ и ВИД, а также ОСВЕЖИ можно вызывать в процессе выполнения других команд (напомним, что такой режим выполнения команд называется "прозрачным" режимом).

4.3.3. Режим быстрого зумирования

Чтобы объяснить, что такое режим "быстрого зумирования", рассмотрим еще раз процесс создания изображения на дисплее. Как уже говорилось, для упрощения и ускорения работы Автокад использует виртуальный экран, который является промежуточным звеном между геометрическим описанием чертежа и изображением на экране. Графическая информация хранится на виртуальном экране в специальном виде: во-первых, только в целых числах (это необходимо для быстрого перенесения изображения на дисплей), а во-вторых, в виде координат векторов, которыми аппроксимируются также все дуги и кривые (такое представление графической информации облегчает и ускоряет построение изображения по его геометрическому описанию).

Чем больше векторов на виртуальном экране должно быть создано при регенерации, тем дольше генерируется изображение. Нетрудно видеть, что больше всего векторов требуется для аппроксимации дуг и окружностей. Хотя отрисованная тридцатью векторами окружность практически неотличима на экране от плавной кривой, как правило на виртуальном экране окружность аппроксимируется несколько большим числом сегментов (стандартное значение - 100). Это сделано для того, чтобы при увеличении масштаба отображения не нужно было проводить регенерацию - по возможности вид перерисовывается, т.е. изображение с виртуального экрана без регенерации переносится на дисплей. Такой "избегающий регенерации" режим называется режимом *быстрого зумирования*, так как он значительно ускоряет изменение вида при увеличении изображения.

Режим быстрого зумирования может быть отключен - при этом команды ZOOM (ПОКАЖИ), PAN (ПАН) и VIEW Restore (ВИД Восстанови) будут всегда регенерировать экран.

Команда VIEWRES (НАСТРВИД) позволяет включать и выключать режим быстрого зумирования, а также позволяет управлять точностью аппроксимации дуг и окружностей на виртуальном экране. Попробуйте увеличить число векторов, которыми будут отрисовываться окружности:

```
Command: VIEWRES
Do you want fast zooms?<Y>:
Enter circle zoom percent (1-20000)<100>: 1000
Regenerating drawing.
```

```
Команда: НАСТРВИД
Нужно ли вам быстрое зумирование? <Д>.
Введите точность аппроксимации (1-20000)<100>: 1000
Регенерирую рисунок.
```

Теперь регенерация будет происходить дольше, но зато увеличение изображения во многих случаях быстрее (если вы часто работаете с командой VIEW (ВИД), то это даст определенный выигрыш во времени).

4.3.4. Виды в трехмерном пространстве

Автокад предоставляет богатые возможности работы в трехмерном пространстве - выбор точки зрения с отработкой двух типов проекций (параллельная и перспективная), выполнение сечений в трехмерном пространстве, а также удаление скрытых линий. Команды VPOINT (ТЗРЕНИЯ) и DVIEW (ДВИД) позволяют "рассматривать" трехмерные объекты с разных сторон, при этом команда VPOINT (ТЗРЕНИЯ) позволяет определять точку зрения в трехмерном пространстве с параллельной проекцией, а команда DVIEW (ДВИД), в основном предназначена для построения перспективной проекции и выполнения сечений. Для удаления скрытых линий в трехмерном пространстве используется команда HIDE (СКРОЙ).

Поскольку в рамки настоящего руководства не входит подробное рассмотрение трехмерной графики Автокада, в настоящем пункте мы рассмотрим только команду VPOINT (ТЗРЕНИЯ):

```
Command: VPOINT
*** Switching to the WCS ***
Rotate/<Viewpoint> <текущая>: 1,1,1

Regenerating drawing

*** Returning to the UCS ***
```

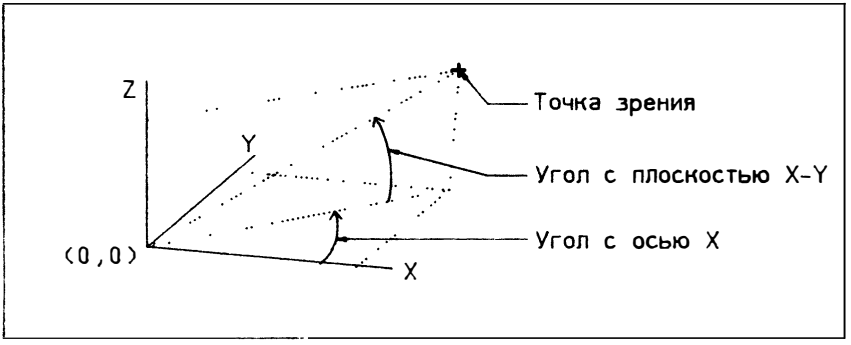
```
Команда: ТЗРЕНИЯ
*** Переключение в МСК ***
Поверни/<Точка зрения> <текущая>: 1,1,1

Регенерирую рисунок.

*** Возврат в ПСК ***
```

Эта команда позволяет задать точку зрения в трехмерном пространстве, по которой строится параллельная проекция. Точку зрения можно задать как явным заданием координат точки (режим по умолчанию), так и в сферической системе координат (опция Rotate (Поверни). Поскольку точка зрения задает фактически только направление проецирования трехмерных объектов, при задании сферических координат радиус указывать не нужно:

```
Enter angle in X-Y plane from X axis <текущий>. 45
Enter angle from X-Y plane <текущий> 45
```

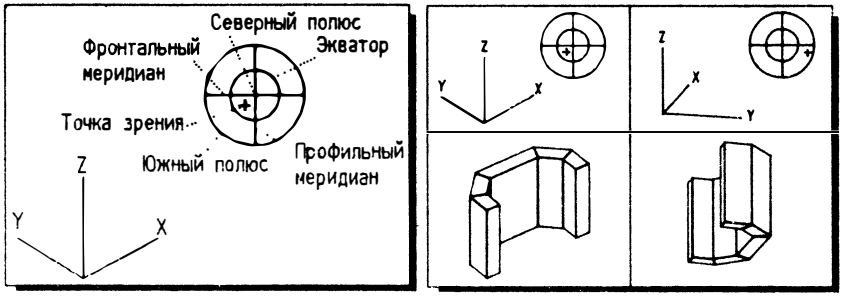
```
Введите угол в плоскости X-Y относительно оси X <текущий>:45
Введите угол с плоскостью X-Y <текущий>:45
```

На рисунке показан пример задания углов и координат для точки зрения:



Точка зрения и углы для команды VPOINT (ТЗРЕНИЯ) по умолчанию задаются в мировых координатах, при этом на время выполнения команды VPOINT (ТЗРЕНИЯ) Автокад переключается в МСК, выдавая соответствующее сообщение. Чтобы задавать точку зрения и углы в текущей системе координат, следует системной переменной WORLDVIEW присвоить значение 0 (по умолчанию - 1).

Если, не вводя координат точки зрения и не выбирая опцию Rotate (Поверни), нажать на клавишу RETURN, то будет предоставлен еще один способ задания точки зрения - по развертке сферы на плоскости:



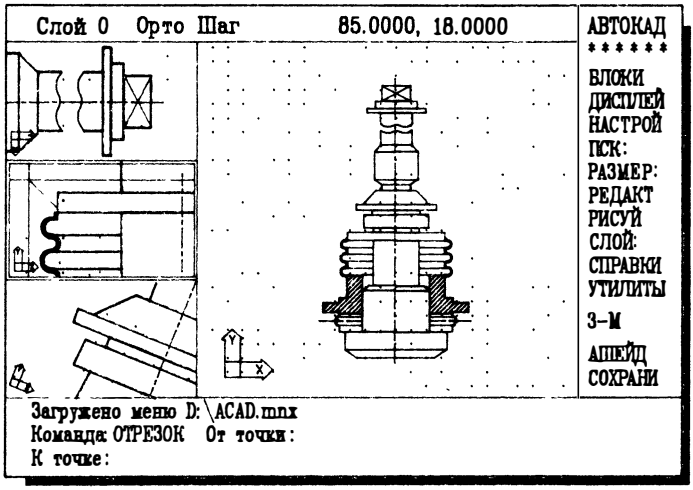
В падающем меню Display (Дисплей) имеется графическое меню команды VPOINT (ТЗРЕНИЯ) - пункт Vpoint 3D... (Точка зрения), в котором используется сферическая система координат и предлагается выбрать направление взгляда, после чего вы должны с клавиатуры задать угол с плоскостью XOY.

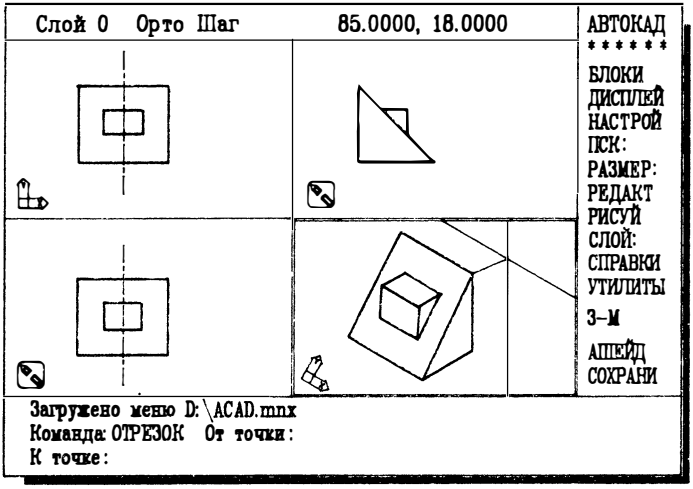
4.3.5. Видовые экраны

Автокад позволяет одновременно работать с несколькими видами, отображая каждый в своем *видовом экране*. Видовые экраны очень удобно использовать:

- для показа крупным планом отдельных участков чертежа
- для разномасштабных изображений (например, в схемах и картах)
- для трехмерной графики

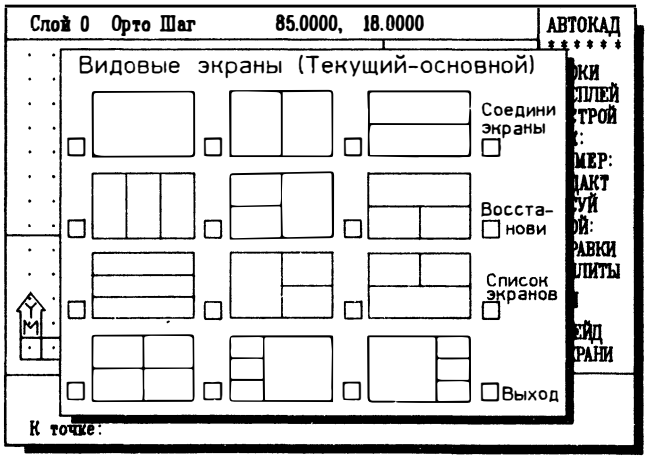
Использование видовых экранов иллюстрируется ниже:





Управление видовыми экранами Автокада осуществляется при помощи команды VPORTS (ВЭКРАН), которая позволяет создавать различные конфигурации видовых экранов, присваивать текущей конфигурации видовых экранов имя (при этом она заносится в список определенных в рисунке конфигураций видовых экранов), восстанавливать ранее поименованную конфигурацию видовых экранов и удалять конфигурацию видовых экранов из чертежа.

При работе с видовыми экранами удобно пользоваться не самой командой VPORTS (ВЭКРАН), а графическим меню Display (Дисплей), пункт Set Viewports... (Видовые экраны...), где приводятся двенадцать наиболее употребимых конфигураций видовых экранов:



Тот экран, на котором происходит отрисовка объектов, называется текущим Текущий видовой экран выделяется широкой рамкой; курсор за пределами текущего экрана имеет вид стрелки, а не перекрестия. Переключаться с одного экрана на другой можно прямо в процессе выполнения команд, что позволяет, например, начальную точку отрезка указать на одном виде, а конечную - на другом (более крупном плане или другой проекции). Чтобы сделать видовой экран текущим, следует, поместив в него курсор, нажать левую клавишу "мыши"

4.4. Получение справок

Система Автокад может в любой момент предоставить дополнительную справочную информацию.

- о наборе команд Автокада
- о каждой команде в отдельности - т.н. "контекстная справка"
- о геометрических характеристиках объектов: координатах точек, углах и т.п.
- о площади, периметре и т.п. вспомогательной вычисляемой информации

Справки о командах Автокада могут быть получены по команде HELP (ПОМОЩЬ), которая (единственная из всех команд) имеет также второе имя - знак "?":

```
Command:?
Command name (RETURN for list):
```

```
Команда:?
Имя команды (RETURN для выдачи списка команд):
```

Если вы нажмете клавишу RETURN, то Автокад выведет на экран список всех команд. Набрав имя команды, вы получите справку по отдельной команде.

В Автокаде предусмотрена контекстная помощь, т.е. справку о команде можно получить в процессе выполнения этой команды. Для этого в ответ на любой запрос (кроме запроса на ввод текста) нужно выполнить прозрачную команду HELP (ПОМОЩЬ):

```
Command: LINE
From point: '?
Resuming LINE command.
From point:
```

```
Команда: ОТРЕЗОК
От точки: '?
Возобновляю команду ОТРЕЗОК.
От точки:
```

4.4.1. Модификация файла справок

Текст, выводимый на экран по команде HELP (ПОМОЩЬ), находится в файле acad.hlp. Это обычный текстовый файл, который может быть отредактирован с помощью любого текстового редактора, работающего в режиме "программист" (не добавляющего в текстовый файл своих служебных символов).

Файл помощи организован в виде разделов, которые выделяются обратной косой чертой - "\", после которой следует имя команды, по которой должна быть выдана помощь. Страницы экрана в одном разделе также отделяются друг от друга знаками обратной косой черты. Для одного и того же раздела можно указывать несколько имен, для чего альтернативные имена следует также выделять обратной косой чертой и располагать на соседних строчках.

При вызове команды HELP (ПОМОЩЬ) Автокад пользуется индексным файлом acad.hdx, в котором перечислены все команды и смещение соответствующей страницы в байтах от начала файла помощи. Если вы добавили в этот файл свою справочную информацию, то вам необходимо удалить индексный файл acad.hdx. Тогда при первом же вызове команды ПОМОЩЬ в следующем сеансе редактирования Автокад автоматически построит новый индексный файл помощи.

4.4.2. Получение информации о чертеже

Графическая база данных Автокада открыта для вас - вы можете получить практически любую необходимую информацию. Кроме того, существуют специальные команды, по которым Автокад производит вспомогательные вычисления. Все справочные команды расположены в подменю INQUIRY (СПРАВКИ) экранного меню:

AREA (ПЛОЩАДЬ)

Позволяет вычислить площадь, ограниченную либо точками. указываемыми пользователем, либо примитивом. Площади можно складывать и вычитать средствами этой же команды

DBLIST (БДСПИСОК)

Обеспечивает вывод информации о всех примитивах в рисунке (эквивалентно указанию всех примитивов в команде LIST (СПИСОК)

DIST (ДИСТАНЦ)

Обеспечивает измерение длины и угла наклона в текущей ПСК вектора, задаваемого двумя точками

ID (КООРД)

Позволяет, указав точку на экране, получить ее координаты

LIST (СПИСОК)

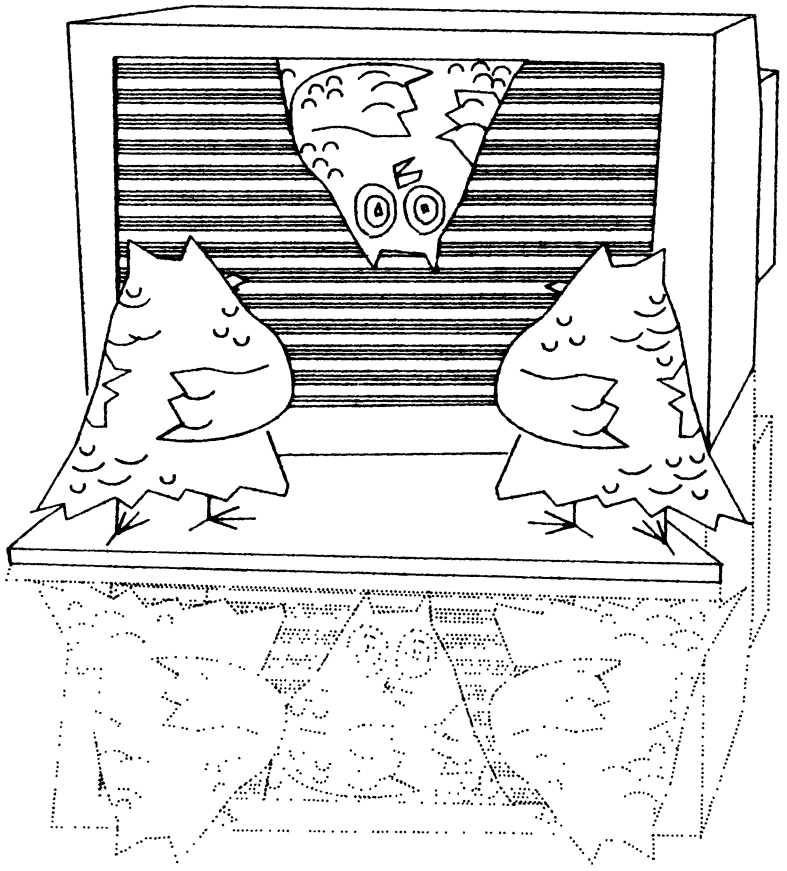
Выводит на экран из графической базы данных Автокада основные (не все!) данные по указанному примитиву (или группе примитивов). Общие свойства примитива (слой, цвет и т.п.) выводятся всегда. Иногда выводится дополнительная информация

STATUS (СТАТУС)

Выдает на экран основные умолчания - лимиты, состояние шага, сетки и т.п.

TIME (ВРЕМЯ)

Позволяет вести регистрацию времени работы с чертежом, времени работы в данном сеансе редактирования и др.



ГЛАВА 5

ПРОСТОЕ РЕДАКТИРОВАНИЕ

Известно, что в процессе проектирования чертежа много времени конструктором тратится на редактирование. Несмотря на то что квалифицированный конструктор может традиционным способом начертить новый чертеж (начиная с нуля, т.е. без применения библиотек типовых фрагментов изображений и прототипов) быстрее, чем с помощью системы Автокад, для исправления чертежа или его модификации применение редакторских возможностей Автокада поможет значительно сократить временные затраты и повысить точность геометрических построений. Существенной особенностью автоматизированного проектирования является использование прототипов создаваемого изделия. Чем больше используются ранее разработанные конструкции, тем быстрее создаются новые. Это сравнительно легко осуществить, используя функции редактирования, предоставляемые системой Автокад, которые позволяют:

- удалять фрагменты изображения
- восстанавливать случайно удаленные фрагменты
- перемещать или поворачивать фрагменты или отдельные изображения относительно других
- копировать созданные фрагменты и располагать их в указанном месте
- увеличивать или уменьшать объекты
- создавать зеркально-симметричное изображение
- изменять свойства (принадлежность к слою, цвет и тип линии) созданных объектов
- сопрягать линии и строить фаски
- делить объекты на равные части или размечать на сегменты с заданным интервалом
- расчленять блоки или полилинии на составные части
- редактировать полилинии (сглаживать, изменять свойства и т.д.)
- растягивать части рисунка
- проводить линии, расположенные на заданном (постоянном) расстоянии относительно других

5.1. Полный перечень команд редактирования

Большинство команд редактирования сосредоточено в экранном подменю EDIT (РЕДАКТ), которое находится в корневом меню. Ниже перечислены все команды редактирования с краткими пояснениями. Простейшие команды редактирования будут рассмотрены в этой главе, а остальные команды - в главе 6.

ATTEDIT (АТРЕД)
Дает возможность редактирования атрибутов вне связи с блоками, изменяя их свойства

SELECT (ВЫБЕРИ)
Обеспечивает предварительный выбор объектов для дальнейших различных операций редактирования (при ссылке на них как на "текущий")

MIRROR (ЗЕРКАЛО)
Позволяет формировать зеркальные отражения существующих объектов, удаляя или сохраняя при этом оригиналы

CHANGE (ИЗМЕНИ)
Изменяет объекты и их свойства

COPY (КОПИРУЙ)
Копирует созданные объекты, оставляя оригиналы нетронутыми, и размещает копии в заданном месте или на заданном расстоянии от оригинала

ARRAY (МАССИВ)
Позволяет получать несколько копий выбранных объектов, группируя их в прямоугольной или круговой матрице

SCALE (МАСШТАБ)
Изменяет размеры одного или нескольких созданных объектов. Изображение при этом изменяется относительно заданной базовой точки

TRIM (ОБРЕЖЬ)
Удаляет части объектов между пересекающими его другими объектами (т.н. режущими кромками)

UNDO (ОТМЕНИ)
Отменяет последовательно выполненные действия предыдущих команд

MOVE (ПЕРЕНЕСИ)
Обеспечивает плоскопараллельный перенос одного или нескольких объектов в указанное место

ROTATE (ПОВЕРНИ)
Поворачивает группу объектов вокруг заданной точки на заданный угол

DIVIDE (ПОДЕЛИ)
Делит примитив на заданное число равных частей и может размещать блоки в точках деления

OFFSET (ПОДОБИЕ)
Позволяет проводить эквидистантные линии на заданном расстоянии

PEDIT (ПОЛРЕД)
Позволяет редактировать двумерные и трехмерные полилинии и трехмерные многоугольные сети (изменение ширины, сглаживание и т.п.)

MEASURE (РАЗМЕТЬ)

Располагает метки вдоль объекта (отрезок, дуга, окружность или полилиния) с заданным интервалом. В точки разметки можно вставлять блоки

BREAK (РАЗОРВИ)

Стирает часть отрезка, полосы, окружности, дуги или двумерной полилинии и разбивает эти примитивы на две части

STRETCH (РАСТЯНИ)

Обеспечивает перемещение выбранной части изображения, сохраняя при этом связь с остальной частью

EXPLODE (РАСЧЛЕНИ)

Заменяет блоки на отдельные примитивы, а также разбивает полилинии на сегменты. Заменяет многоугольную сеть набором трехмерных граней

PROPERTIES (СВОЙСТВА)

Обеспечивает подмножество действий команды ИЗМЕНИ: изменяет свойства созданных примитивов - цвет, принадлежность слою, тип линии, высоту и т.д.

FILLET (СОПРЯГИ)

Плавно сопрягает отрезки, дуги и окружности дугами разного (в т.ч. и нулевого) радиуса, а также сопрягает полилинии. При этом "лишние" части примитивов автоматически удаляются

ERASE (СОТРИ)

Удаляет из чертежа выбранные объекты

CHAMFER (ФАСКА)

Проводит линию фаски, удаляя ненужные части примитивов

EXTEND (УДЛИНИ)

Удлиняет отрезки, дуги и двумерные полилинии до пересечения их с аналогичными примитивами (т.н. граничными кромками)

5.2. Управление режимом слежения

При отрисовке и редактировании объектов Автокад позволяет динамически отслеживать их положение на экране. Режимом отслеживания управляет команда DRAGMODE (СЛЕДИ), которая находится в подменю SETTINGS (НАСТРОЙ). Существует три режима слежения - ON (Вкл), OFF (Откл) и Auto (Авто):

Auto (Авто)

В командах редактирования (там, где это возможно) режим слежения включается автоматически. Этот режим обычно установлен по умолчанию

ON (Вкл)

Режим слежения включен, но в командах редактирования для отслеживания объектов следует его каждый раз запрашивать явно

OFF (Откл)
Режим слежения отключен

Режим слежения требует значительных вычислительных ресурсов, и поэтому, если у вас медленная машина, при редактировании большого набора объектов или сложного примитива (например, штриховки) Автокад может работать медленно. В таких случаях следует подкомандой DRAG OFF (СЛЕДИ Откл) или DRAG ON (СЛЕДИ Вкл) отключать режим Auto (Авто). В режиме DRAG ON (СЛЕДИ Вкл) можно включать режим слежения на одну команду редактирования. Для этого нужно в процессе редактирования после завершения выбора объектов указать ключевое слово DRAG (СЛЕДИ).

Начальная установка режима слежения в новом рисунке определяется рисунком-прототипом. Последнее состояние режима слежения сохраняется вместе с другими настройками при сохранении файла чертежа.

Режим слежения управляется системной переменной DRAGMODE, значение которой можно менять при помощи команды SETVAR (УСТПЕРЕМ) или из Автолиспа.

5.3. Средства выбора объектов

Все действия редактирования могут производиться не только над одним, но и над несколькими объектами одновременно. Поэтому каждая команда редактирования требует выбора объектов, которые будут составлять набор объектов для редактирования:

```
Select objects:
```

```
Выберите объекты:
```

Объекты можно выбирать различными способами, которые рассматриваются ниже. Все выбранные объекты подвергнутся действию команды редактирования. Последний набор объектов хранится в памяти и при выполнении новой команды редактирования можно либо использовать предыдущий набор, либо составить новый (при этом новый набор замещает предыдущий).

Процедура выбора объектов едина для всех команд редактирования и включает в себя несколько режимов выбора объектов. Во всех командах редактирования она вызывается автоматически, однако, используя команду SELECT (ВЫБЕРИ), можно составить набор объектов без вызова команд редактирования.

Режимы выбора можно комбинировать, вызывая один режим за другим. Завершение выбора объектов осуществляется нажатием клавиши RETURN или правой клавиши "мыши".

При любом режиме выбора объектов могут быть выбраны только видимые на экране примитивы. После каждого указания Автокад просматривает изображение на экране и выделяет выбранные объекты подсветкой. Исключение составляет

режим Multiple (Несколько), в котором можно указать несколько объектов, а рисунок просматривается Автокадом по завершении режима.

Выбор объектов прицелом является основным режимом выбора объектов. При завершении других режимов выбора объектов он возобновляется автоматически. В этом режиме перекрестие курсора заменяется на квадратик, который называется *прицелом выбора объектов*. При нажатии на левую клавишу будет выбран один объект, пересекающий или попавший в прицел. Из нескольких объектов будет выбран тот, который был создан последним. Координаты центра прицела можно ввести с клавиатуры любым возможным способом. Для выбора закрашенных объектов необходимо, чтобы прицел пересекал их край. Размер прицела в пикселах определяется системной переменной PICKBOX, значение которой можно установить командой SETVAR (УСТПЕРЕМ), пунктом PICKBOX: (ПРИЦЕЛ:) подменю SETTINGS (НАСТРОЙ) экранного меню или средствами Автолиспа.

Window (Рамка)

Объекты выбираются путем обрамления их прямоугольной рамкой. Рамка определяется указанием двух точек - ее противоположных вершин. В набор редактирования заносятся те объекты, которые целиком находятся в рамке

Crossing (Секрамка)

То же, что Window (Рамка), но выбираются не только целиком попавшие в рамку объекты, но и те, которые пересекаются рамкой

BOX (БОКС)

Этот режим объединяет в себе свойства Window (Рамки) и Crossing (Секрамки). Если второй угол рамки указывается правее первого, то работает Window (Рамка), если левее - Crossing (Секрамка) (используется в основном в макроопределениях меню)

Auto (Авто)

Этот режим объединяет в себе режим указания объектов прицелом и режим BOX (БОКС). При вызове этого режима сначала предлагается указать объекты прицелом; если при таком указании не будет найдено ни одного объекта, то задействуется режим BOX (БОКС), причем в качестве первого угла рамки берется центр прицела при предыдущем указании (используется в основном в макроопределениях меню)

Multiple (Несколько)

В этом режиме, в отличие от всех остальных, можно указать несколько объектов; рисунок будет просмотрен только по завершении режима (правая клавиша "мыши" или ENTER)

SIngle (Единственный)

Выбор этой опции приводит к автоматическому завершению выбора объектов при нахождении первого же примитива (используется в основном в макроопределениях меню)

Last (Последний)
Выбирается последний отрисованный объект (из видимых на экране)

Previous (Текущий)
Передает команде редактирования набор объектов, составленный в предыдущей команде редактирования или командой SELECT (ВЫБЕРИ)

Undo (Отмени)
Отменяет результат работы последней опции

Remove (Удалить)
По умолчанию объекты заносятся в набор. Выбрав эту опцию, вы можете, еще раз указав подсвеченные объекты, вывести их из набора

Add (Добавить)
Эта подкоманда отменяет предыдущий режим, позволяя продолжить заполнение набора

До тех пор, пока не будет завершен выбор объектов, по выходе из любого режима Автокад возвращается к выбору объектов прицелом (для отмены этого правила пользуйтесь опцией Single (Единственный). В течение одного "сеанса выбора" можно пользоваться несколькими режимами, выбирая их в любой последовательности.

По завершении каждого режима Автокад просматривает чертеж и сообщает, сколько найдено объектов, выдавая сообщение:

```
Select objects: ... selected, ... found
(... duplicate)
```

```
Выберите объекты: ... выбран(ы), ... найден(ы)
(дублированных : ...)
```

Некоторые команды редактирования налагают определенные ограничения на способ выбора объектов. Эти ограничения упоминаются ниже при описании команд.

В падающем меню Edit (Редакт) все команды редактирования вызываются с автоматической установкой выбора объектов в режимах SIngle (Единственный) и Auto (Авто), поэтому при вызове из падающего меню они работают несколько иначе, чем при вызове из экранного меню или с клавиатуры. Разбирая команды редактирования, пробуйте вызывать их всеми возможными способами.

Вернемся к работе над чертежом. На примере команды ПЕРЕНЕСИ подробно рассмотрим средства выбора объектов.

5.4. Перенос объектов

Попытаемся сдвинуть группу отрезков, скажем, вправо. Выберите команду MOVE (ПЕРЕНЕСИ) из меню EDIT (РЕДАКТ), которое находится в корневом меню.

Для выбора объекта воспользуйтесь режимом Window (Рамка):

```
Command: MOVE
Select objects: W
First corner:
```

```
Команда: ПЕРЕНЕСИ
Выберите объекты: Р
Первый угол:
```

Укажите точку, которая будет являться одним из углов прямоугольной области выделения отрезков. При этом в командной строке появится сообщение:

```
Other corner:
```

```
Другой угол:
```

Попробуйте подвигать курсор. Вы увидите рамку, которая подобно "резиновой линии" отслеживает перемещения курсора. Увеличивайте рамку до тех пор, пока нужные отрезки не будут полностью находиться внутри рамки. Зафиксируйте другой угол рамки нажатием левой клавиши "мыши". Автокад поместит в набор редактирования все объекты, которые целиком попали в рамку. Те объекты, которые "вылезают" за рамку, в набор не вносятся. Число выбранных объектов выводится в командной строке:

```
3 selected, 3 found
```

```
3 выбран(ы), 3 найден(ы)
```

При помещении объекта в набор редактирования он выделяется на экране белым пунктиром. После этого можно продолжить выбор объектов при помощи любого другого средства выбора объектов.

```
Select objects:
```

```
Выберите объекты:
```

Если чертеж насыщенный и в рамку попали лишние объекты, то мы можем их исключить из набора при помощи подкоманды Remove (Удали):

```
Select objects: R
Remove objects:
```

```
Выберите объекты: У
Удалите объекты:
```

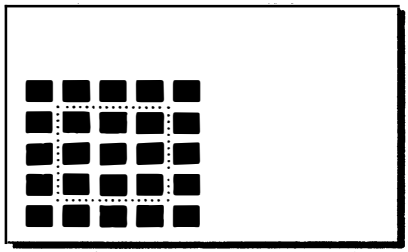
При удалении объектов из текущего набора можно пользоваться всеми средствами выбора объектов, в том числе Window (Рамка). Можно многократно переходить из режима удаления объектов из набора в режим пополнения набора. Окончив выбор объектов, надо нажать правую клавишу "мыши" или RETURN.

Нажав на клавишу RETURN, вы переходите к собственно команде MOVE (ПЕРЕНЕСИ) (описанный выше процесс выбора объектов практически одинаков для всех команд редактирования). В команде MOVE (ПЕРЕНЕСИ) требуется указать вектор переноса выбранной группы объектов:

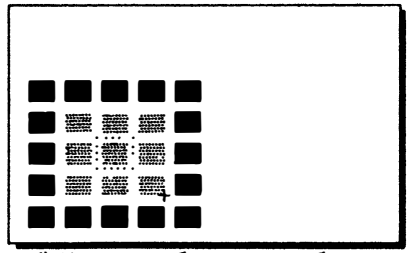
```
Base point or displacement: 4,4
Second point of displacement: 1.8,2
```

```
Базовая точка или смещение: 4,4
Вторая точка смещения: 1.8,2
```

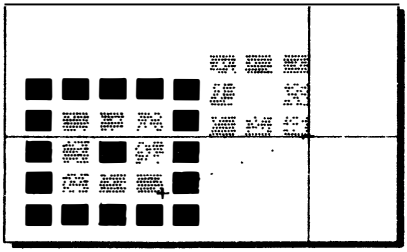
После указания вектора переноса выбранная группа объектов переносится на заданный вектор.



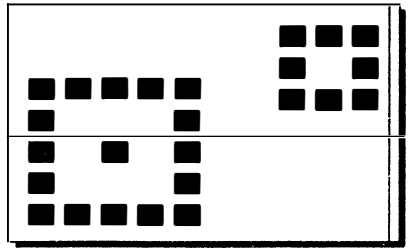
а) Выбор объектов рамкой



б) Удаление объекта из набора



а) Режим слежения



б) Результат

5.5. Копирование объектов

Команда COPY (КОПИРУЙ), так же как и команда MOVE (ПЕРЕНЕСИ), обеспечивает плоскопараллельный перенос созданных объектов, но оставляет

оригинал нетронутым. Созданная копия не зависит от оригинала и является самостоятельным примитивом.

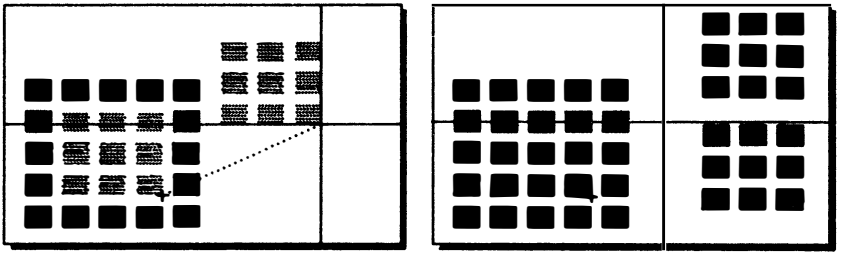
```
Command: COPY
Select objects:
<Base point or displacement>/Multiple:
Second point of displacement:
```

```
Команда: КОПИРУЙ
Выберите объекты:
<Базовая точка или перемещение>/Несколько:
Вторая точка перемещения:
```

Выбрав в команде COPY (КОПИРУЙ) опцию Multiple (Несколько) многократного копирования, вы можете копировать выбранные объекты несколько раз. Для этого после вызова команды наберите с клавиатуры букву "M" ("Н") или выберите эту опцию из экранного меню:

```
Command: COPY
Select objects:
<Base point or displacement>/Multiple: M
```

```
Команда: КОПИРУЙ
Выберите объекты:
<Базовая точка или перемещение>/Несколько: Н
```



а) Перенос копий выбранных объектов

б) Результат копирования в режиме Несколько

После этого вы определяете базовую точку и многократно указываете вектор копирования, помещая копируемый объект в разные места. Для выхода из режима многократного копирования нажмите правую клавишу "мыши" или RETURN.

```
Base point: <укажите точку>
Second point of displacement: <укажите точку>
Second point of displacement: <укажите точку>
Second point of displacement: RETURN
```

```
Базовая точка: <укажите точку>
Вторая точка перемещения: <укажите точку>
Вторая точка перемещения: <укажите точку>
Вторая точка перемещения: RETURN
```

5.6. Размножение объектов

Эта команда позволяет размещать копии объекта или группы объектов в прямоугольной или круговой структуре (матрице). С каждым получившимся в результате выполнения команды объектом можно манипулировать независимо от остальных.

```
Command: ARRAY
Select objects:
Rectungular or Polar array (R/P):
```

```
Команда: МАССИВ
Выберите объекты:
Прямоугольный или Круговой массив (П/К):
```

Прямоугольный массив

В случае прямоугольного массива Автокад запрашивает количество строк и столбцов. По умолчанию эти значения полагаются равными единице

```
Command: ARRAY
Select objects:
1 select, 1 found
Rectungular or Polar array (R/P): R
Number of rows (---) <1>.
Number of columns (|||) <1>.
```

```
Команда: МАССИВ
Выберите объекты:
1 выбран, 1 найден
Прямоугольный или Круговой массив (П/К). П
Число строк (---) <1>:
Число столбцов (|||) <1>.
```

Количество строк и столбцов должно быть целым числом (в машинной арифметике наибольшее целое число равно 32000). При вводе неверного числа Автокад выдает сообщение

```
Requires an integer value
```

```
Требуется целое значение
```

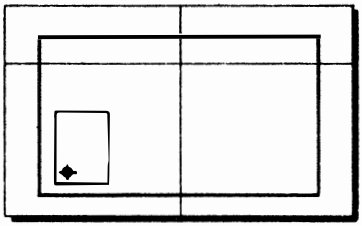
После этого предлагается повторить ввод числа. Если ни число строк, ни число столбцов не отличается от единицы, то команда отвергается. После указания числа строк и столбцов система выдает запросы:

```
Unit cell or distance between rows (---):
Distance between columns (||||):
```

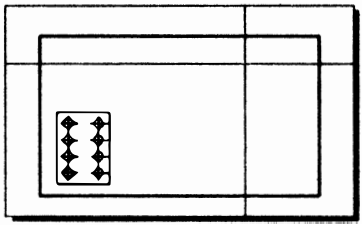
```
Размер ячейки или расстояние между строками (---):
Расстояние между столбцами (||||):
```

Вы можете указать смещение между строками или столбцами указанием двух точек на экране или с клавиатуры. Можно задавать как положительные, так и отрицательные смещения:

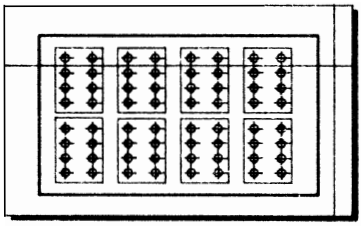
```
+X размещает столбцы вправо от выбранного объекта
-X размещает столбцы влево
+Y размещает строки вверх
-Y размещает строки вниз.
```



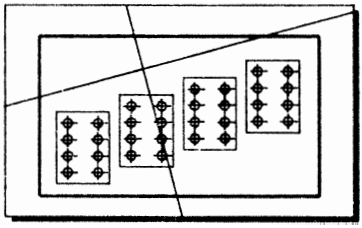
а) Объекты для формирования прямоугольного массива



б) Прямоугольный массив из одного объекта



в) Прямоугольный массив из группы объектов



г) Прямоугольный массив с повернутым углом шага

Для создания повернутого массива следует предварительно изменить угол шага.

Круговой массив

Для создания кругового массива надо в ответ на запрос о его типе ввести:

```
Command: ARRAY
Select objects:
1 select, 1 found
Rectungular or Polar array (R/P): P
```

```
Команда: МАССИВ
Выберите объекты:
1 выбран, 1 найден
Прямоугольный или Круговой массив (П/К): К
```

Автокад выдаст запрос:

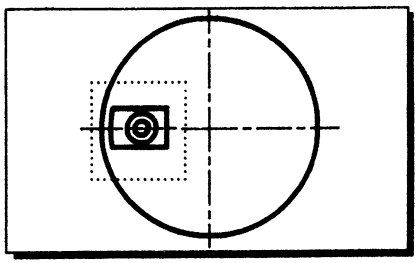
```
Center point of array:
```

```
Центр массива:
```

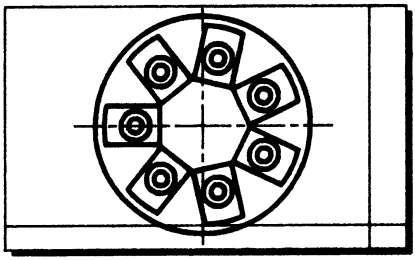
В ответ следует указать точку, вокруг которой нужно разместить выбранные объекты для формирования массива. Затем Автокад запросит три параметра:

- число элементов массива
- угол заполнения
- угол между элементами массива

Объекты могут быть повернуты или не повернуты при копировании вдоль дуги кругового массива.



а) Выбор объектов для кругового массива



б) Круговой массив с изменением ориентации объектов

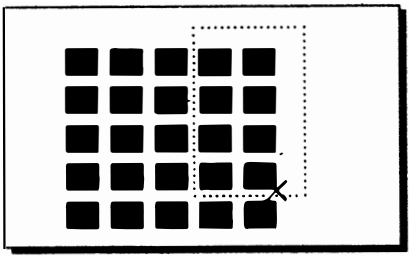
5.7. Поворот объектов

Команда ROTATE (ПОВЕРНИ) позволяет поворачивать объекты вокруг точки вращения (базовой точки). Попробуем повернуть объект на 30 градусов по часовой стрелке вокруг некоторой точки:

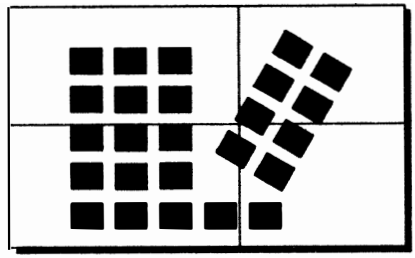
```
Command: ROTATE
Select objects:
Base point:
<Rotation angle>/Reference: -30
```

```
Команда: ПОВЕРНИ
Выберите объекты:
Базовая точка:
<Угол поворота>/Ссылка: -30
```

Задание отрицательного угла вращения приводит к повороту объекта по часовой стрелке, а положительного - против часовой стрелки, что определяется настройкой по умолчанию UNITS (ЕДИНИЦ). Удобно использовать режим Reference (Ссылка). Обычно угол поворота отсчитывается от оси X. Если же надо задать угол поворота относительно некоторого произвольно расположенного вектора, нужно воспользоваться режимом Reference (Ссылка). В этом режиме вы должны задать угол ссылки, от которого будет отсчитываться угол поворота.



а) Выбор объектов для поворота



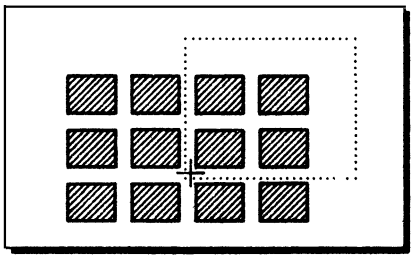
б) Результат поворота на -30 градусов

5.8. Изменение размеров объектов масштабированием

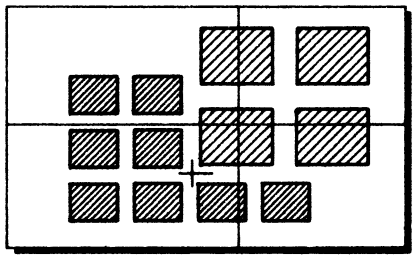
Данная команда изменяет размер существующих примитивов и подобна команде ROTATE (ПОВЕРНИ). После выбора объектов надо указать относительный масштабный коэффициент, на который умножаются оба измерения X и Y выбранных объектов. Масштабирование осуществляется как гомотетия относительно указываемой базовой точки (аналогично повороту). Так же как и в команде ROTATE (ПОВЕРНИ), для указания масштабного коэффициента можно использовать режим Reference (Ссылка):

```
Command: SCALE
Select objects:
1 selected, 1 found
Select objects:
Base point:
<Scale>/Reference: R
Reference length <1>:
```

```
Команда: МАСШТАБ
Выберите объекты:
1 выбран, 1 найден
Выберите объекты:
Базовая точка:
<Масштаб>/Ссылка: С
Длина ссылки <1>:
```



а) Выбор объектов для масштабирования



б) Результат (масштабный коэффициент 1.5)

5.9. Удаление объектов

При помощи команды ERASE (СОТРИ) вы можете удалить из чертежа ненужные объекты:

```
Command: ERASE
Select objects:
```

```
Команда: СОТРИ
Выберите объекты:
```

Здесь мы еще раз обратим ваше внимание на то, что команда ERASE (СОТРИ), так же как и другие команды редактирования, работает по-разному при выборе ее из падающего и из экранного меню. При выборе команды ERASE (СОТРИ) из падающего меню в том случае, если в квадратик "стирательной резинки" не попал ни один объект, Автокад считает, что вы хотите использовать рамку для указания объектов. При этом если второй угол рамки находится на экране правее первого, то работает Window (Рамка), а если левее - Crossing (Секрамка). Это большое удобство, поэтому мы рекомендуем выбирать команду ERASE (СОТРИ) из падающего меню.

```
First corner:
```

```
Первый угол:
```

Подвигайте курсор и посмотрите, как меняется рамка.

```
Second corner:
```

```
Второй угол:
```

После удаления выбранных объектов команда ERASE (СОТРИ) автоматически возобновляется, так что ее надо отменять явно, либо выбором другой команды из меню, либо нажатием клавиш CTRL C.

Отмена последней команды ERASE (СОТРИ)

Все объекты, стертые последней командой ERASE (СОТРИ), запоминаются и могут быть восстановлены командой OOPS (ОЙ). Команда OOPS (ОЙ) не может восстановить объекты, удаленные из чертежа несколькими последовательными командами ERASE (СОТРИ), поскольку команда ERASE (СОТРИ) каждый раз обновляет список.

Команду OOPS (ОЙ) проще всего вызывать с клавиатуры:

```
Command: OOPS
```

```
Команда: ОЙ
```

Список объектов, используемый командой OOPS (ОЙ), очищается командами PLOT (ЧЕРТИ) и PRPLOT (ПЕЧАТАЙ). Команду OOPS (ОЙ) можно использовать после команд BLOCK (БЛОК) и WBLOCK (ПБЛОК), поскольку они также стирают выбранные объекты. Команду OOPS (ОЙ) можно отменить командами U (О) и UNDO (ОТМЕНИ).

5.10. Отмена действия команд

Автокад может вести полный протокол вашей работы с момента начала сеанса редактирования. Это предоставляет дополнительные возможности при создании чертежей - можно отменить одну или даже все команды с момента начала сеанса редактирования и даже отменить саму команду UNDO (ОТМЕНИ). Протокол вашего общения с Автокадом записывается во временный файл на диске. Этот файл все время открыт (в него все время пишется протокол) и никогда не закрывается (не сохраняется), поэтому по окончании сеанса редактирования вы его на диске не обнаружите. Туда записываются все действия, изменяющие чертеж. Временный файл протокола обновляется также командами PLOT (ЧЕРТИ) и PRPLOT (ПЕЧАТАЙ), так что вы не можете отменить команды, которые им предшествовали.

Если отменяются команды, в которых устанавливались какие-то системные переменные или выполнялись прозрачные команды, то последние отменяются

наряду с основной командой. Отмену последней выполненной операции можно произвести командой U (О):

```
Command: U
```

```
Команда: О
```

Команду U (О) можно выполнить несколько раз, возвращаясь каждый раз на один шаг назад до тех пор, пока рисунок не возвратится в свое первоначальное состояние в момент начала сеанса редактирования (до вызова редактора чертежей). Командой U (О) можно отменить команду OOPS (ОЙ).

Для отмены сразу нескольких команд пользуйтесь командой UNDO (ОТМЕНИ):

```
Command: UNDO
Auto/Back/Control/End/Group/Mark/<number>: 5
```

```
Команда: ОТМЕНИ
Авто/Обратно/Управление/Конец/Группа/Метка/<число>: 5
```

Указание числа отменяемых команд - режим по умолчанию. В вышеприведенном примере в этом режиме отменяется сразу 5 команд.

Команда UNDO (ОТМЕНИ) выполняет также ряд специальных операций. Она позволяет, в частности, пометить место, к которому можно вернуться затем при отмене объектов. Можно также объединить несколько команд в группу и затем отменить всю группу. Команда UNDO (ОТМЕНИ) имеет шесть подкоманд: Auto (Авто), Back (Обратно), Mark (Метка), Control (Управление), Group (Группа) и End (Конец).

Auto (Авто)

Требуется ответ ON/OFF (Вкл/Откл). Если опция UNDO Auto (ОТМЕНИ Авто) включена, любой пункт меню, независимо от его сложности, будет восприниматься как одна команда, отменяемая командой U (О)

Поясним это. Отключите режим UNDO Auto (ОТМЕНИ Авто). Используя пункт экранного меню Setup (Формат), настройте масштаб чертежа, единицы измерения, границы формата. Попытайтесь отменить настройку командой U (О). Отменится только последняя команда этого сложного пункта меню. В режиме UNDO Auto ON (ОТМЕНИ Авто Вкл.) этот пункт отменится целиком, как одна команда. Заметим, что если режим Auto (Авто) включен, то при отмене команд выдается сообщение GROUP (ГРУППА):

```
Command: U
GROUP
```

```
Команда: O
ГРУППА
```

Если же режим Auto (Авто) отключен, то при отмене команд Автокад сообщает имя отменяемой команды:

```
Command: U
LINE
```

```
Команда: O
ОТРЕЗОК
```

Back (Обратно) и Mark (Метка)

Команда UNDO Back (ОТМЕНИ Обратно) отменяет все команды до тех пор, пока не исчерпается протокол или не встретится метка начала группы. Если в протоколе не осталось ни одной метки, выдается предупреждение:

```
Back This will undo everything. OK? <Y>:
```

```
Эта команда отменит все. Продолжить? <Д>:
```

Вы отвечаете "Да" ("Y") или "Нет" ("N"). Снять Метку можно командой ОТМЕНИ Метка.

Control (Управление)

Включение и отключение команды UNDO (ОТМЕНИ). Хранение на диске файла протокола может занимать много места. Опция Control (Управление) позволяет отключать команду UNDO (ОТМЕНИ) - протокол не ведется, так что отменить ранее введенные команды невозможно. При вызове команды UNDO (ОТМЕНИ) Управление предлагается выбрать один из трех режимов:

All (Все)	-	ведется полный протокол (этот режим установлен по умолчанию)
None (Ничего)	-	файл протокола не создается
One (Одну)	-	в этом режиме в файле протокола сохраняется только одна (предыдущая) команда

Group (Группа) и End (Конец)

Объединение нескольких команд в группу. Группа команд отменяется целиком, как одна команда. Для того чтобы отметить начало группы, нужно выполнить команду UNDO Group (ОТМЕНИ Группа). После этого продолжайте работать с чертежом. После того как вы выполните команду UNDO End (ОТМЕНИ Конец), все команды, которые вы выполняли между ними, будут отменяться командами U (О) и UNDO 1 (ОТМЕНИ 1) как одна команда. Повторный ввод UNDO Group (ОТМЕНИ Группа) также завершает объявление группы.

Следите за тем, чтобы командой UNDO (ОТМЕНИ) не отменилось больше, чем требуется. Любые настройки, например SNAP (ШАГ) или прозрачные команды (например, 'SETVAR ('УСТПЕРЕМ) или 'ZOOM ('ПОКАЖИ), будут отменены вместе с основной командой.

Некоторые команды отменить нельзя. Нельзя отменить команду SAVE (СОХРАНИ) или внешние команды. Напомним, что вы не можете отменить команды, выполнявшиеся до выполнения команд PLOT (ЧЕРТИ) или PRPLOT (ПЕЧАТАЙ), поскольку они обновляют протокол.

Когда команда U (О) или UNDO 1 (ОТМЕНИ 1) встречает метку, выдается сообщение:

```
Mark encountered
```

```
Найдена метка
```

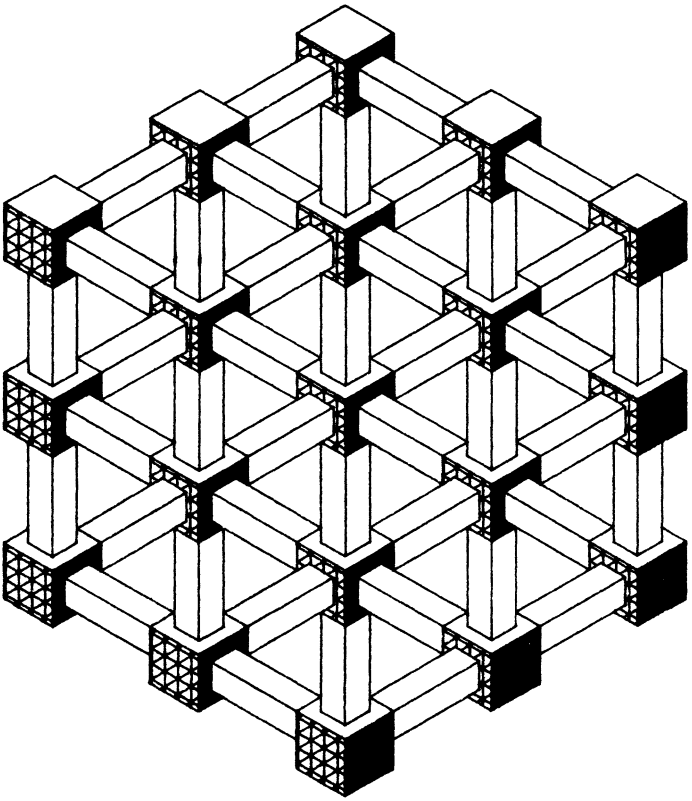
Еще одна команда U (О) UNDO 1 (ОТМЕНИ 1) стирает метку.

Отмена отмены - повторное выполнение

Предположим, что команда UNDO (ОТМЕНИ) отменила больше, чем хотелось. Если вы сразу же выполните команду REDO (ВЕРНИ), любой вариант команды UNDO (ОТМЕНИ) будет отменен. Это позволяет не беспокоиться о последствиях использования команды UNDO (ОТМЕНИ).

Например, при выполнении команды UNDO Back (ОТМЕНИ Обратно) или UNDO 20 (ОТМЕНИ 20) уничтожилось больше того, что требовалось. Можно вернуться к исходному состоянию командой REDO (ВЕРНИ) и попытаться повторить отмену правильно, отменив меньшее число шагов - например, UNDO 5 (ОТМЕНИ 5).

Будьте осторожны - команда UNDO (ОТМЕНИ) может уничтожить за одну команду весь чертеж. И, поскольку команда REDO (ВЕРНИ) позволяет отменить только одну (последнюю) команду UNDO (ОТМЕНИ), если сразу же не выполнить команду REDO (ВЕРНИ), чертеж может быть испорчен



ГЛАВА 6

СЛОЖНОЕ РЕДАКТИРОВАНИЕ

В предыдущей главе были рассмотрены очевидные средства редактирования изображений - команды COPY (КОПИРУЙ), MOVE (ПЕРЕНЕСИ) и подобные им, а также группа команд исправления ошибок. Команды этой главы более полно раскрывают возможности графического редактора, и, вероятно, освоение их приведет к пересмотру вами подхода к методике создания чертежей. Дело в том, что команды редактирования графического редактора Автокада настолько развиты и удобны, а набор их настолько хорошо продуман, что, освоив их, вы сможете более оперативно и более точно создавать чертежи, чем с помощью команд отрисовки примитивов.

6.1. Разделение сложных примитивов на составные

Примитивы могут быть простыми (отрезок, дуга и т.п.) и сложными - составными (полилинии, размеры, трехмерные сети и блоки). Если вы хотите изменить свойство (цвет, слой, определенные геометрические характеристики) какой-либо части сложного примитива, то такой сложный примитив предварительно следует расчленить на простые составные примитивы. Для этого используется команда EXPLODE (РАСЧЛЕНИ):

```
Command: EXPLODE
Select block reference, polyline, dimension or mesh:
```

```
Команда: РАСЧЛЕНИ
Выберите блок, полилинию, размер или сеть:
```

Ясно, что если составной примитив обладает некоторым свойством, которого нет у простых примитивов, то при расчленении оно исчезает. Например, полилиния, имеющая ненулевую ширину, при расчленении ее потеряет. В такой ситуации Автокад предупредит вас об этом:

```
Exploding this polyline has lost width information.
The UNDO command will restore it.
```

```
Расчленение этой полилинии привело к потере информации о ширине.
Команда ОТМЕНИ восстановит ее.
```

Аналогично при расчленении блока атрибут теряет свое значение (при этом никакого сообщения не выдается).

Блоки, вставленные командой MINSERT (МВСТАВЬ), и блоки, вставленные с неравными масштабными коэффициетами по X, Y и Z, нельзя расчленить.

6.2. Изменение свойств и параметров примитивов

Каждый примитив на рисунке принадлежит к какому-то слою, имеет определенный цвет и тип линии, уровень возвышения плоскости построений над текущей плоскостью X0Y и высоту по Z (иногда называемой величиной выдавливания). Эти свойства - общие свойства примитивов всех типов. Кроме них каждый объект характеризуется также геометрическим описанием (например, отрезок описывается двумя точками, круг описывается центром и радиусом и т.п.). Если необходимо изменить какое-то из этих свойств примитива, воспользуйтесь командой CHANGE (ИЗМЕНИ).

Следует учитывать, что некоторые из возможностей команды CHANGE (ИЗМЕНИ) можно использовать только в тех случаях, когда направление выдавливания редактируемого примитива параллельно оси Z текущей системы координат. Автокад контролирует, удовлетворяет ли этим требованиям каждый объект из набора. Если для какого-либо из объектов эти условия не справедливы, выдается сообщение об ошибке. Формат команды ИЗМЕНИ:

```
Command: CHANGE
Select objects:
1 select, 1 found
Select objects: RETURN
Properties/<Change point>:
```

```
Команда: ИЗМЕНИ
Выберите объекты:
1 выбран, 1 найден
Выберите объекты: RETURN
Свойства/<Точка изменения>:
```

6.2.1. Изменение общих свойств примитивов

Изменение общих свойств примитивов осуществляется подкомандой CHANGE Properties (ИЗМЕНИ Свойства) и командой CHANGE (СВОЙСТВА). Проиллюстрируем их работу на примере изменения цвета объекта.

```
Command: CHANGE
Select objects:
1 select, 1 found
Select objects: RETURN
Properties/<Change point>: P
```

```
Команда: ИЗМЕНИ
Выберите объекты:
1 выбран, 1 найден
Выберите объекты: RETURN
Свойства/<Точка изменения>: C
```

Тем самым мы выбрали подкоманду CHANGE Properties (ИЗМЕНИ Свойства). Команда PROPERTIES (СВОЙСТВА) практически полностью дублирует подкоманду CHANGE Properties (ИЗМЕНИ Свойства). Разница заключается в том, что команда PROPERTIES (СВОЙСТВА) в отличие от подкоманды CHANGE Properties (ИЗМЕНИ Свойства) может работать со всеми примитивами, независимо от их направления выдавливания. Отметьте, что именно команда PROPERTIES (СВОЙСТВА) находится в падающем подменю Edit (Редакт), опция Modify (Свойства).

После выбора команды нужно указать, какое свойство примитива вы хотите изменить. Обратите внимание на появление экранного меню команды CHANGE Properties (ИЗМЕНИ Свойства), из которого вы можете выбрать все свойства примитивов. Причем, если вы работаете с меню, то ключевое слово Properties (свойства) указывать необязательно, поскольку оно указывается автоматически при выборе свойства из меню. Если же вы работаете с клавиатурой, то нужно явно выбрать подкоманду PROPERTIES (СВОЙСТВА), как показано выше. Теперь выберите свойство изменения:

```
Change what property(Color/Elevation/LAyer/LType/Thickness)?: C
New color <3 (green)>: cyan
```

```
Изменить какое свойство(Цвет/Уровень/СЛой/Типлинии/Высота)?: Ц
Новый цвет <3 (зеленый)>: фиолетовый
```

В угловых скобках показывается текущее свойство примитива.

Аналогично изменяются и другие общие свойства. В одном вызове команды можно несколько раз изменять разные свойства: после завершения каждого изменения вновь появляется запрос:

```
Change what property (Color/Elevation/LAyer/LType/Thickness)?:
```

```
Изменить какое свойство(Цвет/Уровень/СЛой/Типлинии/Высота)?:
```

Выход из команды осуществляется нажатием правой клавиши "мыши" или клавиши RETURN. Нажатие CTRL C или выбор другой команды без завершения команды CHANGE Properties (ИЗМЕНИ Свойства) отменит все изменения. Заметим, что при работе с экранным меню команда работает иначе: вы можете изменить только одно свойство и только один раз на вызов команды, но зато вам не надо явно завершать команду, это делается автоматически после выбора значения свойства - экономится три операции. Тем самым вам предоставляется возможность выбрать тот режим работы, который вас наиболее устраивает. Заметим, что в подменю редактирования почти все команды работают несколько иначе, чем с клавиатуры. Также по-своему работают те же команды из падающего меню. Мы не будем подробно описывать эти детали: осваивая команду, пробуйте все варианты вызова из разных меню, а также при

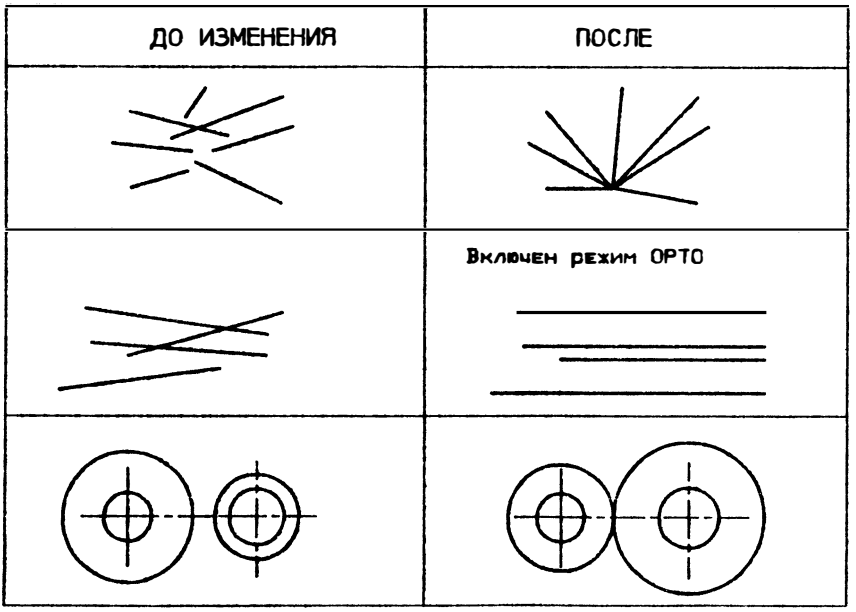
помощи клавиатуры (мы описываем везде работу с клавиатурой, потому что меню является всего лишь надстройкой над командами Автокада).

6.2.2. Изменение геометрических свойств примитивов

Можно менять геометрические свойства примитивов: координаты определяющих точек, размерные параметры и т.п. Для этого после выбора объекта нужно вместо выбора опции Properties (Свойства) указать точку изменения (заметим, что это выбор по умолчанию):

```
Properties/<Change point>:
```

```
Свойства/<Точка изменения>:
```



Конкретные действия команды зависят от типа примитива.

LINE (ОТРЕЗОК)

Ближайший к указанной точке конец отрезка переносится в новую точку. Можно соединить несколько отдельных отрезков в "пучок" с общей вершиной. Если режим ORTHO (ОРТО) включен, то в результате команды CHANGE (ИЗМЕНИ) могут строиться только ортогональные отрезки

CIRCLE (КРУГ)

Использование команды ИЗМЕНИ по отношению к окружности поможет вам провести окружность через заданную точку, не изменяя центра окружности

TEXT (ТЕКСТ)

Команда ИЗМЕНИ дает возможность менять базовую точку текстовой строки, гарнитуру, высоту, угол поворота строки, а также сам текст. Ответ RETURN оставляет прежние значения неизменными

ATTRIBUTE (АТРИБУТ)

Кроме аналогичных изменений свойств текста можно изменять определение атрибута: имя, текст "подсказки" и значение по умолчанию

BLOCK (БЛОК)

Команда ИЗМЕНИ дает возможность менять базовую точку вставки блока и угол поворота. Ответ RETURN оставляет прежние значения неизменными

6.3. Деление объекта на равные части

Командой DIVIDE (ПОДЕЛИ) можно разделить примитив на несколько равных частей, помещая точки в местах деления. Поделить можно отрезок, дугу, круг или полилинию:

```
Command: DIVIDE
Select object to divide:
```

```
Команда: ПОДЕЛИ
Выберите объект для деления:
```

Заметим, что эта команда работает с одним примитивом, поэтому набор не заполняется и сразу после выбора вами примитива выдается сообщение:

```
<Number of segmets>/Block:
```

```
<Число сегментов>/Блок:
```

Если вы введете число, то DIVIDE (ПОДЕЛИ) поместит в местах деления точки, отрисованные текущим цветом. Поэтому мы рекомендуем вам перед выполнением этой команды изменить цвет создаваемого примитива, иначе вы не увидите результатов выполнения команды.

Точки деления можно определить как точки вставки какого-то блока. Для этого нужно выбрать опцию Block (Блок):

```
<Number of segmets>/Block: B
```

```
<Число сегментов>/Блок. Б
```

После этого нужно указать имя блока:

```
Block name to insert:
```

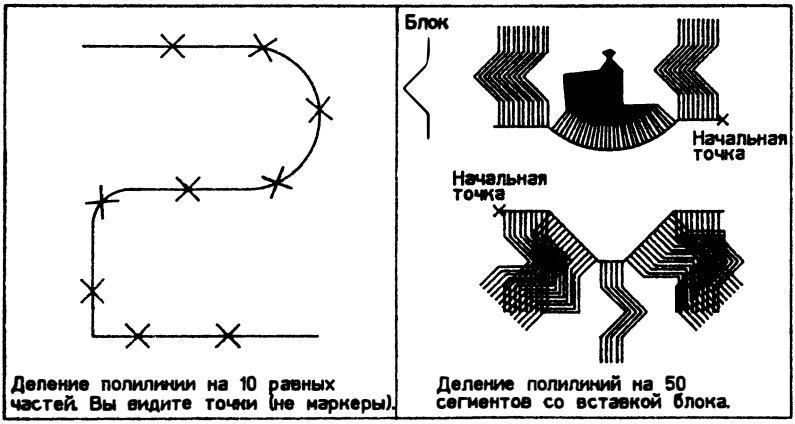
```
Имя блока для вставки:
```

При вставке блока можно определить его ориентацию по отношению к объекту деления:

```
Align block with object? <Y>:
Number of segmets:
```

```
Согласовать ориентацию блока с ориентацией объекта? <Д>:
Число сегментов:
```

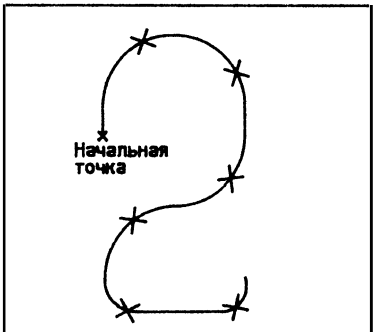
При ответе "No" ("Нет") блок будет вставляться с нулевым углом поворота. При ответе "Yes" ("Да") блок будет при вставке поворачиваться так, что его горизонтальные линии будут касательными к объекту в точках вставки.



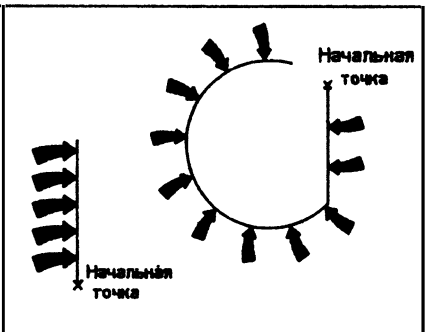
Обратите внимание на то, что при вставке с выравниванием по объекту ориентация вставляемых блоков зависит от того, как вы отрисовывали примитив: какая точка, например, отрезка была начальной, а какая конечной.

6.4. Разметка объекта равными интервалами

Команда MEASURE (РАЗМЕТЬ) подобна команде DIVIDE (ПОДЕЛИ) (см. выше) с той лишь разницей, что она обеспечивает разметку примитива, размещая точки вдоль объекта с заданным интервалом.



Разметка полилинии на заданные интервалы с использованием точек.



Размещение блоков вдоль объекта с заданным интервалом.

Вместо числа сегментов требуется указать длину сегмента разметки. Примитив будет делиться на сегменты заданной длины, начиная с его начальной точки.

6.5. Выполнение сопряжений

Выполнение плавных сопряжений дугами осуществляется при помощи команды FILLET (СОПРЯГИ). Эта команда обеспечивает плавное сопряжение двух отрезков, дуг и окружностей. При вызове команды можно либо указать радиус сопряжения, либо сразу указать два объекта:

```
Command: FILLET
Polyline/Radius/<Select two objects>:
```

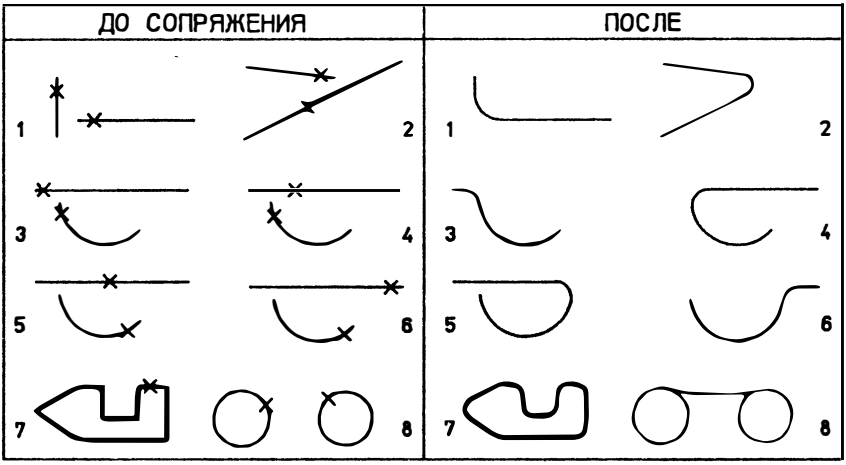
```
Команда: СОПРЯГИ
Плиния/Радиус/<Выберите два объекта>:
```

Сопряжение производится текущим радиусом сопряжения. Текущий радиус остается неизменным до тех пор, пока вы его не переопределите:

```
Command: FILLET
Enter fillet radius <0.0000>:
```

```
Команда: СОПРЯГИ
Введите радиус сопряжения <0.0000>:
```

В момент начала сеанса редактирования значение радиуса сопряжения равно нулю. При сопряжении объекты модифицируются: могут удлиняться или обрезаться.



Если сопрягаемые отрезки находятся на одном слое, то дуга сопряжения помещается на этот же слой, в противном случае она помещается на текущий слой. Аналогичное правило действует на цвет и тип линии дуги сопряжения.

Если вы рисовали контур детали полилинией, то можно одной командой скруглить все ее вершины:

```
Command: FILLET
Polyline/Radius/<Select two objects>: P
```

```
Команда: СОПРЯГИ
Плиния/Радиус/<Выбери два объекта>: П
```

Сопряжение нельзя построить для параллельных линейных сегментов, слишком коротких сегментов, для сегментов, пересекающихся за лимитами рисунка при включенном контроле соблюдения лимитов и для расходящихся сегментов

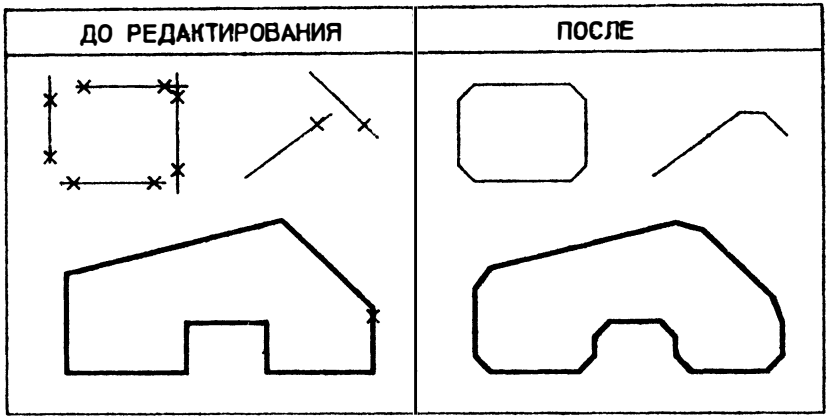
Если некоторые вершины полилинии не удалось скруглить текущим радиусом по разным причинам, Автокад сообщает, почему не выполнено сопряжение:

```
2 were parallel, 5 lines were filleted
4 were too short
```

```
2 были параллельны. 5 отрезков были сопряжены
4 были меньше допустимой величины
```

6.6. Построение фасок

Команда CHAMFER (ФАСКА) похожа на команду FILLET (СОПРЯГИ) (см. выше) с той лишь разницей, что она обеспечивает "подрезку" двух пересекающихся отрезков на указанном расстоянии от точки пересечения и соединение концов отрезков новым линейным сегментом. Подобно тому как в команде FILLET (СОПРЯГИ) можно устанавливать радиус сопряжения, в команде CHAMFER (ФАСКА) можно устанавливать обе длины фаски, которые принимаются как значения по умолчанию до следующей установки.



Команда CHAMFER (ФАСКА) использует расстояние между двумя "указанными" точками. По этой причине более точные результаты получаются при работе в плане. Если же направление взгляда не перпендикулярно плоскости XY текущей системы координат, то результаты могут заметно отличаться от желаемых. В таком случае Автокад выдает соответствующее предупреждение.

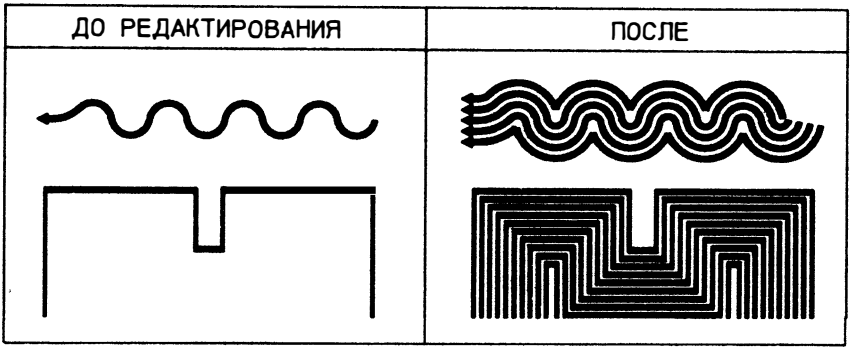
6.7. Проведение эквидистантных линий

Построение примитива, линия которого эквидистантна линии выбранного примитива, осуществляется командой OFFSET (ПОДОБИЕ):

```
Command: OFFSET
Offset distance or Through <Through>:
```

```
Команда: ПОДОБИЕ
Величина смещения или Точка <Точка>:
```

Если вы укажете величину смещения, то система предложит вам сначала выбрать объект для построения "подобного", а затем сторону смещения, после чего будет построен (если это возможно) новый примитив.



Можно явно задать точку, через которую будет проходить новый примитив: надо выбрать из меню или ввести с клавиатуры опцию Through (Точка).

6.8. Стирание части объекта

Команда BREAK (РАЗОРВИ) обеспечивает стирание части отрезка, полосы, окружности круга, дуги или двумерной полилинии, разбивая тем самым объект на два объекта того же типа (окружность преобразуется в дугу, а замкнутая полилиния преобразуется в незамкнутую). Команда запрашивает объект для разбиения и две конечные точки той части, которую нужно удалить:

```
Command: BREAK
Select object:
Enter second point (от F for first point):
```

```
Команда: РАЗОРВИ
Выберите объект:
Введите вторую точку (или П для первой точки):
```

Указываемая точка на выбранном объекте считается первой точкой, хотя вы можете ее переопределить (в этом случае введите в ответ не точку, а "Р" ("П").

Разбиение замкнутой полилинии Автокад начинает от первой вершины, и удаляет часть, находящуюся между заданными точками разрыва.

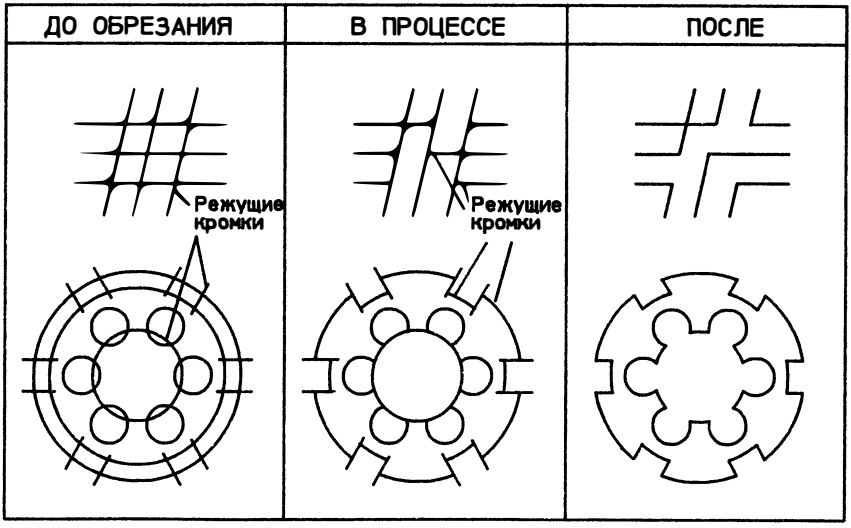
6.9. Отсечение части объекта

Команда TRIM (ОБРЕЖЬ) отсекает части выбранного объекта по режущим кромкам (примитивам, пересекающим выбранный объект).

```
Command: TRIM
Select cutting edge(s)...
Select objects:
Select object to trim:
```

```
Команда: ОБРЕЖЬ
Выберите режущие кромки...
Выберите объекты:
Выберите объект, который нужно обрезать:
```

При обрезании полилинии, вершины которой были сглажены дугами или сплайном информация о сглаживании пропадает и ее сегменты преобразуются в обычные дуговые сегменты.



6.10. Удлинение объекта

Для удлинения объектов до граничных кромок используется команда EXTEND (УДЛИНИ). Данная команда дополняет команду TRIM (ОБРЕЖЬ) и удлиняет существующие на рисунке объекты до граничных кромок (примитивы, которые пересекут выбранный объект при его удлинении). Причем предварительно нужно выбрать объекты, которые будут использоваться в качестве граничных кромок (их также может быть несколько), и, лишь закончив их выбор (нажатием RETURN), нужно выбрать объекты для удлинения:

```
Command: EXTEND
Select boundary edges(s)...
Select objects:
1 selected, 1 found
Select objects: RETURN
Select objects to extend:
```

```
Команда: УДЛИНИ
Выберите граничные кромки...
Выберите объекты:
1 выбран, 1 найден
Выберите объекты: RETURN
Выберите объект, который нужно удлинить
```

При использовании команды EXTEND (УДЛИНИ) каждый удлиняемый объект указывается отдельно - набор выбора объектов здесь не используется. При помощи этой команды нельзя укоротить объекты.

6.11. Трансформация фрагментов объекта

Очень интересные возможности предлагает команда STRETCH (РАСТЯНИ). Эта команда работает несколько необычно. Она сложным образом модифицирует выбранные примитивы, перенося в указанное место определяющие точки примитивов, выбранные при помощи рамки. При этом определяющие точки, оставшиеся за пределами рамки, не переносятся:

```
Command: STRETCH
Select objects to stretch by window
```

```
Команда: РАСТЯНИ
Выберите растягиваемые объекты с помощью рамки
```

Результат выполнения команды STRETCH (РАСТЯНИ) не всегда очевиден, поэтому при работе с ней полезно включать режим слежения, если он у вас отключен.

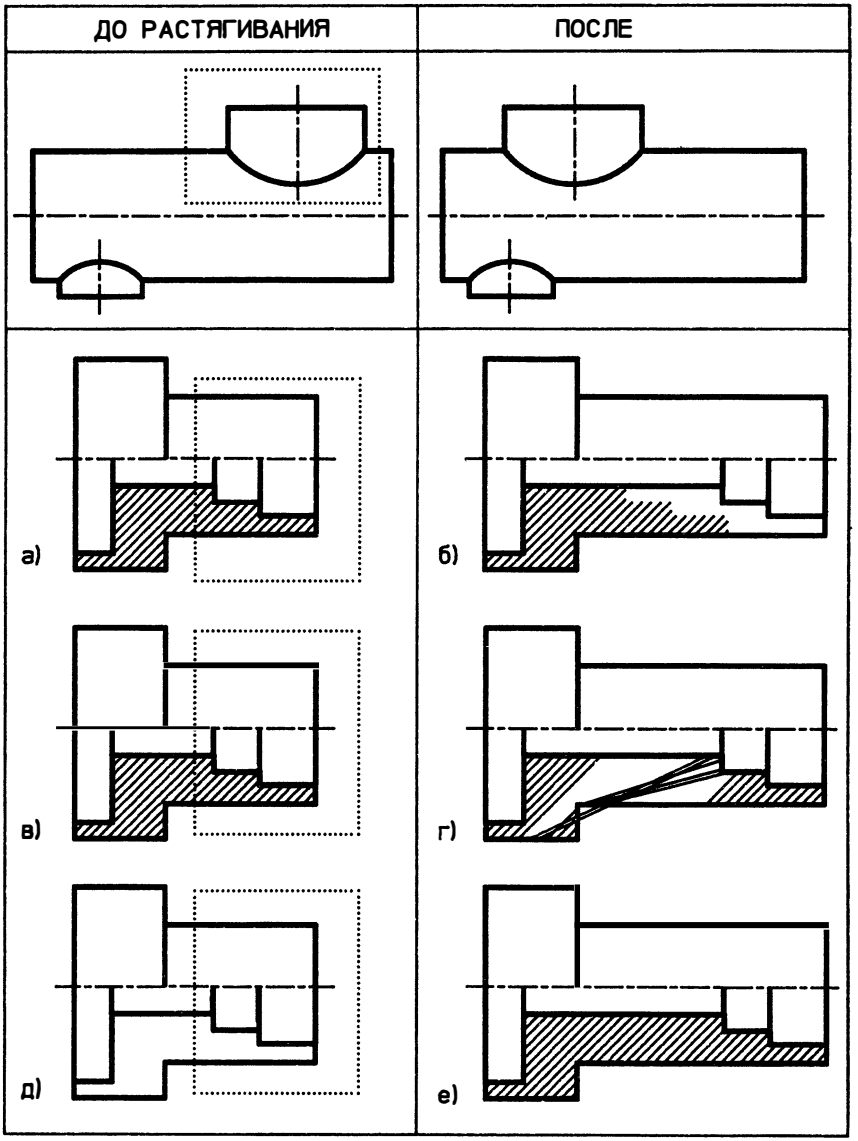
Для выбора объектов в команде STRETCH (РАСТЯНИ) нужно воспользоваться рамкой хотя бы один раз. В противном случае Автокад выдаст сообщение:

```
You must select a window to stretch
```

```
Здесь выбор разрешен только с помощью рамки
```

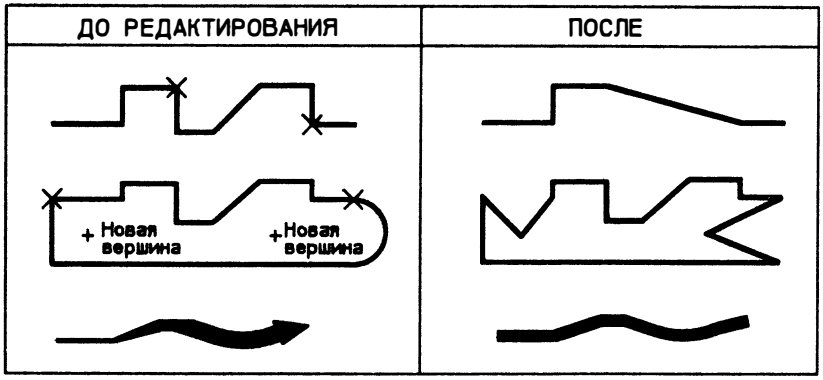
Если рамкой воспользовались более одного раза, то переноситься будут определяющие точки из последней рамки. Объекты могут свободно добавляться или удаляться из набора выбора.

Обратите внимание, как команда STRETCH (РАСТЯНИ) работает с блоками. На рисунке на следующей странице показано, что попытка удлинить деталь с заштрихованной областью (поз. б) оказывается неудачной. Расчленение блока штриховок перед командой STRETCH (РАСТЯНИ) также не помогает достичь желаемого результата. В подобных случаях следует предварительно удалить штриховку, а после команды STRETCH (РАСТЯНИ) выполнить ее снова.


ДО РАСТЯГИВАНИЯ
ПОСЛЕ
а)
б)
в)
г)
д)
е)

6.12. Редактирование полилиний

Команда PEDIT (ПОЛРЕД) является средством редактирования двух- и трехмерных полилиний и трехмерных многоугольных сетей:




```
Command: PEDIT
Select polyline:
```

```
Команда: ПОЛРЕД
Выберите полилинию:
```

После указания вами объекта Автокад проверяет, является ли выбранный примитив полилинией, и если нет, то выдает запрос:

```
Entity selected is not a polyline
Do you want to turn it into one? <Y>:
```

```
Выбранный объект не полилиния
Сделать его полилинией? <Д>:
```

В этот момент команда PEDIT (ПОЛРЕД) превращает выбранный объект в полилинию, состоящую из одного сегмента (если мы выбрали несколько отрезков, то превращается в полилинию первый из них). После этого выбранные объекты можно редактировать командой PEDIT (ПОЛРЕД), которая предлагает следующий выбор:

```
Close/Join/Width/Edit vertex/Fit curve/Spline curve/Decurve/
Undo/eXit<X>:
```

```
Замкни/Добавь/Ширина/Вершина/СГладь/СПлайн/Убери сгл./
Отмени/выХод <X>:
```

Если выбрана замкнутая полилиния, то первой опцией вместо Close (Замкни) будет Open (Разомкни). Рассмотрим вкратце опции команды PEDIT (ПОЛРЕД):

Close/Open (Замкни/Разомкни)

В результате выполнения этой команды на экране отрисовывается сегмент, соединяющий начальную и конечную точки полилинии. Скажем для пояснения, что полилиния с точки зрения Автокада - связанный список вершин, между которыми отрисовываются линейные сегменты. Замкнутая полилиния отличается от разомкнутой тем, что между первой вершиной полилинии и последней также отрисовывается линейный сегмент. Отсюда ясно, что если замыкающий полилинию сегмент построен явно (например, с помощью привязки к конечной точке полилинии), то замыкание такой полилинии не будет иметь видимого эффекта, впрочем как и размыкание

Join (Добавь)

Позволяет добавить к незамкнутой полилинии несколько сегментов других полилиний. Напомним, что при вызове команды PEDIT (ПОЛРЕД) мы могли создать набор из нескольких отрезков, каждый из которых был превращен нами в отдельную полилинию. Теперь, выбрав опцию Добавь, мы можем превратить их в одну полилинию. Заметим, что если отрезки отрисованы неточно и их вершины не совпадают, то их нельзя объединить в полилинию

Width (Ширина)

Позволяет установить единую ширину для всех сегментов полилинии

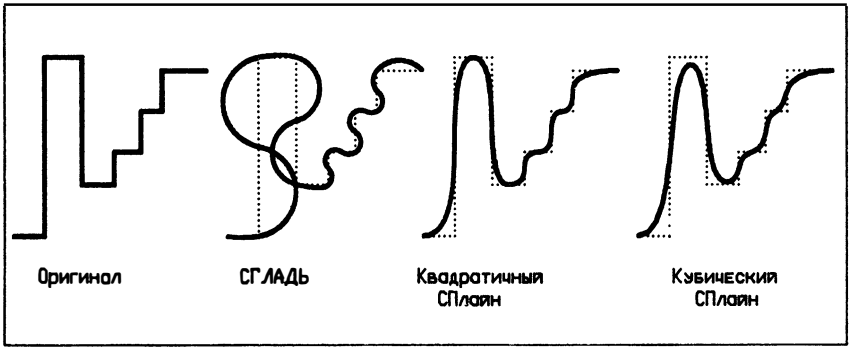
Edit vertex (Вершина)

Предлагает выбрать одну вершину полилинии и выполнить над ней и прилегающими сегментами различные операции редактирования: разорвать полилинию, вставить дополнительную вершину, перенести выбранную вершину в другое место, заменить несколько сегментов между двумя выбранными вершинами одним сегментом ("выпрямить" участок полилинии), установить разную ширину начала и конца выбранного сегмента

Fit curve/Decurve (Сгладь/Убери сгл.)
Spline curve/Decurve (Сплайн/Убери сгл.)

Эти опции вычисляют гладкую кривую, сглаживающую все вершины полилинии. Полилинию можно сгладить дугами окружности (Fit curve (Сгладь), используя при этом заданные направления касательных) и полиномами различного порядка (Spline (Сплайн). При сглаживании полиномами тип полинома (квадратический, кубический, Безье) можно выбрать из графического меню пиктограмм, которое может быть вызвано из экранного меню (тип сплайна управляется системной переменной SPLINETYPE). При сглаживании в полилинию добавляются дополнительные вершины, а старые вершины полилинии, сохраняясь в описании полилинии, переводятся в особое состояние "контрольных точек". Поэтому такое сглаживание обратимо: мы в любой момент можем отменить сглаживание - (Decurve (Убери сгл.), при этом добавленные при вычислении кривой дополнительные вершины будут удалены, а истинные

вершины полилинии будут восстановлены. В любой момент можно также отобразить истинные вершины вместе с вычисленной кривой. Для этого нужно установить в 1 системную переменную SPLFRAME. Число добавляемых при вычислении кривой вершин определяется системной переменной SPLINESEGS, которая по умолчанию равна 8



Undo (Отмени)

Отменяет действие самых последних операций редактирования команды PEDIT (ПОЛРЕД). Путем многократного ввода ОТМЕНИ можно вернуться к началу выполнения команды PEDIT (ПОЛРЕД). Эта команда не может отменить преобразование отрезков в полилинии перед вызовом команды PEDIT (ПОЛРЕД) - для этого нужно вернуться в режим команд и воспользоваться командой UNDO (ОТМЕНИ)

eXit (выХод)

Выход из команды ПОЛРЕД

Пакет Time Line фирмы Symantec - это решение проблем планирования для Вас

Если при переходе к работе в условиях рыночной экономики Вы хотите:

- в любой момент времени иметь информацию о состоянии дел по проектам, над которыми работаете
- обезопасить себя от неожиданных срывов в производстве
- эффективно использовать и экономить ресурсы, финансы и время

то Вам необходим пакет Time Line, который можно использовать для планирования и контроля собственной деятельности, отдельных проектов или предприятия в целом.

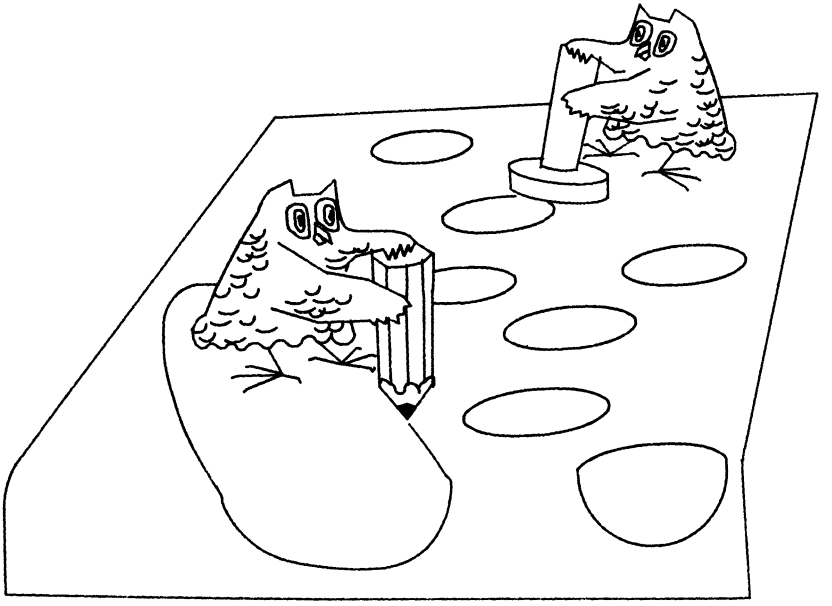
Пакет Time Line фирмы Symantec - это более чем 60% рынка программных продуктов в области средств управления проектами в США, где этим пакетом оснащены более 500 тысяч персональных компьютеров.

Наиболее широко он используется в строительстве, машиностроении, научно-исследовательских проектах. При разработке программных средств его используют такие всемирно известные фирмы, как Microsoft и Ashton-Tate.

"ДИАЛОГ-МИФИ"
организует демонстрации возможностей этого пакета
и обеспечит Вас дополнительной информацией и демонстрационными программами.
По демпинговым ценам распространяет учебную версию Time Line.

Специально для Советского Союза фирма Symantec и "ДИАЛОГ-МИФИ" разработали русскую версию Time Line с полным комплектом документации.

ДИАЛОГ-МИФИ
СССР, 115409 Москва, ул. Москворечье, 31-2
Телефон (095) 3203211, (095) 3247166
Телефакс (095) 3243055,
(095) 2926511 для 6206 TimeLine



ГЛАВА 7

РАБОТА С ПРИНТЕРОМ И ПЛОТТЕРОМ

Несмотря на прогресс информационных технологий, чертежная документация на бумаге не теряет своего значения. В этой главе мы вкратце рассмотрим вопросы вывода чертежей Автокада на принтер и плоттер. Принтер и плоттер должны быть правильно установлены в системе, т.е. вы должны указать Автокаду, устройством какого типа и модели вы пользуетесь, а также определить некоторые вспомогательные параметры. Такая настройка производится один раз, при установке Автокада (или при смене периферийного устройства), поэтому, если в вашей системе принтер или плоттер установлены правильно, вы можете пропустить следующий пункт и сразу перейти к пункту 7.2.

7.1. Установка принтера и плоттера

Все настройки Автокада, в том числе установка принтера и плоттера, осуществляются в пункте 5 Главного Меню - Configure AutoCAD (Настроить Автокад). При выборе этого пункта Автокад сначала выводит на экран текущую конфигурацию системы:

```
Configure AutoCAD

Current AutoCAD configuration

  Video display:    IBM Enhanced Graphics Adapter
  Digitizer:        Microsoft Serial or Bus Mouse
  Plotter:          None
  Port: Asynchronous Communications Adapter COM1 at address 3F8 (hex)
  Printer plotter:  None

Press RETURN to continue:
```

```
Текущая конфигурация Автокада

  Монитор:          IBM Enhanced Graphics Adapter
  Дигитайзер:       Microsoft Serial or Bus Mouse
  Плоттер:          Отсутствует
  Порт: Асинхронный коммуникационный адаптер COM1 по адресу 3F8 (hex)
  Принтер:          Отсутствует

Для продолжения нажмите RETURN:
```

После нажатия RETURN вы попадаете в меню настройки Автокада, из которого следует выбрать пункт настройки принтера или плоттера (мы рассмотрим настройку плоттера, поскольку настройка принтера производится аналогично):

```
Configuration menu
 0.  Exit to Main Menu
 1.  Show current configuration
 2.  Allow detailed configuration
 3.  Configure video display
 4.  Configure digitizer
 5.  Configure plotter
 6.  Configure printer plotter
 7.  Configure system console
 8.  Configure operating parameters
Enter selection <0>: 5
Your current plotter is: None
Do you want to select a different one? <N>:Y
```

```
Меню настройки
 0.  Выход в Главное Меню
 1.  Текущая конфигурация
 2.  Настройка портов ввода/вывода
 3.  Настройка монитора
 4.  Настройка дигитайзера
 5.  Настройка плоттера
 6.  Настройка принтера
 7.  Настройка консоли
 8.  Настройка рабочих параметров
Ваш выбор <0>: 5
Ваш плоттер: Отсутствует
Хотите ли вы выбрать другой? <Н>:Д
```

Если вы устанавливаете другое устройство, Автокад выводит на экран список доступных в системе плоттеров (принтеров):

```
Available plotters:
   1.  None
   2.  ADI plotter
   3.  Alpha Merics
   4.  Calcomp 906/907/PCI
   5.  Canon Laser Printer LPB-8
   6.  Cordata Laser Printer
   7.  Gould Colorwriter
   8.  Hewlett-Packard
   9.  Houston Instrument
  10.  IBM 7300 Series
  11.  IBM Personal Pageprinter
  12.  IOLINE LP 3700
Select device number or ? to repeat list <1>:8
```

```
Имеющиеся плоттеры:
   1. Отсутствует
   2. ADI plotter
   3. Alpha Merics
   4. Calcomp 906/907/PCI
   5. Canon Laser Printer LPB-8
   6. Cordata Laser Printer
   7. Gould Colorwriter
   8. Hewlett-Packard
   9. 'Houston Instrument
  10. IBM 7300 Series
  11. IBM Personal Pageprinter
  12. IOLINE LP 3700
Введите номер устройства или ?, чтобы повторить список <1>: 8
```

Для каждого типа периферийного устройства поддерживается, как правило, несколько различных моделей (например, для плоттера Hewlett Packard поддерживается 12 моделей). После выбора типа устройства Автокад может предложить вам выбрать одну из нескольких выпускаемых моделей:

```
Supported models:
   1. 7220
   2. 7470
   3. 7475
   4. 7550
   5. 7580
   6. 7585
   7. 7586
   8. 7600 240D
   9. 7600 240E
  10. ColorPro
  11. DraftPro
  12. DraftMaster
Enter selection, 1 to 12 <1>: 3
```

```
Поддерживаемые модели:
   1. 7220
   2. 7470
   3. 7475
   4. 7550
   5. 7580
   6. 7585
   7. 7586
   8. 7600 240D
   9. 7600 240E
  10. ColorPro
  11. DraftPro
  12. DraftMaster
Ваш выбор, 1 к 12 <1>: 3
```

Автокад поддерживает 15 типов принтеров и 24 типа плоттера (поддерживаются только те принтеры, которые способны работать в графическом режиме).

Для каждого из них в Автокаде имеется свой драйвер (файл с расширением .drv). Прежде чем предложить вам выбрать устройство, Автокад проверяет, файлы-драйверы каких моделей имеются на диске, и выводит на экран имена только тех устройств, драйверы которых имеются в наличии. Если в списке доступных Автокаду устройств вы не нашли "своего" плоттера (принтера), то следует выяснить, с каким устройством из списка он совместим. Эта информация должна быть приведена в документации на ваш плоттер (принтер). Если и там вы ничего не найдете, следует обратиться к специалисту.

Дальнейшие вопросы служат для определения параметров и значений по умолчанию выбранного вами устройства и определяются его типом.

7.1.1. Вывод на плоттер при наличии одного порта

В Автокаде одновременно могут быть установлены один плоттер и один принтер. Принтер подключается через параллельный порт, а плоттер может подключаться как к параллельному, так и к последовательному порту - в зависимости от модели. После того, как вы указали Автокаду тип и модель устройства, Автокад определяет, как это устройство должно подключаться к компьютеру и, если возможны варианты, просит указать номер порта.

Если вы используете "мышь", которая подключается не к шине, а к последовательному порту, и если у вас только один порт, то плоттер вам придется подключить к тому же порту. При этом перед выводом чертежа вы должны будете отключить от порта "мышь" и подключить плоттер. По завершении вывода чертежа Автокад делает паузу, которая позволяет снова подсоединить устройство указания. Такой метод не может быть использован для работы с принтером, поскольку Автокад обращается к принтеру через операционную систему.

```
Connects to Asynchronous Communications Adapter port.
Standard ports are:
   COM1
   COM2
Enter port name, or address in hexadecimal <COM1>:com2
```

```
Подсоединение к порту Асинхронный коммуникационный адаптер.
Стандартные порты:
   COM1
   COM2
Введите имя порта, или адрес в шестнадцатеричном (Hex) формате <COM1>:
com2
```

7.1.2. Получение цветных чертежей на одноперьевых плоттерах

Автокад позволяет получать цветные чертежи даже на одноперьевых плоттерах. При установке в системе одноперьевого плоттера Автокад задаст вопрос:

```
Do you want to change pens while plotting? <N>:
```

```
Хотите ли вы чертить в цвете различными перьями? <Н>:
```

Если вы положительно ответите на этот вопрос, при выводе рисунка на плоттер Автокад будет просить вас поменять перо вручную. При этом следует подождать остановки плоттера и заменить соответствующее перо. Для возобновления процесса вывода нажмите клавишу RETURN:

```
Insert pen number 2, color 3 (green).
Press RETURN to continue:
```

```
Установите перо номер 2, цвет 3 (зеленый).
Нажмите RETURN, чтобы продолжить:
```

7.1.3. Калибровка плоттера (принтера)

При выводе чертежей на плоттер и принтер предполагается, что устройство точно откалибровано производителем - отрезок, например, длиной 10 см должен иметь длину на бумаге ровно 10 см. Если это не так, то следует откалибровать устройство. При начальной установке калибровать плоттер (принтер) не нужно, и на следующий вопрос следует отвечать отрицательно:

```
If you have previously measured the lengths of a horizontal
and a vertical line that were plotted to a specific scale,
you may use these measurements to calibrate your plotter.
Would you like to calibrate your plotter? <N>: n
```

```
Если вы уже измерили длину горизонтального и вертикального
отрезков, вычерченных на плоттере в определенном масштабе,
то вы можете использовать эти данные для калибровки плоттера.
Хотите ли вы откалибровать ваш плоттер? <Н>: н
```

При этом дальнейшие вопросы этого раздела будут опущены.

Позднее, если вы захотите откалибровать устройство, следует вывести на бумагу горизонтальный и вертикальный отрезки известной длины (скажем, 10 см) и измерить их длину на бумаге (предположим, что эти длины оказались равными, соответственно 11 см и 12 см). Эти данные потребуются для ответа на запросы, которые выдаются в том случае, если вы ответили утвердительно на предыдущий вопрос:

```
Enter measured length of horizontal line <1.0000>: 11
Enter correct length of horizontal line <1.0000>: 10
Enter measured length of vertical line <1.0000>: 12
Enter correct length of vertical line <1.0000>: 10
```

```
Введите измеренную длину горизонтального отрезка <1.0000>: 11
Введите действительную длину горизонтального отрезка <1. 00>: 10
Введите измеренную длину вертикального отрезка <1.0000>: 12
Введите действительную длину вертикального отрезка <1.0000>: 10
```

При перекалибровке устройства следует отвечать отрицательно на вопрос:

```
Enter selection <0>: 5
Your current plotter is: Hewlett-Packard
Do you want to select a different one? <N>: N
```

```
Ваш выбор <0>: 5
Ваш плоттер: Hewlett-Packard
Хотите ли вы выбрать другой? <Н>: Н
```

Если вы ответили на вопрос утвердительно, а затем выбрали при установке ту же модель, то Автокад сбросит все калибровки и установит калибровку по умолчанию.

После калибровки устройства в ответ на все последующие вопросы следует нажимать клавишу RETURN - дальнейшие параметры вы сможете установить из графического редактора. Для выхода из меню настроек выберите пункт 1. По умолчанию Автокад считает, что вы произвели установку правильно, и записывает произведенные изменения в системе. Для отмены установок следует ответить отрицательно на запрос:

```
If you answer N to the following question, all configuration
changes you have just made will be discarded.

Keep configuration changes? <Y>:
```

```
Если на следующий вопрос вы ответите "Н", то все
изменения, внесенные в конфигурацию, будут проигнорированы.

Сохранить новую конфигурацию? <Д>:
```

Настройка конфигурации Автокада на этом завершается.

7.2. Вывод чертежа на принтер и плоттер

Для вывода чертежа на плоттер используется пункт 3 Главного Меню или команда PLOT (ЧЕРТИ) графического редактора Автокада. Для вывода чертежа на принтер следует использовать пункт 4 Главного Меню или команду PRPLOT (ПЕЧАТАЙ) графического редактора Автокада. В дальнейшем мы не будем делать различия между принтером и плоттером и все сказанное ниже относится к обоим устройствам.

При выборе устройства, на которое вы будете выводить чертеж, следует иметь в виду, что качество плоттера заведомо выше матричного принтера, хотя

почти всегда принтер работает быстрее, чем плоттер (исключение составляют лазерные принтеры, тоже долго обрабатывающие графическое изображение). В качестве примера скажем, что все иллюстрации в данной книге выполнены с помощью Автокада на плоттере фирмы Hewlett Packard (модель 7475).

Команды PLOT (ЧЕРТИ) и PRPLOT (ПЕЧАТАЙ) находятся как в экранном меню графического редактора, так и в падающем меню Files (Файлы).

При использовании пунктов 3 и 4 Главного Меню Автокад запрашивает имя файла чертежа, а при использовании команд графического редактора выводится текущий рисунок. В остальном эти два способа получения твердых копий абсолютно идентичны.

В ответ на первый запрос при выводе чертежа на печать требуется определить область чертежа, выводимую на печать:

```
What to plot - Display, Extents, Limits, View or Window <D>:
```

```
Что чертить - Экран, Границы, Лимиты, Вид или Рамку <Э>:
```

При включенной перспективе опция Window (Рамка) не появляется.

Display (Экран)
Выводится текущий вид. При работе из Главного Меню будет выведен вид, который был перед последней командой SAVE (СОХРАНИ) или END (КОНЕЦ)

Extents (Границы)
Будет выведена минимальная прямоугольная область, содержащая все объекты чертежа (можно посмотреть командой STATUS (СТАТУС). Поскольку эта область определяется не после каждой команды редактирования, а только в момент выполнения команд ZOOM ALL (ПОКАЖИ Все) ZOOM Extents (ПОКАЖИ Границы), перед выводом с опцией Extents (Границы) следует выполнить одну из этих команд

Limits (Лимиты)
Выводятся область лимитов. Если текущий вид не является видом в плане, то выводятся границы рисунка (предыдущая опция)

View (Вид)
Выводится поименованный вид

Window (Рамка)
Выводится область рисунка, ограниченная рамкой, определяемая двумя точками. При работе с командами PLOT (ЧЕРТИ) и PRPLOT (ПЕЧАТАЙ) рамку можно задавать на экране устройством указания. Эта опция не работает для перспективных проекций

После определения области вывода чертежа Автокад выдает на экран текущие значения основных параметров вывода и запрашивает их подтверждение:

```
Plot will NOT be written to a selected file
Sizes are in Inches
Plot origin is at (0.00,0.00)
Plotting area is 7.99 wide by 11.00 high (MAX size)
Plot is NOT rotated 90 degrees
Hidden lines will NOT be removed
Plot will be scaled to fit available area

Do you want to change anything? <N>:
```

```
Чертеж не будет записан в отдельный файл
Размеры в мм
Точка начала отсчета на чертеже (0.00,0.00)
Область черчения. Ширина - 210.00; Высота -280.00 (размер ПОЛЬ)
Чертеж НЕ поворачивается на 90 градусов
Скрытые линии НЕ будут удалены
Масштаб чертежа НЕ будет изменен в соответствии с областью черчения

Хотите что-либо изменить? <Н>:
```

Подтверждением использования значений по умолчанию является нажатие клавиши RETURN. Чтобы изменить один или несколько параметров вывода, следует ответить "Y" ("Д").

Если плоттер многоперьевой и обеспечивает аппаратное вычерчивание различными типами линий или имеет программно-управляемую скорость перемещения пера, Автокад вначале запросит, нужно ли модифицировать эти параметры. Для многоперьевых плоттеров рекомендуется проверять соответствие номеров и цветов перьев цветам Автокада.

На производстве чертежи выполняются в черно-белом варианте, что обусловлено технологией репрографирования. И все же использование различных цветов на экране в чертеже Автокада может оказаться очень удобным. Вы можете, например, все сплошные основные линии чертежа при работе с Автокадом отрисовывать красным цветом, при установке параметров плоттера назначить красный цвет, скажем перу номер 2, а при выводе на плоттер вставить в держатель номер 2 перо нужной толщины. Таким образом вы избавите себя от необходимости обводить чертеж и значительно ускорите процесс вывода (на некоторых плоттерах широкие полилинии рисуются медленно и недостаточно оптимально).

Не следует путать тип линии плоттера с типом линии Автокада. Если вы при настройке плоттера назначите, например, пунктирный тип линии перу номер 2, то все линии этого пера будут отрисовываться пунктирным типом линии, вне зависимости от типа линии, определяемого Автокадом (вы можете, конечно, пользоваться аппаратными типами линий вашего плоттера, если они вас устраивают).

Рассмотрим некоторые параметры, которые можно установить при выводе чертежа на бумагу (самоочевидные параметры будем опускать).

7.2.1. Вывод в файл

При выводе чертежа на бумагу Автокад просматривает графическую базу данных чертежа и в указанной области вывода каждый раз заново формирует последовательность символов (кодов), которые при передаче на устройство собственно управляют принтером (плоттером). Если вы хотите несколько раз вывести на бумагу сложный чертеж или вид с удалением скрытых линий, то процедура формирования документа будет каждый раз отнимать много времени.

В Автокаде имеется возможность вместо передачи управляющих кодов на плоттер (принтер) записывать их в файл. Такой файл представляет собой готовый "пакет" кодов управления устройством и может быть распечатан или вычерчен на плоттере средствами ДОС - для этого надо просто скопировать такой файл на соответствующее устройство. Это средство позволяет создавать на Автокаде готовые чертежные документы и многократно получать абсолютно идентичные копии чертежей (заметим, что возможность редактирования таких документов абсолютно исключена).

При выводе чертежа в файл Автокад просит указать имя файла. Имя следует вводить латинскими буквами. Если не будет указано расширение, Автокад присвоит файлам для принтера стандартное расширение .lst, а файлам для плоттера - стандартное расширение .plt.

7.2.2. Точка начала отсчета и формат чертежа

Для графопостроителей точкой начала отсчета чертежа является левый нижний угол бумаги, а для принтеров - левый верхний угол бумаги. Начальную точку можно сместить. При этом в левом и нижнем краях бумаги появится пустое поле и уменьшиться площадь для вычерчивания, т.к. формат чертежа измеряется от начальной точки. Уменьшение поля черчения отражается в таблице форматов.

Если для принтера нельзя задать формат больший максимального, то для плоттера это возможно (при этом будет выдано соответствующее предупреждение). Существует, правда, абсолютный максимальный формат чертежа, который не может быть превышен при черчении (65535 шагов плоттера). При превышении этого формата чертеж усекается.

7.2.3. Толщина пера

При отработке закрашивания фигур, полос и широких полилиний Автокад столько раз проводит перо, сколько требуется для закраски области пером установленной в Автокаде толщины. Если вы укажете Автокаду толщину пера меньше истинной, то закрашивание будет производиться гуще.

Если требуется получить документ с точностью не менее половины толщины пера (например, при получении фотошаблонов), то следует положительно ответить на запрос:

```
Adjust area fill boundaries for pen width?<N>:
```

```
Настраивать границы областей закрашивания под толщину пера?<Н>:
```

Тогда при выводе широких полилиний, полос или фигур их границы будут смещены внутрь на половину толщины пера. Поскольку обычно такая точность не требуется, по умолчанию эта опция отключена.

7.2.4. Масштаб чертежа

В предыдущих главах мы неоднократно говорили, что в Автокаде можно всегда чертить в масштабе 1:1, а масштабирование осуществлять при выводе чертежа на бумагу. Предположим, что размеры чертежа в условных единицах Автокада составляют 297x210 метров. Тогда при выводе чертежа на плоттер для формата А4 (297x210 мм) необходимо указать масштаб 1:1000:

```
Specify scale by entering:
Plotted Millimeters=Drawing Units or Fit or ? <F>:1:1000
```

```
Укажите масштаб, введя:
Число мм=числу единиц на рисунке, или Вписать, или? <В>:1:1000
```

7.2.5. Подготовка плоттера

Непосредственно перед выводом чертежа на бумагу Автокад выводит на экран размеры области черчения и выдает запрос:

```
Effective plotting area:  7.99 wide by 4.98 high
Position paper in printer.
Press RETURN to continue:
```

```
Область чертежа: ширина - 210.00; высота - 210.00
Установите бумагу на принтер.
Для продолжения нажмите RETURN:
```

В этот момент следует включить плоттер, установить на него бумагу, произвести необходимые начальные установки на самом плоттере и для продолжения нажать клавишу RETURN. Процесс передачи данных на плоттер можно прервать нажатием клавиш CTRL C. Заметим, однако, что буфер устройства, в котором хранятся уже переданные данные, при этом не освобождается и поэтому собственно черчение немедленно не останавливается.

**АО "ДИАЛОГ-МИФИ" -
ОФИЦИАЛЬНЫЙ ДИСТРИБУТОР
ФИРМЫ SYMANTEC CORP.**

АО "ДИАЛОГ-МИФИ", официальный дистрибутор фирмы Symantec Corp., предлагает следующие программные продукты:

Интегрированные пакеты	*Цена*
Q&A 3.0	5350
Q&A Network Pack 3.0	5800
Системы управления планированием	
TIME LINE 4.0	10600
GrandView 2.0	4400
Time Line LAN Pack 1.0	9000
Утилиты	
Norton Antivirus 1.0	1200
Norton BackUp 1.2	1300
Norton Utilities 6.0	2700
Norton Comander 3.0	1300
Just Write 1.0	2200
On Target 1.0	5350

Цены даны в рублях.
Зарегистрированным пользователям предоставляется *скидка* и *гарантируется сопровождение* программных продуктов.

Акционерное общество "ДИАЛОГ-МИФИ 115409 Москва, ул Москворечье, 31, корп. 2
Телефоны 324-83-56, 320-32-11, 324-71-66 Факс (095) 3243055
Р/с 467416 в коммерческом банке "Интерпрогрессбанк" МФО 201508

(setq l5 (
(command "
(command "
(setq l5 (
(command "
(setq l2 (

ГЛАВА 8

ПРОГРАММИРОВАНИЕ В АВТОКАДЕ

Автокад не только графический редактор, но и среда программирования на Автолиспе. Являясь частью Автокада, Автолисп позволяет определять переменные различных типов и передавать их значения командам Автокада при вводе данных. При ответах на запросы команд Автокада можно также использовать выражения Автолиспа, в которых могут выполняться различные арифметические и условные операции над числовыми значениями и значениями определенных переменных.

Кроме очевидных средств выполнения вычислений Автолисп содержит в себе функции, предоставляющие средства доступа к графической базе данных текущего чертежа Автокада. Автолисп позволяет также управлять графическим редактором Автокада. Написание на Автолиспе собственных программ позволяет создавать настроенные на конкретную предметную область применения команды, включающие в себя запросы к пользователю (диалог), возможность выбора по условию из нескольких вариантов или использования значений по умолчанию. Хотя макроопределения, которые вы можете создавать при написании меню Автокада, могут быть достаточно сложными и мощными, без Автолиспа они представляют собой всего лишь наборы стандартных команд Автокада. Включением в макроопределения меню функций Автолиспа можно превратить меню Автокада в интеллектуальное средство автоматизации проектирования.

Хотя в этом кратком руководстве мы подробно не разбираем вопросы связи Автокада с другими системами, нельзя не упомянуть о возможности доступа из Автолиспа к графической базе данных Автокада. Итак, Автолисп позволяет:

- использовать переменные и выражения при ответах на запросы команд Автокада
- создавать различные функции и новые команды Автокада, настраивая и расширяя таким образом графические возможности Автокада
- осуществлять программный доступ (считывание и изменение) к данным, относящимся к объектам рисунка, а также таблицам Автокада, в которых хранится информация о блоках, слоях, видах, стилях и типах линий
- осуществлять программное управление графическим экраном Автокада и вводом/выводом с различных устройств

8.1. Что такое Автолисп

Автолисп - это созданный специально для Автокада диалект Лиспа, полученный в результате изменения языка XLISP.

Язык LISP разработан в 1961 году американским ученым Дж. Маккарти и является родоначальником функциональных языков. Автолисп относится к т.н. функциональным языкам, т.е. к языкам, в основу которых положено понятие

функции (в отличие от обычных операторных языков программирования - Фортрана, Паскаля, Си и др.). К таким языкам относится также, например, язык Пролог. Все вычисления, преобразования и управления программы в функциональных языках осуществляются с помощью элементарных (встроенных) функций или функций, определяемых программистом при написании программы. Программа в целом является суперпозицией некоторых функций и, в свою очередь, может быть использована как функция другими программами.

Язык Лисп идеально подошел для Автокада, "передав" Автолиспу свои очень удобные средства работы с глубоко структурированной информацией. Автолисп - это единственное средство, позволяющее программно работать с объектами Автокада, справочными таблицами, считывать и записывать файлы из Автокада. Автолисп можно считать прямым (и единственным!) окном внутрь Автокада. Кроме того, Автолисп очень прост в изучении и весьма гибок.

8.2. Установка Автолиспа

Автолисп поставляется с каждой копией Автокада, имеющей модуль ADE-3. Для операционных систем PC-DOS/MS-DOS файл acad.ovl на дистрибутивном диске системы Автокад является оверлейным файлом интерпретатора Автолиспа.

Один из дистрибутивных дисков содержит файл с именем readme.doc. Следует ознакомиться с содержимым этого файла, поскольку он содержит все последние изменения и дополнения к документации по Автокаду и Автолиспу.

> *Для работы в операционных системах PC-DOS/MS-DOS имеется дополнительная версия Автолиспа, т.н. расширенный Автолисп, содержащийся в файлах acadlx.ovl, extlisp.exe и remlisp.exe.*

Обычный (не расширенный) Автолисп может работать на любой машине, на которой идет Автокад версии 10 с модулем ADE-3 (требования к оборудованию были перечислены нами в главе 1). Для работы расширенного Автолиспа необходимо также, чтобы компьютер был построен на базе микропроцессора типа Intel 80286 или 80386 (*не* 8086 или 8088). Для работы расширенного Автолиспа желательно, чтобы ваш компьютер имел не менее 512K расширенной памяти типа IBM AT extended memory (*не* типа Lotus/Intel/Microsoft expanded memory), не занятой под другие цели, например под виртуальный (электронный) диск.

При настройке Автокада одним из рабочих параметров является отключение Автолиспа по желанию пользователя. Кроме того, в операционных системах PC-DOS/MS-DOS конфигуратор дает возможность отключения расширенного Автолиспа. (Если вы не работаете с Автолиспом, на этот вопрос можно безболезненно отвечать "Нет".)

В операционных системах PC-DOS/MS-DOS для Автолиспа должна быть выделена определенная часть свободной оперативной памяти, объявленная как память-стек и память-хип, при помощи задания значений переменных среды LISPHEAP и LISPSTACK. В случае одновременной работы с каким-либо другим прикладным пакетом, например Автокад AEC, следует ознакомиться с содержа-

щимися в его описании рекомендациями по установке значений переменных среды LISPHEAP и LISPSTACK. Размером области расширенной памяти, которая будет использоваться расширенным Автолиспом, можно управлять с помощью переменной среды LISPXMEM (если переменная LISPXMEM не задана, расширенный Автолисп использует всю оперативную память, кроме занятой виртуальным диском). Расширенный Автолисп сообщает Автокаду, какую область памяти он использует, тем самым делая эту область памяти недоступной для использования самим Автокадом. Расширенный Автолисп не использует хип.

Расширенный Автолисп действует как самостоятельная программа с именем EXTLISP, которую необходимо запустить перед запуском Автокада. EXTLISP является резидентной программой, которая может размещаться в расширенной памяти, и связывается с Автокадом через оверлейный файл acadxl.ovl. Для автоматического запуска программы EXTLISP ее вызов можно поместить в файл настроек системы Автокад, acad.bat или даже (если вы ни с чем, кроме Автокада, на машине не работаете и вас устраивает, что EXTLISP будет автоматически загружаться в память при включении компьютера) в файл autoexec.bat.

Программа EXTLISP может быть удалена из памяти после окончания сеанса работы с Автокадом. Для этого в ответ на подсказку ДОС следует ввести команду REMLISP, вызывающую на выполнение программу с тем же именем. При этом расширенный Автолисп выгружается из оперативной памяти.

8.3. Работа с Автолиспом с командной строки Автокада

К Автолиспу можно обращаться прямо с командной строки Автокада. Когда вы вводите какую-то группу букв, интерпретатор командной строки пытается найти введенное вами слово в списке команд Автокада. Если введенного слова нет в списке команд, то выдается сообщение об ошибке. Напомним лишний раз, что команда Автокада не может содержать пробелов и нажатие клавиши ПРОБЕЛ воспринимается так же, как нажатие клавиши RETURN.

Если первым символом вы введете в командной строке круглую открывающую скобку, то интерпретатор командной строки Автокада переходит в специальный режим - режим ввода выражения Автолиспа. Выход из этого режима осуществляется при вводе скобки, закрывающей вводимое выражение (оно может быть сложным и содержать вложенные выражения). Работа с клавиатурой в этом режиме имеет несколько особенностей. Так, нажатие клавиши ПРОБЕЛ не рассматривается больше как терминатор ввода, поскольку любое выражение Автолиспа содержит пробелы. Кроме того, выражение Автолиспа может занимать несколько строчек и поэтому передается вычислителю Автолиспа только тогда, когда он завершит ввод последней правой скобки. Поэтому, если при вводе вами какого-то выражения Автокад выдаст сообщение "1>", не беспокойтесь - это просто подсказка, которая говорит о том, что для правильного завершения ввода функции или выражения необходима еще одна правая круглая скобка - ")". Вообще, ответ Автолиспа типа "n>" (n - целое число) говорит о том, что интерпретатору Автолиспа требуется n закрывающих скобок или кавычек для строковых

констант (каждая открывающая скобка или кавычка должна иметь свою закрывающую скобку или кавычку). Введите скобку, нажмите RETURN, затем CTRL C и попытайтесь набрать пример еще раз, внимательно следя за балансом скобок и кавычек. Если добавление скобок в ответ на этот запрос не помогает, введите кавычку, а после нее соответствующее количество скобок, потому что если вы забыли кавычку, то все последующие данные (в том числе и круглые скобки) интерпретируются как текст и не распознаются вычислителем Автолиспа. Самое трудное в Автолиспе - это отслеживать правильные пары скобок и кавычек ("") для строковых констант.

Повторяя приводимые ниже упражнения на Автокаде, вы не увидите на экране строки "Автолисп возвращает:", однако все, что написано после этой строки, должно появиться на экране. Изучая настоящее руководство, "обкатывайте" не только примеры, но и новые для вас функции - только таким способом можно обогатить свой арсенал.

Мы уже говорили выше, что Автолисп может быть полезен не только как средство программирования, но и просто как калькулятор, которым можно воспользоваться в любой команде Автокада в ответ на запрос ввести какое-то значение или координаты точки. Парадокс, однако, заключается в том, что работа с этим "калькулятором" предполагает достаточно свободное владение Автолиспом и умение на нем программировать. Исходя из этого, мы надеемся, что, изучив эту главу и написав несколько небольших программ, вы начнете применять Автолисп в своей повседневной деятельности и самостоятельно найдете удобные для себя приемы и методы использования Автолиспа при интерактивной работе с графическим редактором Автокада.

Попробуем автоматизировать отрисовку простого закрашенного кольца. При помощи Автолиспа мы создадим новую команду Автокада, которую назовем, скажем, "ОТВЕРСТИЕ":

```
Command: (defun C:HOLE () (setq rad (/ (getdist "\nDiameter:
4>") 2)) (while T (command "Donut" pause rad)))
Автолисп возвращает: C:HOLE
```

```
Команда: (defun C:ОТВЕРСТИЕ () (setq rad (/ (getdist "\nДиаметр:
4>") 2)) (while T (command "Кольцо" pause rad)))
Автолисп возвращает: (C:ОТВЕРСТИЕ)
```

Итак, буквально двумя строчками мы добавили в Автокад новую (свою!) команду. Попытаемся выполнить ее:

```
Command: HOLE
Diameter: 1
CIRCLE 3P/2P/TTR/<Center point>: 10,10
Diameter/<radius>: 0.5
Command: CIRCLE 3P/2P/TTR/<Center point>: CTRL C
```

```
Команда: ОТВЕРСТИЕ
Диаметр: 1
КРУГ 3Т/2Т/ККР/<Центр>: 10,10
Диаметр/<радиус>: 0.5
Команда: КРУГ 3Т/2Т/ККР/<Центр>: CTRL C
```

Когда вы набирали эту программу, то, наверное, столкнулись с тем, что при ошибке приходится начинать все сначала. На самом деле так программы на Автолиспе, конечно, не пишутся. Позже мы опишем некоторые приемы работы с Автолиспом в среде Автокада.

8.4. Типы данных и переменные

Автолисп использует переменные. Переменная - это просто метка, используемая для обращения к изменяемой величине. Вам приходится сталкиваться с переменными каждый день. Например, получаемая вами ЗАРПЛАТА формируется из ОКЛАДА, из которого вычитается НАЛОГ и добавляется ПРЕМИЯ. ЗАРПЛАТА, ОКЛАД, НАЛОГ и ПРЕМИЯ - переменные.

Автолисп позволяет поставить в соответствие имя переменной и ее значение, которое будет потом использоваться в Автолиспе в выражениях и функциях.

8.4.1. Типы данных Автолиспа

Благодаря Автолиспу вы имеете доступ к стандартным системным переменным (о них рассказывается в п. 8.4.3) и можете создавать переменные Автолиспа. Как и любой другой язык программирования, Автолисп при работе с переменной должен знать, к какому типу данных она относится: типом данных определяются операции, которые можно проделать с переменной (например, одно число поделить на другое число, но нельзя поделить одну строку на другую).

Понятие типа данных - ключевое понятие в программировании. "Качество" языка программирования, т.е. его потенциальные возможности, определяется во многом тем, какие типы данных им поддерживаются. Перечислим сначала основные типы данных, которые есть в любом языке программирования, в том числе и в Автолиспе:

- строковые переменные
- целые переменные
- действительные переменные

Строковые переменные могут содержать в себе различные текстовые константы, т.е. совокупности букв (и символов), заключенных в кавычки, например "Введите число <12.80>:". Кавычки говорят Автолиспу, что данную совокупность символов следует рассматривать как текстовую константу. Если системная переменная является строковой, то это значит, что в ней может храниться только строковая константа (строковая константа может, конечно, с нашей точки зрения являться числом, например "56.88", однако Автолисп не понимает, что это число и работает с такой константой просто как с группой символов).

Целые переменные - это положительные или отрицательные целые числа (без дробей и десятичной точки). Для определения того, включен какой-то режим или нет, Автокад часто использует значения 0 и 1, например если режим ORTHO (ОРТО) включен, то системная переменная ORTHOMODE установлена в единицу. Заметим, что целые числа представлены в машине двумя байтами и не могут поэтому выходить за диапазон (-32 768, 32 767).

Действительные переменные - это положительные или отрицательные числа с десятичной точкой. В отличие от многих других языков программирования в Автолиспе не разрешается начинать или заканчивать число десятичной точкой. Это связано с существованием в Автолиспе специфического типа данных - точечной пары, разделителем которой является точка. Поэтому, если значение меньше единицы, вы должны явно указать 0 перед десятичной точкой (0.123), иначе Автолисп воспримет ошибку (error: invalid dotted pair (ошибка: неверная точечная пара). В качестве примеров действительных системных переменных можно указать переменные FILLETRAD, LTSCALE и AREA. Диапазон представления действительных чисел зависит от типа вашего компьютера и в любом случае достаточно велик. Правильно записанными действительными числами являются числа 2.12, 3.11592652543, -92722.121344 и 1.23544E+17.

Требование принадлежности переменной к какому-либо типу данных - бич для программиста и вообще требование, чуждое обыденному человеческому мышлению. Ведь тип данных - чистая абстракция, соглашение-уступка вычислительной технике, которая пока еще не слишком-то совершенна. Человек не оперирует понятием "тип данных", для него данные не имеют фиксированного типа (как говорится, с какой точки зрения посмотреть), а скорее имеют все типы одновременно: написанное на бумаге число - это одновременно и группа символов, и число, и вообще все что угодно, например художественное произведение абстрактной живописи. Мы уже говорили, что Автолисп был создан как язык, предназначенный для разработок в области искусственного интеллекта, и проблема типа данных в нем в какой-то степени решена. Первое удобство заключается в том, что переменные в Автолиспе не надо описывать - тип данных Автолисп определяет автоматически при присвоении переменной значения (те, кто программировал на Фортране, оценят это). Второе принципиальное отличие Автолиспа от, например, Бейсика или Паскаля состоит в том, что в Автолиспе есть типы данных нового, более высокого уровня. Эти типы данных называются *списком* и *атомом*. Попытаемся дать определения этих типов данных.

Список - это некий (заранее не определено какой) набор элементов списка, заключенный в круглые скобки. Элементом списка может быть, в свою очередь, список. Примеры списков:

```
(12.6 45.7 77.8)
("CAT" "EATS" "MOUSE")
(1 (12 6.78) ("Иван" "Грозный"))
(* 2 5)
(setq point '(1.0 5.0 6.7))
```

Неделимый элемент списка называется *атомом*. Примеры атомов:

```
12.45
"True Love"
()
```

Как уже ясно из примеров, списки - идеальное средство работы со сложным образом организованной геометрической информацией. Точка, например, не что иное, как список трех действительных чисел. Отрезок - список двух точек. Полилиния - список многих вершин (как видим, полилиния - геометрический объект, органически вытекающий из "идеологии" описания данных Автолиспа). Практически любую структурированную информацию можно представить в виде списка, что и делается в Автолиспе. Более того, любая определяемая пользователем функция Автолиспа (это соответствует подпрограмме в операторных языках программирования), вообще говоря, тоже список.

Теперь мы можем перечислить типы данных Автолиспа:

INT	Целые величины
REAL	Числа с плавающей точкой
STR	Строковые константы
FILE	Дескрипторы файлов
SYM	Символы
LIST	Списки и функции пользователя
SUBR	Внутренние функции Автолиспа
ENAME	Имена примитивов Автокада
PICKSET	Наборы примитивов Автокада
PAGETB	Таблицы диспетчера страниц

Итак, переменная в Автолиспе может быть атомом (целого, вещественного или строкового типа), а может быть списком. Забегая вперед, приведем пример, в котором формируется список координат точки:

```
Команда: (setq a '(100 200 300))
Автолисп возвращает: (100 200 300)
```

В этом примере создается переменная A, которая является списком. В Автолиспе имеются мощные средства работы со списками. Позже (см. п. 8.16) мы рассмотрим средства работы со списками Автолиспа более подробно.

Опять-таки забегая вперед, скажем, что в Автолиспе всегда можно определить тип данных объекта - для этого существует функция Автолиспа TYPE:

(type объект) Возвращает (как символ) тип данных **объекта**.

8.4.2. Переменные Автолиспа

Присваивая значение символу (идентификатору, имени переменной), вы автоматически создаете переменную (при этом Автолисп фиксирует ее тип). Так, в примере HOLE мы присваивали идентификатору RAD численное значение расстояния от одной точки до другой. То же относится и к макроопределениям.

При создании переменной (а точнее, при присвоении идентификатору переменной какого-то значения) Автолисп автоматически назначает ей тип. Переменные Автолиспа совершенно независимы от системных переменных Автокада и могут повторять их по именам. При каждом обращении к переменной Автолисп использует то значение переменной, которое было установлено последним. Имя переменной может состоять из любых печатных символов (в т.ч. и цифр), однако не должно включать в себя зарезервированные Автолиспом символы, поскольку их Автолисп интерпретирует специальным образом. *Не рекомендуется* использовать для имен переменных следующие символы:

Зарезервированные символы:	. ' " ; ( ) или пробел
Знаки операций Автолиспа:	- * = > < + - /
"Плохие" символы:	? ' ! \ ^
	а также управляющие символы

Нельзя также использовать в качестве имен переменных имена функций Автолиспа. Все имена функций Автолиспа, а также созданные пользователем имена переменных хранятся в переменной Автолиспа ATOMLIST. Выведите на экран содержимое переменной ATOMLIST и вы увидите, какие имена вы не можете использовать в своей программе. (Это можно сделать вводом с командной строки Автокада команды !ATOMLIST.) Некоторые из этих идентификаторов используются исключительно вычислителем Автолиспа и не могут быть вами использованы, остальные описаны в "Руководстве по программированию на Автолиспе", к которому мы рекомендуем вам обращаться за справочной информацией о встроенных функциях Автолиспа.

Заметим, что Автолисп не различает малые и большие буквы. Поскольку имя переменной длиной более шести символов требует для хранения больше памяти, мы не рекомендуем вам превышать этот предел. Не следует также начинать имя переменной с цифры.

Имена переменных

неверные	*верные*
123 - целое число	PT1
10.5 - действительная константа	txt
ANGLE - конфликтует с функцией ANGLE Автолиспа	ANGL
A(1) - содержит запрещенные символы	A_1
OLD SUM - содержит пробел	OLD_SUM

8.4.3. Системные переменные Автокада

Среда графического редактора Автокада позволяет изменять режимы, переопределять лимиты, менять размеры прицелов объектной привязки и выбора объектов, получать информацию о различных установках графического редактора.

Установленные вами режимы сохраняются до тех пор, пока вы их не переопределите. Сохраняются не только общие настройки (например, размер прицела выбора объектов), но и настройки, относящиеся к способу работы с чертежом (единицы измерения, режим отрисовки маркеров и т.п.).

Все настройки графического редактора управляются т.н. системными переменными Автокада. Когда вы изменяете установки, например включаете или выключаете режим ORTHO (ОРТО), изменяете режим привязки и т.п., Автокад сохраняет только что установленный режим в соответствующей системной переменной. Системные переменные, таким образом, формируются в процессе работы с графическим редактором. Фиксировано не только имя, но и тип системной переменной (в вещественную переменную, например, нельзя записать строку).

Системные переменные делятся на модифицируемые пользователем и защищенные. Защищенную переменную нельзя изменять ни программно, ни при помощи команды SETVAR (УСТПЕРЕМ). Те же переменные, которые не имеют такой защиты, могут быть вами изменены прямо в процессе выполнения той или иной команды путем использования команды 'SETVAR ('УСТПЕРЕМ) в прозрачном режиме. Меняя значения системных переменных, вы можете управлять многими (правда, не всеми) режимами графического редактора Автокада. Системные переменные также можно разделить по месту их хранения: некоторые из них записываются по окончании сеанса редактирования в файл конфигурации системы, остальные (таких большинство) - в текущий чертеж. Таким образом, файл чертежа Автокада - это не просто описание геометрических объектов, а еще и полное описание режимов работы и настроек Автокада. Некоторые переменные никуда не записываются, потому что никогда не меняются - например, номер версии Автокада (переменная ACADVER).

При помощи команды SETVAR (УСТПЕРЕМ) вы можете в любой момент посмотреть текущие установки системных переменных:

```
Command: SETVAR
Variable name or ?:?
```

```
Команда: УСТПЕРЕМ
Имя переменной или ?:?
```

Системные переменные Автокада могут быть считаны или установлены из Автолиспа, для этого существуют встроенные функции Автолиспа - GETVAR и SETVAR. (Не надо путать функцию SETVAR Автолиспа и команду SETVAR Автокада - это совершенно разные вещи.) Для того чтобы в программах на Автолиспе не изменять установки системных переменных, используемые Автокадом по умолчанию, необходимо считывать (при помощи функции GETVAR) значение изменяемой системной переменной, сохранять старое значение и затем (при помощи функции SETVAR) восстанавливать его перед выходом из программы. Если вы внимательно посмотрите на текст нашего примера HOLE (ОТВЕРСТИЕ), то увидите, что такая операция там проделана. В макроопреде-

лениях также следует восстанавливать старые значения системных переменных - это хороший стиль программирования на Автолиспе, он поможет вам застраховаться от ошибок.

Список системных переменных приведен в приложении А.

8.5. Выражения Автолиспа

Если с точки зрения Автолиспа все, что заключено в круглые скобки, считается списком, то как Автолисп отличает данные от управляющих конструкций - объект, который оперирует с данными, от самих данных? Дело в том, что любая управляющая конструкция Автолиспа содержит объекты специального типа - функции. Функции Автолиспа могут быть встроенными или определенными пользователем. Встроенная функция Автолиспа - это тип данных, являющийся на самом деле ссылкой на некую подпрограмму, которая выполняет над передаваемыми параметрами некоторые действия. Функция, определенная пользователем, - это обычный сложный список, но все равно он в конечном итоге содержит встроенные функции Автолиспа. Для того чтобы Автолиспу было проще искать среди элементов списка функции, их принято писать на первом месте.

Итак, список, в котором первым элементом является функция, будем называть выражением. Любая функция Автолиспа состоит из выражений и сама является выражением. Перечислим основные свойства выражений (многие из этих свойств обусловлены просто тем, что любое выражение - это список).

- Каждая открывающая круглая скобка должна иметь закрывающую
- Сразу после открывающей круглой скобки должен стоять идентификатор операции, выполняемой при вычислении выражения (имя функции)
- Каждое выражение вычисляется (выполняется), и результат возвращается. Результатом может быть ноль (nil) или результат вычисления последнего подвыражения
- С логической точки зрения любое возвращаемое значение либо истинно, либо ложно. Если значение выражения вычислено быть не может и возвращается ноль (nil), то оно считается ложным. Если значение вычисляется, то выражение считается истинным - не-ноль (non-nil)

Выражение Автолиспа имеет вид:

```
(функция аргумент1 аргумент2 ... аргументN)
```

Здесь функция - имя операции, которая должна быть выполнена. К операциям относятся в том числе и арифметические операции (сложение, вычитание, умножение, деление), а также операция присвоения. Аргументы представляют собой средство передачи значений функции. Аргументами могут быть переменные, константы или выражения. Аргументов функции может быть больше двух. Например, операция перемножения трех чисел запишется так: (* 2 5 8).

Аргумент может быть обязательным и необязательным. Если при создании функции было определено, что ее аргументы являются обязательными, то Автолисп выдаст ошибку при попытке вызвать функцию без указания этого аргумен-

та. Однако аргументы могут быть и необязательными, в этом случае их при вызове функции можно не передавать. Следует также внимательно следить за типом аргументов: если функция производит операции над целыми числами, то ей нельзя передать строку, и наоборот. Иначе говоря, если функция (или команда Автокада) должна принять в качестве аргумента строку, все символы, введенные в ответ на запрос, считаются элементами строки a priori. На первый взгляд это не позволяет передать Автокаду выражение Автолиспа в ответ на запрос, требующий ввода строки (например, при использовании команды TEXT). Указанную трудность помогает преодолеть системная переменная Автокада TEXTEVAL. Установка этой переменной в 1 (оценка текста включена) приводит к тому, что Автокад передает выражение в Автолисп и печатает возвращаемый результат, а не само выражение. Другими словами, если оценка текста включена, то в ответ на запрос Автокада можно вводить не только сам ответ, но и выражение на Автолиспе для его вычисления, используя круглые скобки или восклицательный знак.

Выражение анализируется Автолиспом слева направо, пока не встретится закрывающая или открывающая скобка. Если встречается закрывающая скобка, то Автолисп завершает анализ выражения, выполняет функцию и передает ее значение на более старший уровень вложенности или в Автокад. Если встречается открывающая скобка, Автолисп переходит к анализу выражения более младшего уровня вложенности и, пока не завершит его анализ, не перейдет к дальнейшему анализу выражения предыдущего уровня. Таким образом, порядок анализа выражений в Автолиспе задан жестко раз и навсегда, что исключает путаницу.

8.6. Функции присвоения

Присвоение переменным значений осуществляется в Автолиспе не так, как в других языках программирования. Для присвоения переменным значений в Автолиспе имеется специальная функция SETQ, которая присваивает переменной нужное нам значение, т.е. заносит указанное нами значение в ячейку с именем переменной. Алгебраическая запись присвоения - x=3. В Автолиспе то же действие записывается как (setq x 3). Это связано с тем, что, как уже отмечалось, Автолисп - функциональный язык и в нем нет операторов (в том числе и оператора присвоения), а есть функции, производящие определенные действия над своими параметрами.

После того как вы присвоили некоторой переменной значение, вы можете в любое время посмотреть ее значение с командной строки Автокада. Восклицательный знак "!", введенный с командной строки, сообщает интерпретатору командной строки Автокада, что следующий за ним идентификатор является именем переменной Автолиспа. Встретив восклицательный знак, интерпретатор командной строки передает следующее за ним имя Автолиспу, который возвращает обратно значение переменной с этим именем.

Что такое "SET" - понятно: "положить", а что означает "setq"? Загадочная буква "q" в функции присвоения "setq" происходит от сокращения английского слова "quote" - букв. "цитировать". Функция SETQ является мнемоническим со-

кращением английского словосочетания "SET by Quote" - "присвоить по ссылке". Прежде чем разобраться, что это означает, нужно понять, как хранятся в машине значения переменной. Дело в том, что переменная и ее значение - не одно и то же. На первый взгляд такое утверждение кажется тривиальным, однако когда мы говорим "переменная", то чаще всего подразумеваем "значение, которое хранится в переменной". Специфика Автолиспа заключается в том, что в переменной может храниться не только значение, но и целое выражение, а если быть точным, то его адрес. Функция SETQ позволяет об этом не задумываться и считает, что мы обращаемся к значению переменной. Однако в Автолиспе есть также функция SET, которая считает, что мы обращаемся к самой переменной. Прежде чем проиллюстрировать сказанное на примерах, отметим, что для обращения к значению переменной в Автолиспе существует специальная функция, **(quote выражение)**, которая возвращает само **выражение**, не выполняя его (фактически возвращает ссылку на **выражение**). Эта функция очень часто используется в Автолиспе и поэтому имеет сокращенную запись: **'выражение**. Так, **(quote A)** эквивалентно **'A**.

Теперь продемонстрируем работу функций SETQ и SET:

(set 'A 5.0) Возвращает 5.0 и присваивает это значение A

Эта запись эквивалентна записи **(setq a 5.0)**

(set 'B '(A + C)) Возвращает выражение (A + C) и присваивает его B

Итак, в переменной хранится выражение! Причем это именно выражение, а не текстовая константа - изменяя значения переменных A и C, мы изменяем тем самым значение переменной B. Эта запись эквивалентна записи

(setq B '(A + C))

Обратившись к специальной функции EVAL ("evaluate" - "оценивать"), можно в любой момент вычислить выражение B. Предположим, что перед выполнением предыдущей строки мы положили **(setq A 3)** и **(setq C 6)**. Тогда

(eval B) Возвращает 9

Если две переменные "вложены" друг в друга, то к младшей можно обращаться через старшую, косвенно. Выполните следующую последовательность функций:

(set 'B 'A) Устанавливает A в B и возвращает A

(set B 640) Устанавливает A в 640 и возвращает 640

8.7. Математические функции

Автолисп включает в себя достаточно широкий набор встроенных функций, позволяющих производить математические вычисления. Аргументами математических функций являются числа, которые могут быть как целыми, так и вещественными. Если все аргументы - целые числа, то результат операции также будет

целым числом, а любая дробная часть будет опущена. Если хотя бы один аргумент - вещественное число, то результат операции будет вещественным числом. (Следует отметить, например, что число 2.0 считается вещественным, поскольку в нем присутствует десятичная точка.) К математическим функциям относятся:

(+ число1 число2 ...)
Возвращает сумму всех аргументов

(- число1 число2 ...)
Вычитает **число2** из **числа1** и возвращает разность. Если задано более двух аргументов, то из первого аргумента вычитается сумма всех остальных. Если задан один аргумент, то он вычитается из нуля (инвертируется его знак)

(* число1 число2 ...)
Возвращает произведение всех чисел

(/ число1 число2 ...)
Делит **число1** на **число2** и возвращает частное. Если задано более двух аргументов, то первое число делится на произведение всех остальных

(1+ число)
Увеличение целого или действительного аргумента на единицу

(1- число)
Уменьшение целого или действительного аргумента на единицу

(atan число1 [число2])
Если не задано **число2**, то возвращает арктангенс переменной **число1** в радианах, область допустимых значений - [pi, -pi] радиан. Если заданы оба числа, то возвращается арктангенс переменной **число1/число2** в радианах. Если **число2** - ноль, то в зависимости от знака переменной **число1** возвращается + или -1.570796 радиан ($+90^0$ или -90^0)

(abs число)
Вычисление абсолютного значения действительного или целого **числа**

(cos число)
Возвращает значение косинуса угла, заданного аргументом в радианах

(exp степень)
Вычисляет значение экспоненциальной функции с основанием e и аргументом, равным числу

(expt основание степень)
Вычисляет значение экспоненциальной функции с указанными **основанием** и **степенью**

(gcd число1 число2)
Возвращает наибольший общий делитель (Greatest Common Denominator) указанных аргументов

(log число)
Возвращает натуральный логарифм аргумента

(max число1 число2 ...)
Возвращает наибольший аргумент. Эта функция в отличие от многих других не изменяет типов чисел

(min число1 число2 ...)
Возвращает наименьший аргумент. Не изменяет типов чисел

(rem число1 число2 ...)
Возвращает остаток (Remainder) от деления переменной **число1** на переменную **число2** (математическая запись: число1 mod число2)

(sin число)
Возвращает значение синуса угла, заданного аргументом в радианах

(sqrt число)
Извлекает квадратный корень из аргумента. Возвращаемый результат всегда вещественный

8.8. Работа со строками, функции преобразования

Среди встроенных функций Автолиспа, работающих со строками, можно найти практически все, что имеется в любом развитом языке высокого уровня:

(itoa целое)
Возвращает результат преобразования **целого** числа в строковую константу

(strcase строка признак)
Переводит все символы аргумента **строка** в нижний регистр, если **признак** не nil, если **признак** опущен или равен nil - в верхний

(substr строка целое1 целое2)
Возвращает подстроку аргумента **строка**, начинающегося с символа **целое1** и длиной **целое2**. Первый символ строки имеет номер 1

(strlen строка)
Возвращает длину в символах аргумента **строка**

(strcat строка1 строка2 ...)
Осуществляет сцепление аргументов **строка**

(atof строка)
Возвращает преобразование **строковой константы** в действительное число

(atoi строка)
Возвращает преобразование **строковой константы** в целое число

(chr число)
Возвращает результат преобразования целого **числа** в символьный код ASCII, причем результатом является строковая константа. Например, (chr 65) воз-

вратит символ (строковую константу) "A" (подобным образом действует функция CHR$ в языке Бейсик)

(ascii строка)

Возвращает преобразование одного символа, указываемого в виде строковой константы, в код ASCII

(angtos угол представление точность)

Преобразует аргумент **угол** в строковую константу. Преобразование осуществляется в соответствии с задаваемыми **представлением** и **точностью**:

Представление	*Формат преобразования*
0	Градусы
1	Градусы/мин/сек
2	Грады
3	Радианы
4	Топографические единицы

Аргумент **точность** - целое число, указывающее число цифр после запятой. Аргументы **представление** и **точность** обращаются к переменным Автокада AUNITS и AUPREC. Если **представление** и **точность** опущены, будут использованы текущие значения этих параметров

В любой момент можно преобразовать целое число в действительное и наоборот:

(fix число)

Возвращает результат преобразования действительного числа в целое. Преобразование осуществляется усечением (отбрасыванием) дробной части

(float число)

Возвращает результат преобразования целого числа в действительное

(type элемент)

Возвращает тип элемента

Используя функцию TYPE (о ней уже говорилось выше), можно определить тип элемента. Следующий пример иллюстрирует использование функции TYPE:

```
(defun isint (a)
   (if (= (type a) 'INT)      ; является ли a целым?
       T                      ; если да - возвращаем ИСТИНУ
       nil                    ; если нет - возвращаем nil
   )
)
```

8.9. Использование функций GET для ввода данных

Для ввода данных с клавиатуры в Автолиспе существует семейство функций GET. Для всех основных типов данных есть своя функция GET. Все аргументы функций этого семейства необязательны. В качестве первого аргумента иногда

выступает необязательная двумерная точка в текущей ПСК. Все функции GET могут иметь в качестве аргумента произвольную строковую константу, в которой может содержаться текст запроса или какая-то подсказка, выводимая при запросе пользователю ввести какие-то данные. Все функции GET ожидают ответа пользователя, т.е. приостанавливают выполнение программы до тех пор, пока не будет осуществлен ввод (нажата клавиша RETURN). Ввод может быть осуществлен как с клавиатуры, так и при помощи устройства указания. Вводя точки с экрана, вы можете захотеть, чтобы в процессе перемещения курсора по экрану Автокад показывал "резиновую" линию. Это позволяют делать практически все функции семейства GET. Все вводимые данные автоматически преобразуются в нужный тип данных. В ответ на запрос функций семейства GET нельзя вводить выражение Автолиспа: это приведет к ошибке и выводу сообщения "Can't reenter AutoLISP" ("Не могу войти в Автолисп повторно"). В макроопределениях меню функции GET нужно выделять обратной косой чертой с тем, чтобы Автокад обеспечил паузы для ввода данных. Вот список этих функций:

(getangle точка "текст запроса-подсказки")

Возвращает угол в радианах между задаваемым пользователем вектором и положительным направлением оси X в текущей плоскости построений. Функция всегда возвращает угол в радианах вне зависимости от текущей установки переменных. Начальная точка вектора может быть определена первым аргументом функции. Вторую точку вектора можно указать на экране "мышью", при этом Автокад нарисует "резиновую" линию от точки, указанной первым аргументом, до текущего положения курсора. Если первый аргумент опущен, Автокад потребует ввода двух точек. Не запрещается указывать трехмерные точки (см. также GETORIENT)

(getcorner точка "текст запроса-подсказки")

Эта функция, так же как и GETPOINT, возвращает координаты указанной пользователем точки. Отличие от GETPOINT заключается в том, что функция GETCORNER строит "резиновую" рамку при передвижении курсора по экрану, первый угол которой может быть определен первым аргументом функции. В этой функции первый аргумент обязателен. Если вводится трехмерная точка, то координата Z игнорируется (см. функции GETPOINT и GETDIST)

(getdist точка "текст запроса-подсказки")

Какими бы ни были текущие единицы измерения (например, футы), эта функция всегда возвращает действительное число. Если расстояние указывается с клавиатуры, функция тестирует значения системной переменной FLATLAND и флага трехмерных точек функции INITGET. Как видим из таблицы, если системная переменная FLATLAND установлена в 1, а флаг "трехмерные точки" функции INITGET не установлен, то функция GETDIST будет обрабатывать задаваемую трехмерную точку как двумерную, проецируя ее на текущую ПСК:

FLATLAND	INITGET	Возвращается расстояние между трехмерными точками
0	1	Да
0	0	Да
1	1	Да
1	0	Нет

(getenv имяпеременной)

Возвращает строковое значение, присвоенное переменной среды ДОС. Здесь **имяпеременной** - строковая константа, представляющая собой имя переменной среды. Если эта переменная не определена, возвращается nil

(getint "текст запроса-подсказки")

Ввод целого числа. Ввод может быть осуществлен только с клавиатуры (или через планшет)

(getkword "текст запроса-подсказки")

Запрашивает у пользователя ключевое слово, определенное ранее при помощи функции INITGET. Если ответ не совпадает ни с одним определенным ключевым словом, Автокад попросит повторить ввод. Если ответ совпадает с ключевым словом, функция GETKWORD возвращает это ключевое слово как строковую константу. Пустой ввод, если разрешен, возвращает nil

(getorient точка "текст запроса-подсказки")

То же, что и GETANGLE, однако измерение угла осуществляется относительно текущего направления измерения углов, а не относительно нулевого направления оси X (абсолютный угол, в отличие от GETANGLE, которая измеряет относительный угол: угол относительно нулевого направления оси X)

(getpoint точка "текст запроса-подсказки")

Позволяет ввести точку. Если первый аргумент присутствует, Автокад рисует "резиновую" линию от точки, определяемой первым аргументом. Указание трехмерных точек обрабатывается так же, как и в функции GETDIST

(getreal "текст запроса-подсказки")

Позволяет вводить действительное число. Ввод может быть осуществлен только с клавиатуры (или через планшет)

(getstring флагпробела "текст запроса-подсказки")

Запрашивает ввод текстовой константы. Если **флагпробела** указан и не равен нулю, то строковая константа может содержать пробелы и завершением ввода считается нажатие клавиши RETURN. В противном случае строка не может содержать пробелы и клавиша ПРОБЕЛ работает как терминатор ввода

(getvar имяпеременной)

Возвращает значение системной переменной Автокада. **Имяпеременной** должно быть заключено в кавычки (см. также функцию SETVAR)

Как видим, функции семейства GET имеют много общего. Но кроме сходства их объединяет то, что все режимы ввода устанавливаются одной и той же функцией INITGET:

(initget биты строка)

Данная функция устанавливает различные режимы, в которых работают все функции семейства GET, кроме функций GETSTRING и GETVAR. Возвращается всегда nil. **Биты** - необязательное целое число, биты которого интерпретируются следующим образом:

Бит INITGET	*Значение бита*
1	Запрещение пустого ввода
2	Запрещение ввода нуля
4	Запрещение ввода отрицательных чисел
8	Не контролируются лимиты, даже если включен LIMCHECK
16	Возвращаются не двумерные, а трехмерные точки
32	"Резиновая" линия и рамка рисуются пунктиром

Биты задаются целыми значениями, являющимися степенями двойки, как показано в таблице, и общее целое число формируется путем сложения желаемых целых чисел. (Получается логическое И битов.) Следует избегать установки недокументированных битов, поскольку последние версии Автокада могут их использовать. Если установленные для ввода правила не выполнены, то Автокад попросит повторить ввод. Управляющие биты воспринимаются теми функциями, для которых они имеют смысл.

Второй аргумент функции INITGET, **строка**, используется для задания списка ключевых слов при вводе. Использование ключевых слов описано выше.

Отметим, что если вы используете функцию GET внутри команды Автокада, то "резиновая" линия исчезает. Избегайте этого.

8.10. Логические функции Автолиспа

Автолисп предоставляет богатые возможности конструирования логических выражений и выполнения над ними логических операций.

Логический оператор - это функция, сравнивающая между собой два или больше аргументов. Результат сравнения (т.е. некоторое утверждение, касающееся двух аргументов) может быть либо истиной (non-nil), либо ложью (nil). Основные логические функции - это И (AND), ИЛИ (OR) и НЕ (NOT):

(and выражение1 выражение2 ...)

Возвращает результат выполнения логического И над списком выражений. Возвращается nil, если любое (хотя бы одно) из выражений имеет значение nil, в противном случае - T

(not элемент)

Возвращает результат выполнения логического НЕ над своим аргументом

(or выражение1 выражение2 ...)
Возвращает результат выполнения логического ИЛИ над списком выражений. Возвращается nil, если все аргументы имеют значение nil; если при оценке аргументов слева направо встречается выражение не nil, возвращает T

(= атом1 атом2 ...)
Возвращает T, если все атомы равны, в противном случае возвращается nil

(/= атом1 атом2)
Возвращает T, если **атом1** не равен **атому2**. В противном случае возвращается nil. Функция не определена для числа аргументов большего 2

(< атом1 атом2 ...)
Возвращается T в том случае, если каждый последующий атом меньше предыдущего. В противном случае возвращается nil

(<= атом1 атом2 ...)
Возвращается T в том случае, если каждый последующий атом меньше или равен предыдущему. В противном случае возвращается nil

(> атом1 атом2 ...)
Возвращается T в том случае, если каждый последующий атом больше предыдущего. В противном случае возвращается nil

(>= атом1 атом2 ...)
Возвращается T в том случае, если каждый последующий атом больше или равен предыдущему. В противном случае возвращается nil

(eq выражение1 выражение2)
Определяет, идентичны ли **выражение1** и **выражение2**, т.е. являются ли они фактически одним объектом или нет (т.е. порожден ли один из другого при помощи функции SETQ). EQ возвращает T, если оба выражения идентичны, в противном случае - nil. Обычно функция применяется для определения, являются ли два списка фактически одним или нет. Например, пусть

```
(setq f1 '(a b c))
(setq f2 '(a b c))
(setq f3 f2)
```

Тогда **(eq f1 f3)** возвращает nil (**f1** и **f3** - разные списки), а **(eq f3 f2)** возвращает T (**f3** и **f2** - один и тот же список) (См. также функцию EQUAL ниже.)

(equal выражение1 выражение2 допуск)
Определяет, равны ли **выражение1** и **выражение2**, т.е. равны ли их значения. Например, пусть

```
(setq f1 '(a b c))
(setq f2 '(a b c))
(setq f3 f2)
```

Тогда **(equal f1 f3)** возвращает T (значением **f1** и **f3** является одно и то же), а **(equal f3 f2)** возвращает T (**f3** и **f2** - в точности один и тот же список).

Заметим, что, если два списка EQUAL, они могут не быть EQ. Атомы, которые EQUAL, всегда к тому же EQ. Добавим, что два списка, которые EQ, всегда EQUAL.

При сравнении двух действительных чисел (или двух списков действительных чисел, например точек) важно осознавать, что два "идентичных" числа могут незначительно отличаться друг от друга, если они вычислялись различными методами. Однако факультативный численный аргумент **допуск** позволяет задать максимальную точность, с которой могут отличаться друг от друга **выражение1** и **выражение2**, оставаясь при этом EQUAL. Например, пусть

```
(setq a 1.123456)
(setq b 1.123457)
```

Тогда

```
(equal a b)                возвращает nil
(equal a b 0.000001)       возвращает T
```

Кроме логических операций, производимых над выражениями как над целыми объектами, Автолисп включает в себя встроенные функции, производящие логические операции побитово над целыми числами:

(~ целое)

Возвращает логическое НЕТ (дополнение до единицы) своего аргумента. Аргумент должен быть целым числом

(Boole кодфункции целое1 целое2 ...)

Определяет битовую функцию Булевой алгебры, производимой над двумя целыми аргументами. Функции кодируются следующим образом:

Код функции	Булевая функция
1	И
6	Исключающее ИЛИ
7	ИЛИ
8	НЕТ

Существуют и другие булевые функции, соответствующие им коды находятся в диапазоне от 0 до 15.

8.11. Условное ветвление программ

Каждая программа имеет свою логическую структуру. Ветвление - это способ управления ходом выполнения программы. Условные операторы - это средство управления ветвлением программ. Автолисп имеет две функции ветвления, IF и COND. Условное ветвление подразумевает проверку какого-то условия, по результатам которой осуществляется переход. Условия содержатся в условных выражениях, использующих операторы отношения и логические операторы. Условные выражения могут содержать любые выражения Автолиспа.

(cond (тест1 результат1 ...) ...)

Воспринимает в качестве аргументов любое число списков. Она оценивает по очереди первые элементы списков до тех пор, пока не встретится элемент, отличный от nil. Затем вычисляется то выражение, которое следует за **тестом**, и возвращается значение последнего выражения в субсписке. Если в субсписке только одно выражение (например, **результат** отсутствует), то возвращается значение выражения **тест**. COND - основная функция условия в Автолиспе. Как видно из следующего примера, в котором функция COND возвращает 0 или 1 в зависимости от введенной пользователем строковой переменной s, эта функция может использоваться в качестве условного оператора переключения ("case" во многих языках программирования):

```
(cond   ((= s "Y") 1)
        ((= s "y") 1)
        ((= s "Д") 1)
        ((= s "д") 1)
        ((= s "N") 0)
        ((= s "n") 0)
        ((= s "Н") 0)
        ((= s "н") 0)
)
```

(if тест-выражение выражение-тогда выражение-иначе)

Эта функция использует выражение по условию. Если **тест-выражение** не nil, то исполняется **выражение-тогда**, в противном случае - **выражение-иначе**. Последнее выражение не обязательно. Возвращает значение выполненного по условию выражения; если **выражение-иначе** отсутствует и **тест-выражение** - nil, то IF возвращает nil

Иногда требуется по условию выполнить не одно, а несколько выражений, в то время как функция IF не позволяет этого. В этом случае используют функцию PROGN, которая последовательно вычисляет каждое выражение и возвращает значение последнего: **(progn выражение1 выражение2 ...)**. Например:

```
(if (= a b) (progn
                (setq a (+ a 10))
                (setq b (- b 10))
            )
)
```

8.12. Организация циклов

Как и многие языки программирования, Автолисп имеет средства организации повтора выполнения групп операторов. Циклы полезны, например, для:

- уменьшения числа операторов в программе (в случае повтора однотипных действий)
- повтора выполнения процедуры до отмены ее пользователем

- получения решения математической задачи со сходимостью
- пакетной обработки списков данных (например, при работе с DXF-файлом)

Операторы организации циклов Автолиспа не должны быть для вас новыми, если вы уже программировали на каком-то структурном языке, например Паскале. Следует отметить, однако, определенную бедность Автолиспа в этом отношении, обусловленную, вероятно, спецификой Автолиспа.

Простейшим оператором повтора является оператор REPEAT. Эта функция повторяет любое число операторов указанное число раз. Возвращается последнее значение последнего выражения цикла. Синтаксис:

(repeat число выражение1 выражение2 ...)

Функция WHILE похожа на функцию REPEAT, однако число повтора не определено, а выход из цикла осуществляется по условию. В отличие от структуры IF оператор WHILE не включает в себя выполнение какого-либо выражения в том случае, если условие не выполняется. Как COND, REPEAT и PROGN, WHILE позволяет включать в свое тело неограниченное число операторов. В начале выполнения каждого цикла проверяется условие и, если оно выполняется, выполняется тело цикла, после чего опять проверяется условие, и так до тех пор, пока выражение-условие не станет ложным. Будьте внимательными - неаккуратное использование цикла WHILE может привести к зацикливанию программы. Цикл WHILE возвращает значение последнего вычисленного перед выходом из цикла выражения. Если выражение-условие изначально было ложным и вхождения в тело цикла не было, WHILE возвращает nil. Функцию WHILE удобно использовать для проверки правильности ввода, зацикливая ввод до тех пор, пока введенные данные не будут удовлетворять каким-то условиям (прежде чем использовать для этих целей функцию WHILE, убедитесь, что ту же задачу нельзя решить обычными средствами Автолиспа - GETKWORD). Цикл WHILE можно использовать также для организации итераций. Итерация - это процесс повторения группы вычислений до тех пор, пока результат вычисления одного или более выражений не станет удовлетворять какому-то условию. Условное выражение итерационного цикла WHILE обычно содержит некоторую переменную, значение которой изменяется в процессе выполнения тела цикла. Синтаксис WHILE:

(while тест-выражение выражение1 выражение2 ...)

Еще одним типом функции-организатора цикла является функция FOREACH (букв. "для каждого"). Это специальное средство Автолиспа, предназначенное для работы со списками. Синтаксис:

(foreach имя список выражение)

Эта функция, проходя по **списку**, присваивает каждому элементу **имя** и вычисляет каждое **выражение** для каждого элемента списка. Может быть задано не-

ограниченное число выражений. Возвращается результат последнего вычисленного выражения. Например

```
(foreach n '(a b c) (print n))
```

эквивалентно

```
(print a)
(print b)
(print c)
```

с той разницей, что FOREACH возвращает только значение последнего вычисленного выражения.

8.13. Вызов команд Автокада из программы на Автолиспе

Ввод данных в программу на Автолиспе и вспомогательные вычисления над ними призваны служить автоматизации проектирования. Поэтому в Автолиспе не может не быть средства, позволяющего обращаться к командам Автокада из программ пользователя. Таким средством является функция Автолиспа COMMAND:

```
(command аргумент1 аргумент2 ...)
```

Эта функция выполняет команду Автокада из Автолиспа и всегда возвращает nil. **Аргументы** представляют собой команды Автокада и их опции; каждый **аргумент** вычисляется и посылается в Автокад как ответ на соответствующий запрос. Имена команд и опции представляются как строковые константы, двумерные и трехмерные точки - как списки из двух или трех действительных чисел соответственно. Пустая строка функции COMMAND равносильна нажатию пробела на клавиатуре. Вызов COMMAND без аргументов равносилен нажатию CTRL C с клавиатуры и прерывает большинство команд Автокада.

Если переменная Автокада CMDECHO установлена в 0, то при выполнении этой функции на экран не будет выводиться выполнение команды Автокада.

На использование этой функции налагаются некоторые ограничения:

- Функции семейства GET не могут быть вложены в функцию COMMAND. Следует присваивать все необходимые значения переменным заранее
- С помощью функции COMMAND нельзя работать с такими командами Автокада, как ДТЕКСТ, ЭСКИЗ, ЧЕРТИ, ПЕЧАТАЙ и ПАКЕТ, а также с командами, объявленными пользователем (defun C:команда)
- Нельзя использовать восклицательный знак для указания команде Автокада значения переменной

Если в строке аргументов команды, вызываемой функцией COMMAND, встречается ключевое слово Pause, то функция COMMAND приостановит свое действие, чтобы пользователь непосредственно ввел значение (или произвел отслеживание). В этот момент можно выполнить прозрачную команду, после чего выполнение функции COMMAND возобновится. Это позволяет, в частности, в процессе выполнения функции COMMAND использовать команды 'ПОКАЖИ, 'ПАН и др. Пауза будет длиться до тех пор, пока не будет введен допустимый

аргумент и пока не выполнятся все прозрачные команды. При этом если функция COMMAND требует ввода, запрос может быть удовлетворен с помощью меню. Для приостановления действия меню следует указать обратную косую черту.

8.14. Специальные функции

Из Автолиспа можно управлять некоторыми режимами графического редактора не обращаясь к функциям Автокада. К встроенным функциям Автолиспа, управляющим графическим редактором, можно отнести следующие функции:

(graphscr)

Переключает экран из текстового режима в графический в системах с одним экраном (эквивалентна клавише переключения экрана - F1 на IBM PC)

(textscr)

Переключает экран из графического режима в текстовый в системах с одним экраном (эквивалентна клавише переключения экрана - F1 на IBM PC)

(redraw имяпримитива режим)

Действие данной функции зависит от количества аргументов. Если аргументов нет, то она перерисовывает текущий видовой экран, как это делает команда Автокада 'REDRAW ('ОСВЕЖИ): если она вызывается с аргументом **имяпримитива**, то перерисован будет только этот выбранный примитив. Эту функцию часто используют для идентификации примитива на экране после использования функции Автокада GRCLEAR (имена примитивов описаны в разделе "Доступ к примитивам и средствам Автокада" настоящей главы). Полный контроль за перерисовкой примитива обеспечивается заданием параметра **режим**, который может принимать одно из следующих значений:

Режим REDRAW	Действие
1	Перерисовывает примитив на экране
2	Не рисует примитив (стирает)
3	Подсвечивает примитив (если позволяет дисплей)
4	Перестает подсвечивать примитив (если позволяет дисплей)

Если **имяпримитива** - заголовок сложного примитива (полилинии или блока с атрибутами), то в процессе перерисовки будут участвовать как основной примитив, так и все подпримитивы при условии, что аргумент **режим** положителен. Если же аргумент **режим** отрицателен, то в процессе выполнения функции REDRAW будет участвовать только основной примитив. Функция REDRAW всегда возвращает nil

(vports)

Возвращает список дескрипторов действующих в настоящий момент видовых экранов. Каждый дескриптор видового экрана - это список, содержащий но-

мер видового экрана и координаты его нижнего левого и правого верхнего углов. Значения координат находятся в интервале от 0.0 до 1.0, где (0.0 0.0) - координаты нижнего левого угла зоны графического экрана, а (1.0 1.0) - координаты ее верхнего правого угла. Дескриптор текущего видового экрана всегда стоит в списке первым

Таким образом, из программы на Автолиспе можно гибко управлять режимами отображения графического редактора. Но на этом не исчерпывается список ориентированных на систему Автокад функций Автолиспа. В Автолиспе есть функции, позволяющие использовать внутренние возможности графического редактора как системы управления графической базой данных. К таким функциям относятся функции объектной привязки и определения различных геометрических параметров объектов, описываемые ниже.

(osnap точка режим)

Возвращает точку, которая является результатом применения объектной привязки, задаваемой в строке **режим** для указанной **точки**. **Режим** - строковая константа, состоящая из одного или более идентификаторов объектной привязки, как, например, "середина", "центр" и т.д., разделенных запятыми:

```
(setq pt2 (osnap pt1 "сер"))
(setq pt2 (osnap pt1
"сер,кон,цен")
)
```

Если аргумент **точка** - двумерная точка (список из двух действительных чисел), то будет возвращена двумерная точка. Если **точка** - трехмерная точка (список из трех действительных чисел), то будет возвращена трехмерная точка. Если ни одной точки, соответствующей заданному режиму объектной привязки, не найдено, то будет возвращен nil. Отметим, что действие этой функции зависит также от текущего трехмерного вида и от значения системной переменной FLATLAND

(polar точка угол расстояние)

Возвращает точку в ПСК, находящуюся под заданными **углом** и **расстоянием** от заданной **точки**; **угол** измеряется в радианах в направлении против часовой стрелки от оси X. Хотя **точки** могут быть и трехмерными, **угол** всегда определяется в текущей плоскости построений. Если значение системной переменной FLATLAND равно нулю, то возвращаются трехмерные точки, в противном случае - двумерные

(distance точка1 точка2)

Эта функция возвращает расстояние между двумя трехмерными точками. Если значение системной переменной FLATLAND не равно нулю, то функция DISTANCE предполагает двумерные точки (игнорирует координату Z переданной точки) и возвращает расстояние между точками - проекциями указанных трехмерных точек на текущую плоскость построений

(inters точка1 точка2 точка3 точка4 неопред)

Возвращает точку пересечения двух отрезков (**точка1 точка2**) и (**точка3 точка4**). Все точки выражены в координатах текущей ПСК. Если значение системной переменной FLATLAND равно нулю, то точки считаются трехмерными и контролируется пересечение в трехмерном пространстве. В противном случае отрезки проецируются на текущую плоскость построений и пересечение контролируется на плоскости. Если факультативный аргумент **неопред** присутствует и является nil, то контролируется пересечение не отрезков, а определяемых ими прямых и INTERS будет возвращать точку пересечения даже в том случае, если она не принадлежит ни одному из отрезков. Если же факультативный аргумент **неопред** отсутствует или не является nil, то точка пересечения должна принадлежать обоим отрезкам (отрезки должны пересекаться), иначе будет возвращен nil

(trans точка из в вектор)

Эта функция преобразует координаты точки (или величину перемещения) из одной системы координат в другую. Аргумент **точка** - список из трех действительных чисел, который можно интерпретировать либо как трехмерную точку, либо как трехмерное перемещение (**вектор**). **из** - код системы координат, в которой находится указанная точка, а **в** - код системы координат, в которой происходит преобразование координат точки. Если имеется факультативный аргумент **вектор** и его значение не равно нулю, то аргумент **точка** будет трактоваться как трехмерное перемещение. (Подробнее см. в книге "Автолисп. Версия 10. Руководство по программированию", изданной на русском языке.)

8.15. Работа с файлами: ввод/вывод

Работая с Автолиспом, у вас рано или поздно возникнет желание записать ту или иную информацию в файл или принять ее оттуда. Средства работы с файлами в Автолиспе не слишком развиты, однако для большинства повседневных нужд этого вполне достаточно. Рассмотрим функции работы с файлами в той последовательности, в которой вы, скорее всего, будете ими пользоваться.

(open имяфайла режим)

Эта функция открывает файл. Возвращается дескриптор файла

Функции ввода/вывода работают не с именами файлов, а с т.н. дескрипторами файлов, определяющими не только собственно файл, но и режим доступа и различные другие системные параметры. "Открыть файл" - значит подготовить дескриптор файла к использованию его функциями ввода/вывода Автолиспа. Поэтому возвращаемое функцией OPEN значение дескриптора файла должно присваиваться некоторой символьной переменной, например:

```
(setq a (open "file.ext" "r"))
```

Здесь переменная **a** - дескриптор файла file.ext, открытого для чтения. Флаг чтения или записи - это, как и имя файла, строковая константа, состоящая из

одной буквы, которая должна быть набрана на нижнем регистре. Допустимые значения флага чтения/записи приводятся ниже:

Режим OPEN	*Описание*
r	Открыть файл для чтения. Если файл с указанным именем не существует, возвращается nil
w	Открыть файл для записи. Если файл с указанным именем не существует, создается и открывается новый файл. Если такой файл уже существует, то хранящиеся в нем данные будут утеряны
a	Открыть файл для добавления данных. Если файл с указанным именем не существует, создается и открывается новый файл. Если файл уже существует, то новые данные будут записаны после старых. Добавляя данные в файлы, созданные другими программами, следует осуществлять проверку на присутствие в файле символа ^Z. При обнаружении этого символа (десятичный код ASCII 26) ДОС считает, что файл окончился, даже если после маркера еще есть данные

Имя файла может включать в себя путь (имена надкаталогов). В системах MS-DOS/PC-DOS допускается также использовать букву устройства ввода/вывода; и вы можете пользоваться обратной косой чертой вместо прямой косой черты. (Напомним: чтобы ввести в строку одну обратную косую черту, следует напечатать две обратных косых черты: "\\")

(close дескрипторфайла)

Эта функция завершает все процессы, инициированные функцией OPEN. После закрытия файла использование **дескрипторфайла** невозможно, хотя сам дескриптор файла не изменяется. Использование функции CLOSE обязательно - все открытые файлы должны быть закрыты, иначе данные могут быть потеряны

(findfile имяфайла)

Эта функция отыскивает файл **имяфайла** (т.е. возвращает полное имя файла по указанному основному имени файла) в каталогах, хранящих файлы Автокада. Сначала просматривается текущий каталог, затем каталог, в котором хранится текущий рисунок, и наконец, каталог, имя которого записано в переменной среды ACAD. Такое полное имя можно передавать функции OPEN

(read строка)

Эта функция возвращает первый список или атом из данной строки, причем строка не должна содержать пробелов

(read-char дескрипторфайла)

Считывает единичный символ из буфера клавиатуры или из открытого файла, заданного аргументом **дескрипторфайла** Возвращается целое число - код

ASCII считанного символа. Если **дескрипторфайла** не задан и буфер клавиатуры пуст, функция ожидает ввода символа (пользователь должен ввести что-либо с клавиатуры, заканчивающееся RETURN). Например, если в ответ на запрос функции READ-CHAR пользователь введет с клавиатуры "ABC" и завершит ввод нажатием клавиши RETURN, то функция READ-CHAR возвратит код 65 (код ASCII латинской буквы "A"). При следующих трех обращениях к READ-CHAR она возвратит соответственно 66, 67 и 10 (код перевода строки). Если последует пятый вызов функции READ-CHAR, она снова будет ожидать ввода. Для унификации работы программ на Автолиспе в различных операционных системах функция READ-CHAR возвращает при чтении конца строки один символ с кодом 10

(read-line дескрипторфайла)

Данная функция считывает строку символов с клавиатуры или из открытого файла, заданного аргументом **дескрипторфайла**. Возвращается считываемая строка. Если достигнут конец файла, возвращается nil

(write-char число дескрипторфайла)

Эта функция записывает один символ на экран или в открытый файл, заданный аргументом **дескрипторфайла**. Здесь **число** - код ASCII символа, и является возвращаемым функцией значением. Функция WRITE-CHAR не может записать в файл символ NUL (ASCII - код 0)

(write-line строка дескрипторфайла)

Эта функция записывает строковую константу **строка** на экран или в открытый файл, заданный аргументом **дескрипторфайла**. Она возвращает строку, взятую в кавычки и опускает кавычки при записи в файл

8.16. Работа со списками

Как уже говорилось в п. 8.4.2, список - это группа элементов любого допустимого в Автолиспе типа. Список Автолиспа может содержать любое количество действительных и целых чисел, строк, переменных или других списков. Все, что находится между открывающей и закрывающей круглыми скобками, - список. Списки удобно использовать для организации и обработки больших массивов связанных данных. Автолисп имеет развитые средства работы со списками. Они вкратце описаны ниже:

(list выражение1 выражение2)

Эта функция просто составляет список из своих аргументов. Например, выражение **(list 5.5 8.0)** возвращает список, содержащий два действительных числа (точку): **(5.5 8.0)**; этот способ часто используется в Автолиспе для создания новых точек из известных координат. Подчеркнем, что при составлении списка функция LIST оценивает выражения-элементы

(quote выражение) или 'выражение

Функция QUOTE (буквально - "цитировать") подавляет оценку своего

выражения. Это бывает нужно при формировании списка. Например, выражение **(list 'a 'b 'c)** формирует список **(a b c)**

Поскольку список представляет собой группу элементов, может возникнуть необходимость извлечения одного элемента из списка. Существуют две простейшие функции извлечения элементов списка:

(car список)
Возвращает первый элемент **списка**. Если **список** пуст, возвращается nil

(cdr список)
Возвращает все элементы **списка**, кроме первого. Если **список** пуст, возвращается nil. Если элементом **списка** является точечная пара, функция возвращает последний элемент, не заключая его в **список**

Из этих простейших функций, как из кирпичиков, составлены сцепления функций извлечения элементов списка вплоть до 4 уровней вложенности:

(caar x)	эквивалентно	**(car (car x))**
(cdar x)	эквивалентно	**(cdr (car x))**
(cadar x)	эквивалентно	**(car (cdr (car x)))**
(cadr x)	эквивалентно	**(car (cdr x)**
(cddr x)	эквивалентно	**(cdr (cdr x))**
(caddr x)	эквивалентно	**(car (cdr (cdr x)))**

В Автолиспе CADR часто используется для "извлечения" координаты Y двумерной или трехмерной точки, а CADDR - для извлечения координаты Z. Для этой же цели можно воспользоваться функцией NTH:

(nth номер список) Извлекает элемент **списка** с нужным **номером**

Однако будьте осторожны при работе с этой функцией: первый элемент списка имеет номер 0, а не 1 (функция NTH считает не 1, 2, 3, ..., а 0, 1, 2, 3, ...). Так считает машина, однако вам-то от этого не легче - учтите, что в счете элементов списков у функций Автолиспа нет единообразия, поэтому нужно быть внимательным. Кроме того, к средствам поддержки списков относятся функции:

(last список)
Возвращает последний элемент **списка**, причем **список** не должен быть равен nil. НЕ РЕКОМЕНДУЕТСЯ использовать эту функцию для извлечения Y- или Z-координаты точки, поскольку это требует дополнительных усилий по поддержке правильной размерности точек

(reverse список)
Возвращает **список** с элементами, переставленными в обратном порядке

(length список)
Возвращает целое число, равное числу элементов в **списке**

(append выражение1 выражение2 ...)

Берет любое число выражений (списков) и сливает их в один список

(cons выражение список)

Эта функция, наряду с функцией LIST, используется для создания списков. Она добавляет в начало **списка** новый элемент, которым может быть также и **выражение**

Кроме этих простейших функций существуют функции ASSOC и SUBST, позволяющие работать со структурированными списками, подобно структурам в других языках программирования. Работа со структурированными списками будет рассматриваться ниже, при разборе работы с графической базой данных Автокада (пункт 8.22).

8.17. Создание функции в Автолиспе

Вы можете определять в Автолиспе свои собственные функции. Мы уже создавали свою собственную функцию C:HOLE, используя при этом функцию DEFUN (DEfine FUNction - определить функцию). Функция DEFUN определяет функцию посредством создания структурированного списка операторов программы. Определенная вами функция создает свою собственную замкнутую область локальных переменных. При вызове функции в эту замкнутую область передаются данные, выполняются операторы вашей программы, после чего осуществляется передача данных обратно в среду Автолисп-Автокад.

Вспомним программу HOLE (ОТВЕРСТИЕ), в которой мы уже использовали функцию DEFUN:

```
Command: (defun C:HOLE () (setq rad (/ (getdist "\nDiameter: ") 2))
1>(while T (command "Donut" pause rad)))
```

```
Команда: (defun C:ОТВЕРСТИЕ () (setq rad (/ (getdist "\nДиаметр: ") 2))
1>(while T (command "Кольцо" pause rad)))
```

Как видим, формат функции DEFUN следующий:

```
(defun имя (аргументы / локальные параметры)
   тело функции
)
```

Действительно, функция HOLE выглядит как:

```
(defun C:ОТВЕРСТИЕ ()
   (setq rad (/ (getdist "\nДиаметр: ") 2))
   (while T (command "Круг" pause rad))
)
```

Правила именования функций те же, что и правила именования переменных. Помните, что если вы переопределили встроенную функцию Автолиспа, то она останется для вас недоступной до тех пор, пока вы не начнете новый рисунок.

Функция HOLE не имеет локальных параметров или аргументов. Их мы рассмотрим позже. Операторы вашей программы, такие, как (setq rad ...) или (while T ...), составляют ядро вашей функции. Порядок оценки их вычислителем Автолиспа мы уже обсуждали. Последнее возвращаемое значение передается в среду Автокад-Автолисп.

При запуске графического редактора Автокад загружает файл acad.lsp (если он существует). В этот файл можно внести определения наиболее часто используемых функций, и они будут загружаться автоматически при вызове графического редактора. При этом на экране появится сообщение:

```
Loading acad.lsp...loaded.
```

```
Загружаю acad.lsp...загружен.
```

При определении функции при помощи DEFUN можно указать, что она должна выполняться автоматически сразу же после загрузки. Если вы поместите определенную таким образом функцию в файл acad.lsp, то она будет автоматически загружаться и выполняться сразу же после вызова Автокада.

Определенные вами функции могут быть использованы так же, как и встроенные функции Автокада. Это позволяет сделать программу модульной, многократно использующей какие-то подпрограммы. Гибкость вызовов функций Автолиспа хорошо иллюстрируется в предлагаемом в разделе 8.19 примере.

8.18. Установка текстового редактора

Когда вы создавали функцию C:HOLE, то не сохраняли ее. Между тем программу Автолиспа можно записывать в файл, что позволяет не набирать ее каждый раз заново, а загружать из файла (и конечно, редактировать). Файлы программ на Автолиспе имеют расширение .lsp. Программа может содержать любое количество определений функций и других выражений.

В отличие от файлов, содержащих меню Автокада, файлы программ на Автолиспе являются обычными текстовыми файлами, и вам не надо беспокоиться о разбиении файла на страницы или считать пробелы. Пустые строки не рассматриваются интерпретатором Автолиспа как нажатия клавиши ENTER, а просто игнорируются, как и лишние пробелы. Вы набираете программу-функцию в любом текстовом редакторе, не вставляющем в файл своих служебных символов (это может быть, например, строковый редактор EDLIN, входящий в комплект ДОС, экранные редакторы Norton Editor, MultiEditor или даже Лексикон). Не рекомендуется для создания программы на Автолиспе использовать текстовые процессоры, такие, как Microsoft Word или CharWriter, поскольку они используют свой формат файлов, вставляя в них свою служебную информацию, которая не может быть обработана компилятором Автолиспа (конечно, они позволяют выводить набранный документ в т.н. режиме ASCII без форматирования, но их назначение не набор программ, а создание документов).

В стандартной поставке Автокада в подменю UTILITY/External Commands имеется пункт EDIT:, выбор которого приводит к автоматическому вызову строкового редактора EDLIN. При выборе EDIT на экране высвечивается подсказка:

```
File to edit:<Введите имя файла>
```

```
Редактировать файл:<Введите имя файла>
```

После того как вы введете имя файла, Автокад вызывает текстовый редактор. Если вы не настраивали ваш Автокад, то этот редактор - строковый редактор EDLIN. Если вы сталкивались с ним раньше, то знаете, что это не самый удобный редактор. Мы советуем вам подключить к Автокаду редактор Norton Editor. Его преимущество в том, что он невелик, т.е. требует мало памяти, и в то же время это достаточно мощный для нас экранный редактор.

Все внешние команды Автокада определены в файле acad.pgp:

```
CATALOG,DIR /W,30000,*Files: ,0
DEL,DEL,30000,File to delete: ,0
DIR,DIR,30000,File specification: ,0
EDIT,EDLIN,42000,File to edit: ,0
SH,,30000,*DOS Command: ,0
SHELL,,127000,*DOS Command: ,0
TYPE,TYPE,30000,File to list: ,0
```

```
КАТАЛОГ,DIR /W,27000,*Показать файлы: ,0
DEL,DEL,27000,Удалить файл: ,0
DIR,DIR,27000,Показать файлы: ,0
РЕДАКТ,edlin,30000 Редактировать файл: ,0
ДОС1,,27000,*Команда ДОС: ,0
ДОС,,127000,*Команда ДОС: ,0
TYPE,TYPE,27000,Файл для просмотра: ,0
```

Как видим, каждая строка в этом файле соответствует одной команде подменю External Commands (Внешние команды). Это подменю называется так именно потому, что его команды берутся из файла acad.pgp, а не записаны в самом Автокаде. Это позволяет, в частности, легко модифицировать команды этого подменю. Рассмотрим одну строку (команду) этого файла, EDIT.

Первым элементом строки, выделенным запятой, является команда, высвечиваемая в Автокаде в подменю External commands. Вторым элементом является команда ДОС, которая будет выполнена при выборе из меню команды. Третьим элементом командной строки является объем памяти, который освобождает Автокад перед вызовом редактора. Четвертым элементом строки, если вы уже обратили внимание, является подсказка, которая высвечивается после выбора команды и требует ввести имя файла, который вы хотите редактировать. И наконец, пятым элементом строки является целое число, определяющее, какой экран (графический или текстовый) будет вызван при возврате в Автокад (0 - тексто-

вый экран, 4 - графический). Для того чтобы вместо редактора EDLIN вызывался редактор Norton Editor, нужно заменить строку EDIT на следующую:

```
EDIT, ne, 60000, File to edit: , 0
```

```
РЕДАКТ, ne, 60000, Редактировать файл: , 0
```

Теперь гораздо удобнее редактировать файл с программой на Автолиспе.

8.19. Написание программы на Автолиспе

Используя Автолисп, попробуем ввести в Автокад новую команду. В процессе написания программы мы разберем, как работает Автолисп и дадим вам возможность понять, как при помощи Автолиспа модифицировать среду Автокад. Программа, которую мы будем составлять, относится к области проектирования печатных плат, однако основные понятия и приемы, которыми мы будем пользоваться, носят универсальный характер и не зависят от области применения.

Наша цель заключается в создании новой команды Автокада, рисующей ряд отверстий с металлизированной окантовкой, причем одна из площадок должна иметь хвостик для дополнительной связи на плате. Эта новая команда при вызове должна запрашивать координату первого отверстия, расстояние между двумя рядами ножек микросхемы и число ножек в одном ряду на микросхеме. Будем решать эту задачу, как и большинство подобных задач, переходя от частного к общему, или методом "снизу-вверх". Вы уже знаете, что в Автолиспе углы измеряются в радианах и могут принимать значения от 0 до 2*PI радиан. Поскольку более удобно измерять углы в градусах, введем новую функцию, преобразующую градусы в радианы. Итак, войдите в редактор, указав ему имя myprog.lsp, и наберите текст следующей программы:

```
;    Преобразование углов из градусов в радианы
     (defun dtr (a)
          (* pi (/ a 180.0))
     )
     (prompt "\nПрограмма загружена - синтаксических ошибок нет!")
;    Конец программы
```

Автолисп считает текст, вводимый после точки с запятой, комментарием, т.е. игнорирует его. Расположение скобок не играет никакой роли при интерпретации программы вычислителем Автолиспа, однако программа получается более читабельной, если закрывающие скобки располагаются либо на той же строке, либо в том же столбце, что и открывающие.

Сохраните файл с программой на диске и загрузите его в Автокад:

```
Команда: (load "myprog")
Автолисп возвращает  pi
```

Здесь load - функция Автолиспа, выполняемая с командной строки Автокада. Эта функция вызывает программу из файла в среду Автокад-Автолисп. Указывая ей имя файла, не надо писать расширение - она берет файлы с расширением .lsp по умолчанию. (Не путайте функцию LOAD Автолиспа с командой LOAD Автокада, загружающей файлы форм, - это предупреждение относится особенно к тем, кто работает на английской версии Автокада.) При загрузке функции Автолисп читает из файла определение функции, осуществляет синтаксическую проверку и, если ошибок (синтаксических!) нет, сохраняет функцию в области памяти, специально отведенной для пользовательских программ, после чего отображает на экране имя загруженной функции. Все, что не является определением функции (в нашем случае это подсказка "prompt"), выполняется Автолиспом немедленно. Определенная пользователем функция должна быть вызвана явно.

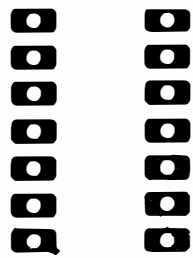
Теперь можно проверить работу этой функции, вводя разные значения аргументов:

```
Команда: (dtr 180)
Автолисп возвращает: 3.141593
```

Аргументы функции - это имена переменных, которые используются для передачи данных в локальную среду функции. Аргументы функции называются также формальными параметрами (в отличие от фактических параметров, т.е. тех параметров, которые указываются при вызове функции). В примере a - формальный параметр, число 180 - фактический. Список фактических параметров должен соответствовать списку формальных параметров, т.е. они должны иметь одинаковое число элементов, без пропусков и перестановок.

Теперь мы можем полностью оформить нашу команду-функцию отрисовки площадок с отверстиями под микросхему:

```
(defun element ()
  (defun dtr (a) (* pi (/ a 180.0)))
  (setq sp (getpoint "\nНачальная точка: "))
  (setq tspac (getdist "\nРасстояние между рядами: "))
  (setq nn (getint "\nЧисло элементов в ряду: "))
  (setq sp1 sp)
  (command "КРУГ" sp 0.6)
  (setq l1 (entlast))
  (setq p (polar sp (dtr 90) 0.75))
  (setq p1 (polar p 0 1.5))
  (setq p3 (polar p1 (dtr -90) 1.5))
  (command "ПЛИНИЯ" p "Ш" 0 0 (setq p (polar p 0 1.5))
           (setq p (polar p (dtr -90) 1.5))
           (setq p (polar p (dtr 180) 3))
           (setq p (polar p (dtr 90) 1.5))
           "Замкни" )
  (setq l5 (entlast))
  (command "ФАСКА" "Длина" 0.2 0.2
           "ФАСКА" "Полилиния" l5)
  (setq l5 (entlast))
```



```
  (setq l2 (entlast))
  (command "ШТРИХ" "ansi31" 1 0  l1 l5 ""
           "МАССИВ" l1 l2 l5 "" "П" nn 2 2.5 tspac
           "ВСТАВЬ" "border" (polar p3 (dtr 180) 0.2) 0.3 0.2828427 -45
           "ШТРИХ" "ansi31" 0.5 0 (entlast) "")
)
(defun c:socket ()
  (setq sblip (getvar "blipmode"))
  (setq scmde (getvar "cmdecho"))
  (setvar "blipmode" 0)
  (setvar "cmdecho" 0)
  (element)
  (setvar "blipmode" sblip)
  (setvar "cmdecho" scmde)
)
```

Основная функция, оформляемая как команда Автокада, определяется последней. Вообще, если одна функция вызывает другую, то вызываемая функция должна быть определена раньше, чем вызывающая.

8.20. Использование Автолиспа в макроопределениях

Использование Автолиспа при создании макроопределений в меню позволяет:

- сохранять данные в переменных, обрабатывать данные и передавать их Автокаду
- управлять отрисовкой примитивов посредством установки системных переменных Автокада, сохранения системных переменных, запрашиванием настроек и изменением их в соответствии с ответами пользователя
- устанавливать переменные в процессе выполнения команд (в прозрачном режиме) для более полного удовлетворения запросов пользователя
- устанавливать ответы по умолчанию на запросы Автокада и использовать их в командах и макроопределениях
- считывать и изменять геометрические данные объектов, используя данные в своих программах
- получить доступ к справочной информации, содержащейся в таблицах Автокада, например об установке слоев, блоков, стилях текста и типах линий

В этом разделе будет показано на примере, как можно воспользоваться средствами Автолиспа для модификации меню Автокада. Вообще-то использование Автолиспа при изменении меню не обязательно - меню может и не обращаться к Автолиспу, используя только стандартные команды Автокада. Однако практика показывает, что на каком-то этапе развития вашего меню пользоваться средствами Автолиспа вам все-таки придется. Кроме того, даже если ваш пункт меню совсем прост и использует только команды Автокада, все равно желательно при помощи Автолиспа сохранять установленные по умолчанию значения используемых системных переменных Автокада и восстанавливать их по окончании работы

вашего подменю. Это позволит вам не беспокоиться об изменении установок Автокада или восстанавливать их вручную.

Для примера решим следующую задачу. Если вы устанавливали в своей системе редактор Norton Editor так, как было показано нами ранее, то, возможно, нашли утомительным каждый раз при вызове редактора задавать ему имя файла, которое надо указывать полностью - с расширением lsp. Кроме того, при ошибке в имени файла этот редактор не выдает никакого сообщения, а просто открывает новый файл с тем именем, которое было ему указано. Поставим себе цель организовать подменю Автокада, названное, скажем, LISP, в котором можно было бы, один раз задав имя файла-программы на Автолиспе, многократно вызывать его на редактирование. Желательно, чтобы имя файла можно было задавать без расширения, т.к. оно в нашем случае меняться не будет.

Решение лежит на поверхности - нужно написать на Автолиспе программу (а если быть более точным, то это макроопределение, поскольку оно будет вставлено в меню), которая запоминала бы один раз указанное имя файла в некоторой текстовой переменной. Поскольку внешним командам, как и любым другим командам Автокада, можно из Автолиспа передавать параметры, нет ничего проще, чем автоматизировать этот процесс.

Ниже показано, как можно это сделать. Придется внести изменения в меню Автокада (файл acad.mnu). Поместим подменю LISP в экранное (SCREEN) меню Автокада. Это делается добавлением в экранное меню всего одной строки:

```
***SCREEN
**S
[AutoCAD]^C^C$S=X $S=S $P1=POP1 $P3=POP3
[* * * *]$S=OSNAPB
[Setup]^C^C^P(progn(prompt "Loading setup...  ")(load "setup")) ^P$S=X+
$S=UNITS

[BLOCKS]$S=X $S=BL
[DIM:]$S=X $S=DIM ^C^CDIM
[DISPLAY]$S=X $S=DS
[DRAW]$S=X $S=DR
[EDIT]$S=X $S=ED
[INQUIRY]$S=X $S=INQ
[LAYER:]$S=X $S=LAYER ^C^CLAYER
[SETTINGS]$S=X $S=SET
[PLOT]$S=X $S=PLOT
[UCS:]$S=X $S=UCS1 ^C^CUCS
[UTILITY]$S=X $S=UT
[LISP]$S=X $S=LP

[3D]$S=X $S=3D
[ASHADE]^C^C^P(progn(setq m:err *error*)(prin1))(defun *error* (msg)+
(princ msg)(setq *error* m:err m:err nil)(princ))(cond ((null C:SCENE)(vmon)+
(if (/= nil (findfile "ashade.lsp"))(progn (terpri);+
(prompt "Please wait...  Loading ashade.  ")(load "ashade")+
(menucmd "S=X")(menucmd "S=ASHADE")(setq *error* m:err m:err nil))+
```

```
(progn (terpri);(prompt "The file 'Ashade.lsp' was not found in your current+
search+ directories.")(terpri)(prompt "Check your AutoShade Manual+
for installation instructions.");(setq *error* m:err m:err nil)(princ))))+
(T (setq *error* m:err m:err nil)(menucmd "S=X")+
(menucmd "S=ASHADE")(princ))) ^P
[SAVE:]^C^CSAVE
```

Как видим, новое подменю отличается от стандартного только одной строчкой:

```
[LISP]$S=X $S=LP
```

С ее помощью определяется подменю LP, в котором будут находиться добавляемые нами средства. Переход в это подменю осуществляется при выборе пункта LISP стандартного экранного меню Автокада. Теперь самое главное - написание собственно макроопределений меню:

```
**LP 3


[Lisp]^C^C^P (progn (terpri) (setq lispfile (strcat (setq lispfile (getstring +
"Input Lisp file name (without extension): ")) ".lsp")))
[File:]^C^C^P (progn (terpri) (setq lispfile (strcat (setq lispfile+
(getstring "Input Lisp file name (without extension): ")) ".lsp")))
[EDIT:]^C^C^P(progn (if (/= lispfile nil) (progn (terpri) (setq lispfile (strcat +
(setq lispfile (getstring "Input Lisp file name (without extension): "))+
".lsp"))) (command "LEDIT"))
[LOAD:]^C^C^P (if (= lispfile nil) (progn (terpri) (setq message +
"Nothing to load: Enter file name first!") (terpri) (setq a nil)) (progn (terpri)+
(load lispfile))^P

[LIST:]^C^C^P(command "DIR" "*.lsp") ^P
```

Мы не будем здесь подробно останавливаться на организации этого подменю, поскольку это не входит в задачу настоящего раздела. Попробуйте разобрать работу этих макроопределений самостоятельно. Достаточно описать алгоритм макроопределений, т.е. какие действия должны осуществляться при выборе пунктов нашего подменю. Всего пунктов 4: Lisp File, EDIT и LOAD. Самый простой из них - пункт LOAD, осуществляющий загрузку в Автокад программы на Автолиспе. Загружается та программа, с которой идет работа в настоящий момент. Пункт Lisp File (фактически это один и тот же пункт, который активизируется выбором как Lisp, так и File) запоминает в строковой переменной lispfile вводимое пользователем имя (без расширения) текущего файла Автолиспа. После ввода имени файла к нему подсоединяется расширение .lsp. При выборе пункта меню EDIT автоматически выполняется внешняя команда EDIT, которой передается в качестве параметра имя файла на Автолиспе.

В пунктах EDIT и LOAD нужно предусмотреть проверку имени файла на тот случай, если пользователь выберет этот пункт до того, как будет задано имя

файла. При этом имя файла запрашивается и в пункте EDIT, а в пункте LOAD выдается сообщение об ошибке.

Описанная выше процедура кажется сложной, однако, один раз ее проделав, вы сэкономите себе много времени при работе с Автолиспом. В этом и заключается смысл автоматизации пользовательских приложений.

8.21. Доступ к примитивам и средствам Автокада

Чуть ли не самой главной особенностью Автолиспа является то, что он позволяет осуществлять доступ к графической базе данных (ГБД) Автокада, многократно умножая возможности адаптации последнего к какому-либо типу задач. Попытаемся вкратце показать, как можно использовать возможности Автолиспа для работы непосредственно с объектами чертежа.

Как вы уже знаете, любой создаваемый в Автокаде чертеж состоит из отдельных примитивов, геометрическое описание которых хранится в специальном формате (формате Автокада) в файле чертежа (расширение .dwg). При загрузке чертежа Автокад заполняет графическую базу данных - заносит в нее системные настройки, создает список объектов и вносит в ГБД геометрическое описание этих объектов, присваивая при этом каждому примитиву уникальное имя. Итак, в сеансе редактирования каждый примитив Автокада (отрезок, дуга, окружность и т.п.) имеет свое имя, по которому его распознает сам Автокад. Поскольку эти имена меняются от одного сеанса редактирования к другому, не имеет смысла их хранить - даже не пытайтесь запомнить их. Вместо этого следует в программе на Автолиспе сначала найти имя примитива в базе данных Автокада с тем, чтобы впоследствии непосредственно манипулировать геометрическими характеристиками примитива или использовать их в макроопределениях меню. Попробуем извлечь это имя из ГБД при помощи Автолиспа. Нарисуйте отрезок и введите с командной строки Автокада строку:

```
Command: (setq ename (entlast))
Автокад возвращает: <Entity name: 60000018>
```

```
Команда: (setq ename (entlast))
Автокад возвращает: <Имя примитива: 60000018>
```

Тем самым мы присвоили переменной ENAME имя последнего примитива (в данном случае отрезка). Напомним еще раз, что имена примитивов меняются от сеанса к сеансу, и вы наверняка увидите на экране другое имя. Имена примитивов в Автокаде - шестнадцатеричные величины; имя примитива может быть, например таким: 60000A14. Используя это имя, вы можете при помощи функции ENTGET получить доступ к данным, связанным с примитивом:

```
(setq edata (entget ename))
```

Имя примитива - это новый для нас тип данных Автолиспа, и если функции Автолиспа ENTGET требуется имя примитива, то бесполезно указывать число 60000018 - надо передать переменную ENAME, в которой это имя хранится.

В результате выполнения команды вы получите малопонятное сообщение:

```
((-1 . <Имя примитива: 60000020>) (0 . "LINE") (8 . "0")
(10 1.0 2.0 0.0) (11 6.0 6.0 0.0))
```

Дело в том, что с точки зрения Автолиспа все данные, описывающие примитив, представляют собой список, состоящий, в свою очередь, из подсписков, в которых сгруппированы по функциональному назначению все данные о примитиве, как геометрические, так и общие - слой, цвет и т.п. Подсписки отличаются друг от друга по специальным кодам формата DXF (Drawing eXchange Format - формат обмена рисунками), которые позволяют определить, какой тип данных хранится в подсписке. Каждый подсписок имеет две части. Первая - код DXF, вторая - данные. Целое число 0, например, представляет собой код типа примитива. Код 8 говорит о том, что следующее за ним число - номер слоя. Код 10 - начальная точка примитива, код 11 - конечная и т.п. Отметим, что набор кодов DXF различен для примитивов разных типов. Однако сами коды относятся ко всем примитивам - имя примитива, например, всегда хранится в подсписке с кодом DXF -1.

Представим полученный список EDATA в более понятном виде:

```
(    (-1 . <Имя примитива: 60000020>)
     (0 . "LINE")              ; Тип примитива
     (8 . "0")                 ; Слой
     (10 1.0 2.0 0.0)          ; Начальная точка
     (11 6.0 6.0 0.0)          ; Конечная точка
)
```

Как нетрудно догадаться, пользуясь кодами DXF, можно извлечь из списка EDATA любую информацию о примитиве. Такой доступ к рисунку более сложен, но предоставляет невиданные ранее возможности. Теперь вы сможете изменять практически все свойства примитивов.

В Автокаде имеется стандартное средство работы не с одним, а с несколькими примитивами. Это средство называется набором. Практически все команды редактирования работают не с отдельными примитивами, а с группой примитивов. Из Автолиспа также можно работать с наборами примитивов - предоставлять пользователю возможность заносить примитивы в набор и затем модифицировать занесенные туда примитивы. Набор формируется функцией SSGET:

(ssget режим точка1 точка2)

Необязательный аргумент **режим** - это строка, которая указывает способ выбора примитива. Им может быть "P" "C" "П" и "T" - соответствующие Рамке, Секущей рамке, Последнему и Текущему наборам примитивов Автокада. Аргументы **точка1** и **точка2** - списки точек, указывающие точки, относящиеся к выбору

Примеры функции SSGET:

(ssget)	**Выбирает по одному объекты чертежа**
(ssget "Т")	**Выбирает текущий набор**
(ssget "П")	**Выбирает последний примитив**
(ssget '(2 2))	**Выбирает примитив, проходящий через точку (2,2)**
(ssget "Р" '(0 0) '(5 5))	**Выбирает примитивы в рамке от (0,0) до (5,5)**
(ssget "С" '(0 0) '(4 5))	**Выбирает примитивы, пересекаемые рамкой от (0,0) до (4,5)**
(ssget "Х" список-фильтр)	**Выбирает примитивы в соответствии со списком-фильтром**

Выбранные объекты подсвечиваются только тогда, когда SSGET используется без аргументов.

Особый режим - режим SSGET "Х" - фильтры выбора. В этом режиме функция SSGET просматривает весь рисунок и создает набор, состоящий из имен всех основных примитивов, удовлетворяющих заданным критериям. Используя этот режим, можно создать набор, состоящий из всех примитивов заданного типа или находящихся на заданном слое заданного цвета.

В Автолиспе имеются хорошие средства работы с наборами примитивов. Рассмотрим их вкратце:

(sslength набор)

Возвращает число примитивов в **наборе**. Если это число больше 32 767, то оно возвращается как действительное

(ssname набор номер)

Возвращает имя примитива под **номером** из **набора**. Запомните, что первый примитив в наборе имеет номер 0! Если примитивов слишком много, используйте действительные числа (округляются до целых)

(ssadd имяпримитива набор)

Если вызвать функцию без аргументов, SSADD создаст новый набор без единого примитива. Если вызвать только с аргументом **имяпримитива**, будет создан набор, содержащий один примитив. Если вызвать с обоими указанными аргументами, примитив **имяпримитива** будет добавлен в **набор**

(ssdel имяпримитива набор)

Удаляет **имяпримитива** из **набора** и возвращает новый набор

(ssmemb имяпримитива набор)

Если примитив **имяпримитива** находится в **наборе**, то возвращается имя примитива, если нет - nil

Рассмотрим далее две функции, позволяющие извлекать имена примитивов из ГБД по их порядку:

(entnext имяпримитива)

Будучи вызвана без параметров, возвращает имя первого неудаленного примитива в ГБД. Если задан аргумент **имяпримитива**, будет возвращено имя

первого неудаленного примитива, следующего в ГБД за примитивом **имяпримитива**. Если примитива нет, будет возвращен nil. Возвращаются как основные примитивы, так и субпримитивы (например, вершины полилинии, полученные в результате аппроксимации последней сплайном)

(entlast)

Возвращается имя последнего неудаленного примитива в ГБД

Из Автолиспа можно непосредственно модифицировать данные о существующих в ГБД примитивах. Однако если мы хотим добавить новый примитив, то должны использовать команды отрисовки или редактирования Автокада. Это ограничение связано с желанием защитить ГБД от неграмотного программиста: неправильно пользуясь командами Автокада, вы не сможете испортить саму ГБД - самое большее, что вы испортите, это свой чертеж.

Примитивы можно удалять из чертежа:

```
Command: (entdel ename)
Автолисп возвращает: <Entity name: 60000018>
```

```
Команда: (entdel ename)
Автолисп возвращает: <Имя примитива: 60000018>
```

Интересно, что если вы повторите эту операцию, то примитив будет восстановлен. Этот способ позволяет удалять из ГБД невидимые примитивы, что невозможно обычным выбором объектов (нельзя выбрать невидимый объект).

Для того чтобы модифицировать геометрические характеристики примитива непосредственно в ГБД , надо уметь находить в DXF-списке данных примитива (наша переменная EDATA) подсписки, в которых хранится нужная информация. Выбор и изменение различных данных, относящихся к примитиву, осуществляется по коду DXF (это всегда целое число) с помощью функций ASSOC и SUBST:

(assoc элементсписка сложныйсписок)

Извлекает из **сложногосписка** элемент списка по ключу **элементсписка**. Если **элементсписка** не найден, ASSOC возвращает nil.

При помощи этой функции можно извлечь, например, из списка **goods ((1 "car" "volvo")(2 "price" 80000))** подсписок **(2 "price" 80000)**:

(assoc 2 goods) возвращает список **(2 "price" 80000)**

(subst новыйэлемент старыйэлемент список)

Возвращает копию исходного списка с заменой всех найденных подсписков, идентичных **старомуэлементу**, на **новыйэлемент**. Если вхождений не обнаружено, SUBST возвращает копию старого списка (не nil!):

(subst '(2 "price" 100000) '(2 "price" 80000) goods)

возвращает:

((1 "car" "volvo")(2 "price" 100000))

Используя эту технику, попробуем извлечь из списка EDATA имя примитива.

```
Команда: (assoc 0 edata)
Автолисп возвращает: (0 . "LINE")
```

В приведенном выше примере мы фактически сказали Автолиспу: "Возврати мне подсписок с DXF-кодом 0". Автолисп просмотрел DXF-список примитива, нашел подсписок с кодом 0 и возвратил его. Разумеется, целиком. Полученный по коду подсписок все еще содержит DXF-код, который необходимо убрать: первый элемент списка, уже "отработал" свое и больше не понадобится. Для этой цели лучше всего использовать функцию CDR:

```
Команда: (cdr (assoc 0 edata))
Автолисп возвращает: "LINE"
```

Извлекая из DXF-списков нужную информацию, можно программно обрабатывать ее и затем, внеся изменения в DXF-список примитива при помощи функции SUBST, модифицировать ГБД при помощи функций ENTMOD и ENTUPD.

Приведем пример того, как можно из Автолиспа модифицировать вершину полилинии:

```
(setq e1 (entnext))              ; Переменной e1 присваивается имя полилинии
(setq e2 (entnext e1))           ; Переменной e2 присваивается имя
                                 ;   1-й вершины
(setq ed (entnext e2))           ; В списке ed данные о вершине 1
(setq ed                         ; Модификация переменной ed
    (subst (10 1.0 3.7)          ; Этот список помещается на место старого
           (assoc 10 ed)         ;   вхождения списка с первым элементом 10
    )                            ;
)
(entmod)                         ; Обновляет данные в ГБД
(entupd)                         ; Обновляет полилинию на экране
```

Заметим, что если вы попытаетесь извлечь данные о вставке блока (ВСТАВЬ, INSERT), то получите информацию о самой вставке, а не об определении блока.

Итак, для модификации примитива (группы примитивов) непосредственно в базе данных нужно:

- предложить пользователю выбрать примитив (функция ENTSEL) или группу примитивов (функция SSGET). Используя функцию SSGET, можно выбирать примитивы из ГБД программно, без обращения к пользователю - по их свойствам (тип примитива, слой и т.п.)
- извлечь из ГБД данные, относящиеся к этому примитиву (если использовался набор, то сначала следует извлечь имя примитива из набора - функция SSNAME, затем извлечь из ГБД данные, относящиеся к этому примитиву, - функция ENTGET)

- средствами работы со списками Автолиспа модифицировать данные о примитиве в переменной - списке данных (функции ASSOC, SUBST)
- модифицировать саму ГБД: внести изменения в чертеж (функции ENTMOD и ENTUPD)

Для того чтобы еще более полно работать с ГБД, надо уметь извлекать из нее не только информацию о примитивах, но и о блоках, слоях, типах линий и т.п.

Эту информацию Автокад хранит в справочных таблицах. Для работы с ними в Автокаде предусмотрена функ-ция TBLSEARCH, которая просматривает таблицу и выдает содержащуюся в ней информацию.

Функция TBLSEARCH имеет два аргумента: таблицу поиска и имя, по которому надо извлечь данные. В качестве примера введите с клавиатуры две строки:

```
Команда: (tblsearch "LAYER" "OBJECTS")
Автолисп возвращает:
((0 . "LAYER") (2 . "OBJECTS") (70 . 64) (62 . 3) (6 . "CONTINUOUS"))
```

Как уже говорилось, имена примитивов действительны только в текущем сеансе редактирования. Однако в Автолиспе есть средство, позволяющее узнавать примитивы из других сеансов редактирования. Это средство называется метки. Метки - это просто номер примитива в ГБД. Метки присваиваются примитивам автоматически в процессе создания (если были включены), и с ними можно работать из Автолиспа. Для этого в Автолиспе существует функция HANDENT:

(handent метка)

Возвращает имя примитива, который указывается строковым параметром **метка**. После того как получено имя примитива, оно может быть использовано для дальнейшей работы.

8.22. Перспективы Автолиспа

Написав на Автолиспе программу, содержащую много раз выполняемый цикл, вы тут же убедитесь в том, что Автолисп - медленный язык. Это связано с тем, что вычислитель Автолиспа - это интерпретатор, и в этом смысле подобен Бейсику. Иначе говоря, программа на Автолиспе - это, собственно, не программа, а данные для вычислителя Автолиспа. Выполняя, например, цикл, вычислитель Автолиспа многократно интерпретирует одни и те же строки программы, не "понимая" того, что эта работа уже была один раз проделана. Чтобы убедиться в этом, загрузите и выполните программу на Автолиспе fplot, входящую в комплект стандартной поставки Автокада. Если вы работаете только с функциями Автокада и вводом/выводом данных, то скорость вычислений не особенно критична, но как только возникает необходимость в большом объеме вычислений, Автолисп перестанет вас удовлетворять.

Значит ли это, что Автолисп исчерпал свои возможности? Оказывается, нет. Этот недостаток Автолиспа уже устранен с появлением компилятора Автолиспа.

Какие же преимущества имеет откомпилированная программа на Автолиспе по сравнению с обычной?

- В 15-20 раз повышается быстродействие как за счет собственно отказа от интерпретации программ, так и за счет увеличения быстродействия постраничной виртуальной памяти
- Загрузка программ в память происходит втрое быстрее, поскольку теперь при загрузке программы не требуется проверять ее на предмет наличия синтаксических ошибок
- В среднем вдвое уменьшаются требования к объему памяти как для кода программ, так и для данных - компилированные функции могут содержать больше данных
- Компилятор обеспечивает работу как со стандартным, так и с расширенным Автолиспом. Компилированные и интерпретированные функции могут комбинироваться как угодно и вызывать друг друга. Большинство имеющихся в настоящее время приложений на Автолиспе можно откомпилировать без каких-либо изменений, обеспечив тем самым резкое увеличение скорости выполнения как самых простых, так и наиболее сложных специализированных программ
- Используя новые возможности отладки программ при работе с расширенным Автолиспом, вы можете прервать выполнение сложной программы в любом месте, просмотреть или изменить значения переменных, выполнить некоторые отладочные операции и продолжить работу
- Как отметил Фил Ротуэлл, один из разработчиков системы Автокад, применение компилятора позволит вам не беспокоиться об авторских правах - защита ваших программ обеспечена

В заключение этого раздела следует упомянуть о новом пакете GLISP - генераторе параметрических программ, позволяющем автоматизировать создание однотипного чертежа с изменяющимися параметрами его элементов. Этот пакет имеет интерфейс с системой AutoCAD и СУБД ABASE. Генератор параметрических программ - это совершенно необходимое средство для реальной конструкторской работы. Опишем вкратце работу с пакетом GLISP.

Будем называть топологически подобными объекты, которые могут быть получены друг из друга без разрезания и склеивания контуров - простым изменением одного или нескольких размеров.

Конструктор, зная, что создаваемый им чертеж будет иметь различные размерные модификации, рисует в Автокаде шаблон такого объекта, специально объявляя размеры, по которым чертеж должен быть параметризован. После этого он вызывает из модифицированного меню Автокада пакет GLISP. В результате работы программы GLISP создается программа на Автолиспе, которая автоматически отрисовывает заданный конструктором шаблон, запрашивая пользователя ввести числовые значения размеров, которые были объявлены параметрами. Таким образом, созданная программа позволяет автоматически создавать черте-

жи целого класса объектов, топологически подобных заданному конструктором шаблону.

Имеется несколько вариантов ввода данных в программу: в интерактивном режиме (как с командной строки, так и из диалогового окна), в пакетном режиме (из файла) и из базы данных ABASE. Последний режим позволяет выбирать предварительно введенные наборы размеров, отвечающие какому-то стандарту. Эти расширения Автокада придают ему завершенный, законченный вид.

8.23. Заключение

В настоящей главе мы только коснулись поверхности океана возможностей, предоставляемых Автокадом, описали логику его построения и использование Автолиспа в работе с базой данных Автокада. Для более глубокого изучения Автолиспа обращайтесь к учебникам программирования на Автолиспе, выпущенным издательством Addison Wesley, - например, к книге Winston and Horn "LISP" (second edition) или прекрасному учебнику T.Hasemer "Looking at LISP". В качестве полезного справочника по Автолиспу можно рекомендовать также книгу "Автолисп. Версия 10. Руководство по программированию", изданную на русском языке.

В заключение мы приводим список полезных приемов, использование которых является признаком "хорошего стиля" работы на Автокаде:

- Используйте Автолисп для создания своих макроопределений меню, более сложных и "умных", чем стандартные
- По возможности вычисляйте значения параметров, старайтесь реже запрашивать пользователя ввести ту или иную величину - это уменьшит вероятность ошибок при вводе
- Старайтесь, чтобы создаваемые макроопределения содержали как можно больше подсказок - это также помогает уменьшить вероятность ошибок
- Используйте функции GET для ввода данных
- Старайтесь без необходимости не объявлять переменные глобальными - это засоряет память, создает дополнительную путаницу, уменьшает скорость выполнения функции
- Для использования функций Автокада используйте стандартную функцию COMMAND
- Помните, что, используя в функции DEFUN объявление C:, вы можете расширить список команд Автокада
- Используйте функции AND/OR для переключения вывода на экран разных сообщений по условию
- Старайтесь защититься от неправильного ввода функцией GETKWORD или, если это невозможно, организуйте цикл WHILE
- Делайте вашу программу нечувствительной к разнице между малыми и большими буквами, используя функции OR или STRCASE

ПРИЛОЖЕНИЕ А

СИСТЕМНЫЕ ПЕРЕМЕННЫЕ АВТОКАДА

Ниже перечислены системные переменные Автокада. Системные переменные содержат информацию о настройках различных режимов Автокада и некоторую дополнительную информацию. Системные переменные можно считывать и изменять (если переменная не помечена "только для чтения") при помощи команды SETVAR (УСТПЕРЕМ) или из Автолиспа (GETVAR и SETVAR). Системные переменные могут быть целого, вещественного, строкового типа или списком (тип каждой переменной указан в таблице). Большинство системных переменных Автокада записываются в файл чертежа. Если переменная записывается в файл конфигурации acad.cfg или вообще не сохраняется, это оговаривается особо.

Имя	*Описание*
ACADPREFICS	Строковая. Содержит имя каталога, в котором Автокад будет искать вспомогательные файлы, если они не будут найдены в текущем каталоге (файлы прототипов, шрифтов, меню, программ Автолиспа и вставляемых рисунков). Если, например, вы работаете с проектом NUM_1, настройки которого хранятся в каталоге D:\ACAD\NUM_1, то перед загрузкой Автокада из каталога, скажем, WORK следует поместить в командный файл запуска Автокада строку SET ACAD=D:\ACAD\NUM_1 и переменная ACADPREFIX получит значение "D:\ACAD\NUM_1" Только для чтения. Не сохраняется
ACADVER	Строковая. Содержит номер версии Автокада, который может принимать значение 10 или 10a. Эта переменная *не идентична* переменной $ACADVER заголовка DXF-файла, в которой содержится номер уровня базы данных рисунка Только для чтения
AFLAGS	Целая. Биты этой переменной определяют флаги режимов создания атрибутов командой ATTDEF (АТОПР). Числа, приведенные ниже, устанавливают соответствующие биты этой переменной. Для установки нескольких битов следует записать сумму нужных чисел в AFLAGS: 1 - Invisible (Скрытый) 2 - Constant (Постоянный) 4 - Verify (Контролируемый) 8 - Preset (Установленный) Не сохраняется

ANGBASE	Вещественная. Направление угла 0° в текущей ПСК
ANGDIR	Целая. Направление отсчета углов в текущей ПСК: -1 - по часовой стрелке, 0 - против часовой стрелки
APERTURE	Целая. Высота прицела объектной привязки в пикселах. Записывается в файл конфигурации
AREA	Вещественная. Значение площади, вычисленной с помощью команд AREA (ПЛОЩАДЬ), LIST (СПИСОК) или DBLIST (БДСПИСОК). Только для чтения. Не сохраняется
ATTDIA	Целая. Способ ввода значений атрибутов командой INSERT (ВСТАВЬ): 1 - через диалоговое окно, 0 - с командной строки
ATTMODE	Целая. Режим отображения атрибутов: 0 - все атрибуты невидимы, 1 - видимость определяется флагами, 2 - видимы даже скрытые атрибуты
ATTREQ	Целая. Режим присвоения значений атрибутам при вставке командой INSERT (ВСТАВЬ): 0 - ввод запрещен (берутся значения по умолчанию), 1 - разрешается ввод значений в командной строке или через диалоговое окно
AUNITS	Целая. Угловые единицы измерения: 0 - десятичные градусы, 1 - градусы/минуты/секунды, 2 - грады, 3 - радианы, 4 - топографические единицы
AUPREC	Целая. Количество всех десятичных разрядов при отображении значений углов
AXISMODE	Целая. Отображение осей: 1 - оси включены, 0 - оси отключены

AXISUNIT	Список из двух действительных чисел. Цена деления в условных единицах разметки осей X и Y (команда AXIS (ОСИ))
BACKZ	Вещественная. Задает смещение задней секущей плоскости для текущего видового экрана в условных единицах. Определена только в том случае, если включен бит "заднего сечения" переменной VIEWMODE. Расстояние до задней секущей плоскости от камеры может быть найдено вычитанием переменной BACKZ из расстояния от камеры до цели. Только для чтения
BLIPMODE	Целая. Отрисовка маркеров: 1 - включена, 0 - отключена
CDATE	Вещественная. Календарная дата/время (стандартный формат ОС UNIX). Только для чтения
CECOLOR	Строковая. Текущий цвет. Только для чтения
CELTYPE	Строковая. Текущий тип линии. Только для чтения
CHAMFERA	Вещественная. Длина первой фаски
CHAMFERB	Вещественная. Длина второй фаски
CLAYER	Строковая. Текущий слой. Только для чтения
CMDECHO	Целая. Отображение текста программы при выполнении функции Автолиспа: 1 - не подавляется, 0 - подавляется ("тихий" режим). Не сохраняется
COORDS	Целая. Режим отображения координат в строке состояний (устанавливается клавишей F6): 0 - отображаются только координаты точки указания, 1 - отображаются текущие абсолютные координаты перекрестья,

	2 - при запросе угла или расстояния отображаются угол и расстояние от последней введенной точки
DATE	Вещественная. Дата/время по Юлианскому календарю (стандартный формат ОС UNIX). Только для чтения. Не сохраняется
DIMxxx	Разные. Размерные переменные. Все размерные переменные могут обрабатываться как системные переменные Автокада (см. гл. 3)
DISTANCE	Вещественная. Расстояние, вычисленное командой DIST (ДИСТ). Только для чтения. Не сохраняется
DRAGMODE	Целая. Режим слежения: 0 - отключен, 1 - включен, 2 - Авто
DRAGP1	Целая. Частота регенерации объекта при слежении. Сохраняется в файле конфигурации
DRAGP2	Целая. Частота регенерации объекта при быстром слежении. Сохраняется в файле конфигурации
DWGNAME	Строковая. Имя рисунка. Может включать путь доступа операционной системы, если он был указан при вводе имени рисунка. Только для чтения. Не сохраняется
DWGPREFIX	Строковая. Путь доступа операционной системы. Только для чтения. Не сохраняется
ELEVATION	Вещественная. Текущий уровень
EXPERT	Целая. Режим отображения подсказок типа "вы уверены?": 0 - все подсказки, 1 - подавляются подсказки: "Выполнять регенерацию?" и "Вы действительно хотите отключить текущий слой?", 2 - подавляются перечисленные выше подсказки, а также: "Блок уже существует. Переопределить его?" (команда BLOCK (БЛОК) и "Рисунок с этим именем уже существует. Заменить его?" (команда WBLOCK (ПБЛОК), 3 - подавляются перечисленные выше подсказки, а также подсказки команды LTYPE (ТИПЛИН), если пользо-

	ватель пробует загрузить уже загруженный тип линии или создать новый тип линии в файле, в котором этот тип линии уже определен, 4 - подавляются перечисленные выше подсказки, а также подсказки команд UCS Save (ПСК Сохрани) и VPORTS Save (ВЭКРАН Запиши), если имя, вводимое пользователем, уже существует. Подавление подсказки означает, что на нее автоматически дается положительный ответ. Впоследствии будет возможно использовать значения этой переменной больше 4, тем самым расширяя круг подавляемых подсказок. Значение по умолчанию - 0. Не сохраняется
EXTMAX	Список из двух действительных чисел Правая верхняя граница рисунка. Уменьшается только командами ZOOM All (ПОКАЖИ Все) или ZOOM Extents (ПОКАЖИ Границы). Результат выдается в мировых координатах. Только для чтения
FILLETRAD	Вещественная. Задает радиус сопряжения
FILLMODE	Целая. Управление режимом закрашивания: 1 - включен, 0 - отключен
FLATLAND	Целая. Эта переменная служит для обеспечения совместимости трехмерных объектов, созданных старыми версиями Автокада, с данной версией: в последующем будет отменена. 1 - объектная привязка, DXF-формат и Автолисп соответствуют старым версиям; 0 - используются все новые возможности. По умолчанию для новых рисунков принимается значение 0, а для старых - значение 1
FRONTZ	Вещественная. Определяет смещение передней секущей плоскости для текущего видового экрана в условных единицах. Расстояние до передней секущей плоскости от точки зрения определяется вычитанием переменной FRONTZ из расстояния от точки зрения до цели. Определена, только если одновременно включены бит "передняя секущая плоскость" переменной VIEWMODE и бит "передняя секущая плоскость не в точке зрения". Только для чтения
GRIDMODE	Целая. Отображение сетки на текущем видовом экране: 1 - включена, 0 - отключена

GRIDUNIT	Список из двух действительных чисел. Задает интервалы сетки текущего видового экрана по X и Y
HANDLES	Целая. Режим присвоения меток примитивам: 0 - отключен, 1 - включен. Только для чтения
HIGHLIGHT	Целая. Выделение объектов при выборе: 1 - включено, 0 - отключено. Не сохраняется
INSBASE	Список из двух действительных чисел. Базовая точка вставки рисунка как блока в текущей ПСК (устанавливается также командой BASE (БАЗА))
LASTANGLE	Вещественная. Конечный угол последней введенной дуги в плоскости XY текущей ПСК. Только для чтения. Не сохраняется
LASTPOINT	Список из двух действительных чисел. Последняя введенная точка, координаты которой выражены в текущей ПСК (ссылка на нее выполняется знаком "@" во время ввода координат с клавиатуры). Не сохраняется
LASTPT3D	Список из трех действительных чисел. Аналогична LASTPOINT. Эта переменная не будет использоваться в последующих версиях
LENSLENGTH	Вещественная. Задает фокусное расстояние (в мм), используемое при построении перспективы для текущего видового экрана. Только для чтения
LIMMAX	Список из двух действительных чисел. Задает координаты правого верхнего угла лимитов, выраженные в мировых координатах
LIMMIN	Список из двух действительных чисел. Задает координаты левого нижнего угла лимитов, выраженные в мировых координатах
LTSCALE	Вещественная. Общий масштабный коэффициент типа линии

LUNITS	Целая. Определяет систему измерения линейных единиц: 1 - научная, 2 - десятичная, 3 - техническая, 4 - архитектурная, 5 - с дробной частью
LUPREC	Целая. Задает число десятичных разрядов или знаменатель в линейных единицах
MENUECHO	Целая. Биты этой переменной управляют эхо-выводом и подсказками меню. Числа, приведенные ниже, устанавливают соответствующие биты этой переменной. Для установки нескольких битов следует записать в MENUECHO сумму нужных чисел: 1 - подавляет эхо-вывод пунктов меню (аналогично переключателю ^P в макроопределениях меню), 2 - подавляет системные подсказки во время действия меню, 4 - подавляет переключатель ^P эхо-вывода пунктов меню, По умолчанию принимается значение 0 (выдаются все пункты меню и системные подсказки). Не сохраняется
MENUNAME	Строковая. Имя загруженного в данный момент файла меню. Включает путь доступа операционной системы, если он был указан. Только для чтения.
MIRRTEXT	Целая. Управление зеркальным отражением текста командой MIRROR (ЗЕРКАЛО): 0 - ориентация текста не меняется, не0 - зеркальное отображение текста
OSMODE	Целая. Биты этой переменной управляют режимами объектной привязки. Числа, приведенные ниже, устанавливают соответствующие биты этой переменной. Для установки нескольких битов следует записать в OSMODE сумму нужных чисел: 1 - ENDpoint (КОНточка) 2 - MIDpoint (СЕРедина) 4 - CENter (ЦЕНтр) 8 - NODe (УЗЕл) 16 - QUAdrant (КВАдрант) 32 - INTersection (ПЕРесечение) 64 - INSert (ТВСтавки) 128 - PERpendicular (НОРмаль) 256 - TANgent (КАСательная) 512 - NEArest (БЛИжайшая) 1024 - QUICK (БЫСтрый)

ORTHOMODE	Целая. Управление режимом ORTHO (ОРТО): 1 - включен, 0 - отключен
PDMODE	Целая. Задает символ отображения примитива "точка" (см. гл. 3)
PDSIZE	Вещественная. Размер символа отображения примитива "точка" (см. гл. 3)
PERIMETER	Вещественная. Периметр, вычисленный командами AREA (ПЛОЩАДЬ), LIST (СПИСОК), DBLIST (БДСПИСОК). Только для чтения. Не сохраняется
PICKBOX	Целая. Высота прицела выбора объекта в пикселах. Сохраняется в файле конфигурации
POPUPS	Целая. Признак наличия Расширенного Пользовательского Интерфейса (возможность работы с диалоговыми окнами, строкой меню и падающими меню): 1 - РПИ реализуется, 0 - текущий драйвер монитора не поддерживает РПИ. Только для чтения. Не сохраняется
QTEXTMODE	Целая. Режим отображения текста: 1 - контурный текст, 0 - нормальное отображение текста
REGENMODE	Целая. Автоматическая регенерация: 1 - включена, 0 - отключена
SCREENSIZE	Список из двух действительных чисел. Задает размер экрана графического монитора в пикселах, X и Y. Только для чтения. Не сохраняется
SKETCHINC	Вещественная. Шаг приращения в команде SKETCH (ЭСКИЗ)
SKPOLY	Целая. Команда SKETCH (ЭСКИЗ) генерирует: 0 - отрезки, 1 - полилинии
SNAPANG	Вещественная. Угол поворота сетки шаговой привязки (в текущей ПСК) для текущего видового экрана

SNAPBASE	Список двух действительных чисел. Начальная точка сетки шаговой привязки для текущего видового экрана (в текущей ПСК)
SNAPISOPAIR	Целая. Текущая плоскость изометрии текущего видового экрана: 0 - левая, 1 - верхняя, 2 - правая
SNAPMODE	Целая. Режим SNAP (ШАГ) для текущего видового экрана: 1 - включен, 0 - отключен
SNAPSTYL	Целая. Задает тип текущего видового экрана: 0 - стандартный, 1 - изометрический
SNAPUNIT	Список двух действительных чисел. Задает шаг привязки по X и Y текущего видового экрана
SPLFRAME	Целая. Управление отображением каркаса сплайна при сглаживании полилинии, определяющей сети поверхности сглаживания (отображается либо поверхность сглаживания, либо сеть), а также невидимых краев 3М граней: 0 - не отображаются, 1 - отображаются
SPLINESEGS	Целая. Задает число прямых сегментов, которые должны быть сгенерированы для каждого участка сплайна
SPLINETYPE	Целая. Тип сплайна, создаваемого командой PEDIT Spline (ПОЛРЕД Сплайн): 5 - квадратичный B-сплайн, 6 - кубический B-сплайн. Остальные значения запрещены
SURFTAB1	Целая. Задает число интервалов, которые должны быть сгенерированы в командах RULESURF (П-СОЕД) и TABSURF (П-СДВИГ), и плотность сети в направлении M для команд REVSURF (П-ВРАЩ) и EDGESURF (П-КРАЙ)
SURFTAB2	Целая. Задает плотность сети в направлении N для команд REVSURF (П-ВРАЩ) и EDGESURF (П-КРАЙ)
SURFTYPE	Целая. Тип поверхности сглаживания, создаваемой командой

	PEDIT Spline (ПОЛРЕД Сплайн): 5 - поверхность квадратичного B-сплайна, 6 - поверхность кубического B-сплайна, 8 - поверхность Безье. Остальные значения запрещены
SURFU	Целая. Плотность поверхности в направлении M
SURFV	Целая. Плотность поверхности в направлении N
TARGET	Список из трех действительных чисел. Задает положение цели в координатах ПСК на текущем видовом экране. Только для чтения
TDCREATE	Вещественная. Время и дата создания рисунка (стандартный формат ОС UNIX). Только для чтения
TDINDWG	Вещественная. Общее время редактирования (только для чтения, стандартный формат ОС UNIX)
TDUPDATE	Вещественная. Время и дата последнего редактирования/записи (только для чтения, специальный формат, стандартный формат ОС UNIX)
TDUSRTIMER	Вещественная. Таймер пользователя (только для чтения, стандартный формат ОС UNIX)
TEMPPREFIX	Строковая. Путь операционной системы, по которому Автокад помещает временные файлы. Если у вас есть возможность использовать виртуальный (электронный) диск, то, указав в этой переменной путь доступа к нему, вы можете ускорить работу Автокада. Эта переменная не может быть установлена из графического редактора (следует воспользоваться подменю настройки Главного Меню). Сохраняется в файле конфигурации
TEXTEVAL	Целая. Режим анализа ввода в тех запросах Автокада, где требуется ввод текстовых строк. Ввод выражений Автолиспа и передача значений переменных Автолиспа (знак "!"): 0 - недопустим (будет воспринят как текст), 1 - допустим (будут переданы Автолиспу). Следует иметь в виду, что команда DTEXT (ДТЕКСТ) с Автолиспом не работает вообще. Не сохраняется

TEXTSIZE	Вещественная. Задает высоту по умолчанию нового текста в текущей гарнитуре (если высота переменная)
TEXTSTYLE	Строковая. Имя текущей гарнитуры шрифта Только для чтения.
THICKNESS	Вещественная. Текущая высота примитивов
TRACEWID	Вещественная. Ширина полосы по умолчанию
UCSFOLLOW	Целая. Режим слежения за изменением системы координат (автоматическое переключение на вид в плане в новой системе): 1 - включен, 0 - выключен
UCSICON	Целая. Биты этой переменной управляют отображением пиктограммы системы координат текущего видового экрана. Для установки битов следует записать в UCSICON нужное число: 1 - пиктограмма отображается, 2 - если отображается, то привязана к началу текущей ПСК (если это возможно)
UCSNAME	Строковая. Содержит имя текущей системы координат. Только для чтения
UCSORG	Список из трех действительных чисел. Начало текущей системы координат в мировых координатах. Только для чтения
UCSXDIR	Список из трех действительных чисел. Задает направление X текущей ПСК. Только для чтения
UCSYDIR	Список из трех действительных чисел. Задает направление Y текущей ПСК. Только для чтения
USERI1-5	Пять свободных целочисленных переменных для использования сторонними разработчиками
USERR1-5	Пять свободных вещественных переменных для использования сторонними разработчиками
VIEWCTR	Список из двух действительных чисел. Центр текущего видового экрана в координатах ПСК. Только для чтения

VIEWDIR	Список из трех действительных чисел. Направление взгляда на текущем видовом экране в мировых координатах. Описывает положение камеры вектором смещения от точки цели. Только для чтения
VIEWMODE	Целая. Биты этой переменной управляют режимами отображения текущего видового экрана (1 - "Вкл."); Числа, приведенные ниже, в сумме определяют значение этой переменной: 1 - перспективный вид, 2 - переднее сечение, 4 - заднее сечение, 8 - переменная UCSFOLLOW равна 1, 16 - переднее сечение не в точке зрения. Если этот бит установлен в 1, то расстояние до переднего сечения (FRONTZ) определяет переднюю секущую плоскость. Если "Откл.", переменная FRONTZ игнорируется и передняя секущая плоскость проходит через точку установки камеры (т.е. векторы за камерой не отображаются). Этот флаг игноррируется, если бит "переднего сечения" (2) в состоянии "Откл." Только для чтения.
VIEWSIZE	Вещественная. Задает высоту изображения на текущем видовом экране в условных единицах. Только для чтения
VIEWTWIST	Вещественная. Задает угол поворота вида для текущего видового экрана. Только для чтения.
VPOINTX,Y,Z	Вещественные. Задают направляющие векторы направления взгляда по осям X, Y и Z в текущем видовом экране в мировых координатах. Таким образом, положение камеры описывается как вектор смещения от цели. Эти переменные не будут использоваться в следующих версиях, так как то же значение возвращается одной переменной VIEWDIR. Только для чтения
VSMAX	Список из двух действительных чисел. Задает правый верхний угол "виртуального экрана" текущего видового экрана, выраженный в координатах ПСК. Только для чтения
VSMIN	Список из двух действительных чисел. Задает левый нижний угол "виртуального экрана" текущего видового экрана, выраженный в координатах ПСК. Только для чтения

WORLDUCS	Целая. Совпадение текущей ПСК с МСК: 1 - совпадает, 0 - не совпадает. Только для чтения
WORLDVIEW	Целая. Система координат, в которой работают команды DVIEW (ДВИД) и VPOINT (ТЗРЕНИЯ), действующие в текущей ПСК: 1 - текущая ПСК, 0 - мировая система координат

ПРИЛОЖЕНИЕ В

ПЕРЕЧЕНЬ КОМАНД АВТОКАДА

Команда	*Описание*	*Опции*	
АПЕРТУРА (APERTURE)	Устанавливает размер прицела режима объектной привязки		
АТОПР (ATTDEF)	Определяет атрибуты	С	Управляет видимостью атрибутов
		П	Присваивает атрибуту статус "постоянный"
		К	Управляет режимом проверки
		У	Задает предварительные установки
АТРЕД (ATTEDIT)	Средство редактирования атрибутов		
АТЭКР (ATTDISP)	Управляет видимостью атрибутов на экране	В	Все атрибуты отображаются на экране
		О	Все атрибуты невидимы
		Н	Нормальный: видимость каждого атрибута определяется отдельно
БАЗА (BASE)	Задание базовой точки для последующей вставки блока в другой рисунок		
БЛОК (BLOCK)	Из группы объектов создает составной неделимый объект	?	Отображает на экране список определенных в рисунке блоков
БДСПИСОК (BDLIST)	Выводит на экран информацию, хранящуюся в базе данных обо всех объектах рисунка		

ВЕРНИ (REDO)	Восстанавливает изменения, сделанные предыдущей командой, если это были команды О или ОТМЕНИ		
'ВИД ('VIEW)	Сохраняет текущее изображение на экране как именованный вид или изображает вид на экране	У	Удаляет именованный вид из списка сохраненных
		В	Восстанавливает поименованный вид на экране
		С	Сохраняет изображение на экране как поименованный вид
		Р	Сохраняет как поименованный вид изображение в указанной рамке
		?	Список имен сохраненных видов
ВПАКЕТ (RSCRIPT)	Возобновляет исполнение прерванного пакета		
ВРЕМЯ (TIME)	Показывает время создания и редактирования рисунка и позволяет управлять таймером пользователя	П	Выводит на экран текущие значения таймера
		В	Включает таймер пользователя
		О	Останавливает таймер пользователя
		С	Сброс таймера пользователя
ВСЕРЕГЕН (REGENALL)	Регенерирует изображение на всех видовых экранах		
ВСТАВЬ (INSERT)	Вставляет копию блока в рисунок	имя	Вставляет целый рисунок "имя-файла-рисунка" в текущий рисунок
		имя=f	Создает блок из целого рисунка "имя-файла-рисунка"
		*имя	Сохраняет графические примитивы отдельных частей
		?	Выводит на экран список определенных в рисунке блоков (в ответ на запрос масштаба по оси X)

		Y	Определяет масштаб указанием двух точек (угловое задание масштаба) (в ответ на запрос масштаба по оси X)
		XYZ	Воспринимается как необходимость задания масштабов по осям X, Y, Z
ВЫБЕРИ (SELECT)	Составляет из выбранных объектов набор для последующего выполнения команд		
ВЭКРАН (VPORTS)	Разделяет графический экран на несколько частей, каждая из которых может содержать собственный вид рисунка	У	Удаляет из списка сохраненных указанную конфигурацию видовых экранов
		С	Соединяет (сливает) два видовых экрана в один
		В	Восстанавливает сохраненную конфигурацию видовых экранов
		З	Записывает имя текущей конфигурации экрана в список сохраненных
		О	Отключает видимость всех видовых экранов, кроме одного текущего
		2	Делит текущий видовой экран на 2 видовых экрана
		3	Делит текущий видовой экран на 3 видовых экрана
		4	Делит текущий видовой экран на 4 видовых экрана
		?	Список имен текущей и сохраненных конфигураций видовых экранов
'ГРАФЭКР ('GRAPHSCR)	При одноэкранной конфигурации переключает монитор в графический режим; команда употребляется в пакетах и при создании меню		
ДАКОМ (REDEFINE)	Восстанавливает встроенную команду, переопределенную с помощью команды НЕТКОМ		

ДВИД (DVIEW)	Определяет параллельные и перспективные проекции видов в динамическом режиме	К	Выбор угла поворота камеры относительно цели
		СЕ	Устанавливает заднюю и переднюю секущие плоскости
		Р	Устанавливает расстояние от камеры до цели, включает перспективу
		СК	Убирает скрытые линии в наборе
		ОТК	Отключает перспективное изображение
		ПА	Панорамирование рисунка
		Т	Задает точки расположения камеры и цели
		Ц	Вращает вид вокруг направления взгляда
		ОТМ	Отменяет действие субкоманды ДВИД
		Х	Прерывает выполнение команды ДВИД
		ПО	Зумирование или задание фокусного расстояния
'ДИАЛАТР ('DDLMODES)	Редактирование атрибутов через диалоговое окно		
'ДИАЛПРИМ ('DDEMODES)	Установка текущего слоя, цвета, типа линии и высоты через диалоговое окно		
ДИАЛПСК (DDUCS)	Изображает на экране диалоговое окно для управления ПСК		
'ДИАЛСЛОЙ ('DDLMODES)	Установка характеристик слоя через диалоговое окно		
'ДИАЛСРЕД ('DDRMODES)	Установка режимов рисования через диалоговое окно		

ДИСТ (DIST)	Определяет расстояние между двумя точками		
ДОС (SHELL)	Предоставляет возможность запуска других программ без выхода из Автокада		
ДОС1 (SH)	Предоставляет доступ к внутренним командам ДОС без выхода из Автокада		
ДСЛАЙД (MSLIDE)	Записывает изображение на экране в слайд-файл		
ДУГА (ARC)	Рисует дугу любого радиуса; метод задания дуги, принимаемый по умолчанию, заключается в указании на две ее конечные точки и на точку, принадлежащую дуге	У Ц Н К Д Р RETURN	Центральный угол Центр Направление Конец Длина хорды Радиус (при нажатии в ответ на запрос "Центр") воспринимается как указание совместить начало дуги с концом последней нарисованной дуги или отрезка
ЕДИНИЦЫ (UNITS)	Выбор формата и точности представления координат и углов		
ЗАГРУЗИ (LOAD)	Загружает файл, содержащий описание определенных пользователем форм	?	Выдает на экран список загруженных файлов описания форм
ЗАДЕРЖИ (DELAY)	Приостанавливает выполнение команды на указанное время; употребляется в пакетных файлах		

ЗАКРАСЬ (FILL)	Определяет, будут ли закрашены изображаемые на экране или выводимые на бумагу полосы, фигуры и полилинии	В О	Полосы, фигуры и полилинии закрашены Полосы, фигуры и полилинии только оконтурены
ЗЕРКАЛО (MIRROR)	Рисует зеркальное отображение указанного объекта		
ИЗМЕНИ (CHANGE)	Позволяет изменять положение, размер и ориентацию примитивов в пространстве и другие характеристики		
ИЗОМЕТР (ISOPLANE)	Задает изометрическую плоскость для рисования	Л П В RETURN	Левая плоскость Правая плоскость Верхняя плоскость Делает следующую по очереди плоскость текущей
ИМПОРТА (DXFIN)	Загружает файл обмена рисунками		
ИМПОРТД (DXBIN)	Вставляет закодированный в двоичном формате файл в рисунок		
ИМПОРТИ (IGESIN)	Загружает обменный файл в формате IGES		
КОЛЬЦО (DONUT)	Рисует кольца с заданными внешним и внутренним диаметрами		
КОНЕЦ (END)	Выход из рисунка с сохранением внесенных изменений		
КООРД (ID)	Выдает на экране в зоне подсказки координаты указанной точки		

КОПИРУЙ (COPY)	Рисует копии указанного объекта	Н	Создает несколько копий указанного объекта
КРУГ (CIRCLE)	Рисует круг любого заданного радиуса; по умолчанию принят метод задания круга по центру и точке на его окружности	2Т	Задает круг по двум конечным точкам на его диаметре
		3Т	Задает круг по трем точкам на его окружности
		Д	Для указания диа-метра вместо запрашиваемого радиуса
		ККР	Задание круга указанием двух касательных и радиуса
КТЕКСТ (QTEXT)	При включенном режиме на экране появляются не символы, а контуры текста	В	Режим контурного текста включен
		О	Режим контурного текста отключен
ЛИМИТЫ (LIMITS)	Изменяет лимиты рисунка и проверяет соблюдение этих лимитов	2точки	Устанавливает лимиты по нижнему левому и верхнему правому углам рисунка
		В	Включает проверку лимитов
		О	Отключает проверку лимитов
ЛМАСШТАБ (LTSCALE)	Устанавливает масштаб для всех типов линий, появляющихся в рисунке		
МАРКЕР (BLIPMODE)	Управляет видимостью маркеров на экране при указании точек	В	На экране появляются временные маркеры
		О	Маркеры не появляются
МАССИВ (ARRAY)	Производит многократное копирование выбранного объекта, располагая копии прямоугольным или круговым массивом	К	Круговой массив
		П	Прямоугольный массив
МАСШТАБ (SCALE)	Пропорционально изменя-ет размеры существующих объектов	С	Изменение размера со ссылкой на величину существующего объекта

МВСТАВЬ (MINSERT)	Вставляет массив копий блока в рисунок	имя	Вставляет целый рисунок "имя-файла-рисунка" и формирует из вставленных рисунков массив
		имя=f	Создает блок из целого рисунка "имя-файла-рисунка" и формирует из созданных блоков массив
		?	Выводит на экран список определенных в рисунке блоков (как ответ на запрос масштаба по оси X)
		У	Определяет масштаб указанием двух точек (Угловое задание масштаба) (как ответ на запрос масштаба по оси X)
		XYZ	Воспринимается как необходимость задания масштабов по осям X, Y, Z
		МЕНЮ	Загружает в рисунок (MENU) файл меню, содержащий команды Автокада (экранное меню, падающее меню, планшетное меню и кнопочные меню)
МЕТКИ (HANDLES)	Присваивает уникальный постоянный номер каждому примитиву рисунка	В	Присваивает метки всем примитивам и устанавливает переменную HANDLES равной 1
		У	Удаляет ранее присвоенные примитивам метки
МНОГОРАЗ (MULTIPLE)	Вызывает повторение последующей за этой спецификацией команды до тех пор, пока она не будет прервана		
МН-УГОЛ (POLYGON)	Рисует правильные многоугольники с заданным числом сторон	С	Задание многоугольника указанием одной его стороны
		О	Описанный вокруг окружности
		В	Вписанный в окружность

НАСТРВИД (VIEWRES)	Заданием числа сторон изображающего на экране круг многоугольника определяется точность изображения и скорость регенерации кругов и дуг		
НЕТКОМ (UNDEFINE)	Переопределяет встроенную команду		
НОВОЕИМЯ (RENAME)	Изменяет имена гарнитур шрифтов, слоев, типов линий, блоков и видов ПСК и конфигураций видовых экранов	Б С Т Г П ВИ ВЭ	Изменяет имя блока Изменяет имя слоя Изменяет имя типа линии Изменяет имя гарнитуры Изменяет имя ПСК Изменяет имя вида Изменяет имя конфигурации видовых экранов
ОБРЕЖЬ (TRIM)	Удаляет части примитивов, которые пересекают указанную границу		
ОЙ (OOPS)	Восстанавливает стертые ранее примитивы		
ОРТО (ORTHO)	Включает режим рисования примитивов только параллельно сетке		
'ОСВЕЖИ ('REDRAW)	Перерисовывает текущий видовой экран		
ОСИ (AXIS)	Изображает на графическом экране "направляющие линии"	О В Ш А число	Отключает видимость направляющих линий Включает видимость направляющих линий Устанавливает интервал делений, равный шагу привязки Устанавливает цену деления по оси X, отличную от цены деления по оси Y Устанавливает цену деления (0=устанавливается равной шагу привязки)

		числоX	Устанавливает цену деления кратно шагу привязки
О (U)	Отменяет действие предыдущей команды		
ОТМЕНИ (UNDO)	Отменяет действие нескольких команд; имеет более широкие возможности, чем команда О	число	Отменяет действие указанного количества команд, начиная с последней
		А	Авто: любая операция из меню интерпретируется командами О и ОТМЕНИ как один шаг
		О	Обратно: восстанавливает состояние рисунка на момент проставления контрольной точки
		У	Управление: определяет количество шагов, запоминаемых командами О и ОТМЕНИ
		К	Конец: последовательность команд, начиная с метки
		Г	"Группа" и заканчивающаяся меткой "КОНЕЦ" воспринимается командами О и ОТМЕНИ как одна команда
		М	Метка: устанавливает метку в протоколе рисунка
ОТРЕЗОК (LINE)	Рисует прямолинейный отрезок указанной длины	RETURN	(как ответ на запрос "От точки") начало изображаемого отрезка совмещается с концом последней нарисованной дуги или отрезка (как ответ на запрос "К точке:")
		З	замыкает ломаную (как ответ на запрос "К точке:")
		О	отменяет последний нарисованный сегмент
П-ВРАЩ (REVSURF)	Создает трехмерную многоугольную сеть аппроксимацией поверхности вращения, полученной в результате вращения кривой вокруг заданной оси		

П-КРАЙ (EDGESURF)	Создает трехмерную многоугольную сеть методом аппроксимации участка поверхности Кунса (бикубическая поверхность, полученная интерполяцией между четырьмя краями)		
П-СОЕД (RULESURF)	Создает трехмерную многоугольную сеть аппроксимацией поверхности между двумя кривыми		
ПАКЕТ (SCRIPT)	Выполняет указанный командный пакет		
'ПАН ('PAN)	Перемещает окно экрана по рисунку		
ПЕРЕНЕСИ (MOVE)	Параллельный перенос выбранного объекта		
ПЕЧАТАЙ (PRPLOT)	Выводит рисунок на матричный принтер		
ПБЛОК (WBLOCK)	Записывает изображение указанных объектов в отдельный файл на диск	имя = * RETURN	Запись определенного ранее блока в отдельный файл Присвоение файлу имени выписываемого блока Запись на диск всего рисунка Запись указанных объектов
ПЛАН (PLAN)	Устанавливается точка зрения (0,0,1)	Т П М	План текущей ПСК План указанной ПСК План МСК
ПЛАНШЕТ (TABLET)	Настраивает планшет для точного копирования изображения на бумаге средствами Автокада	В О К Н	Включает режим "Планшет" Отключает режим "Планшет" Калибровка планшета Настройка зон планшетного меню

ПЛИНИЯ (PLINE)	Рисует двумерные полилинии (последовательности из прямолинейных и дуговых сегментов с возможным указанием их ширины)	П	Устанавливает новую полуширину
		ОТМ	Отменяет последний нарисованный сегмент
		Ш	Устанавливает ширину
		RETURN	Выход из команды ПЛИНИЯ
		В режиме отрезка:	
		ДУ	Переключение в режим дуги
		З	Замыкание прямолинейным сегментом
		ДЛ	Длина сегмента (продолжает предыдущий сегмент)
		В режиме дуги:	
		У	Центральный угол
		Ц	Центральная точка
		З	Замыкание прямолинейным сегментом
		Н	Направление
		ОТР	Длина хорды или переключение в режим отрезка
		Р	Радиус
		В	Вторая точка дуги по трем точкам
ПЛОЩАДЬ (AREA)	Вычисляет площадь многоугольника, замкнутой полилинии или круга	Д	Устанавливает режим добавления
		В	Устанавливает режим вычитания
		П	Вычисляет площадь выбранного объекта
ПОВЕРНИ (ROTATE)	Поворачивает существующий объект	С	Поворот относительно существующего угла
ПОДОБИЕ (OFFSET)	Создает подобные существующим кривые и параллельные линии	число	Определение расстояния смещения
		Т	Точка, через которую будет проходить линия, подобная заданной
ПОЛОСА (TRACE)	Рисует закрашенные линии заданной ширины		

ПОЛРЕД (PEDIT(2D))	Редактирует двумерные полилинии	З	Замыкает открытую полилинию
		У	Убирает сглаживание, т.е. возвращает полилинию после применения сглаживания в первоначальное состояние
		В	Редактирует вершины
		СГ	Сглаживает полилинию, делая из ломаной кривую
		Д	Добавляет к существующей полилинии
		РА	Разрывает в указанной вершине замкнутую полилинию
		СП	Использует вершины полилинии как фрейм для сплайн-аппроксимации (тип определяется значением переменной SPLINETYPE)
		О	Отменяет последнее действие редактирования
		Ш	Устанавливает ширину всех сегментов
		Х	Выход из ПОЛРЕД
		В режиме редактирования вершин:	
		РА	Устанавливает первую вершину для разрыва
		В	Выполни (разрыв или выпрямление)
		ВС	Восставляет новую вершину после текущей
		ПЕ	Переносит текущую вершину в указанную точку
		С	Устанавливает следующую по порядку вершину текущей
		П	Устанавливает предыдущую вершину текущей
		РЕ	Регенерирует изображение полилиний
		ВЫ	Устанавливает первую вершину для выпрямления
		К	Устанавливает касательное направление для текущей вершины
		Ш	Устанавливает новую ширину для следующего сегмента
		Х	Выход из команды или прерывание процесса выпрямления или сглаживания

ПОЛРЕД(3М) (PEDIT(3D))	Редактирование трехмерных полилиний	З	Замыкает открытую полилинию
		У	Убирает сглаживание
		В	Редактирование вершин (субкоманды описаны ниже)
		РА	Разрывает в указанной вершине замкнутую полилинию
		СП	Использует вершины полилинии как каркас для сплайн-аппроксимации (тип определяется значением переменной SPLINETYPE)
		О	Отменяет последнее действие редактирования
		Х	Выход из команды ПОЛРЕД
		В режиме редактирования вершин:	
		РА	Устанавливает первую вершину для разрыва
		В	Выполни (разрыв или выпрямление)
		ВС	Вставляет новую вершину после текущей
		ПЕ	Переносит текущую вершину в указанную точку
		С	Устанавливает следующую по порядку вершину текущей
		РЕ	Регенерирует полилинию
		ВЫ	Устанавливает первую вершину для выпрямления
		Х	Выход из процесса редактирования или прерывание выпрямления или сглаживания
ПОЛРЕД (сеть) (PEDIT) (Mesh)	Редактирует трехмерные многоугольные сети	У	Убирает сглаживание и восстанавливает первоначальный вид сети
		В	Редактирует вершины полилинии
		М	Размыкает или замыкает сеть в направлении M
		Н	Размыкает или замыкает сеть в направлении N
		С	Сглаживает поверхность (тип определяется значением SURFTYPE)
		О	Отменяет последнюю операцию редактирования

		X	Выход из команды ПОЛРЕД
'ПОКАЖИ ('ZOOM)	Уменьшает или увеличивает изображение объекта на экране	число	Масштаб по отношению к первоначальному изображению объекта на экране
		числоX	Масштаб по отношению к текущему изображению объекта на экране
		В	Все
		Ц	Центр
		Д	Динамика
		Г	Границы
		Л	Нижний левый угол
		П	Предыдущий
		Р	Рамка
ПОКИНЬ (QUIT)	Выход из графического редактора и возвращение в Главное Меню без записи сделанных за сеанс изменений		
'ПОМОЩЬ ('HELP)	Выводит на экран список команд и форматы ввода данных; можно применять для получения справки по указанной команде		
ПРИВЯЖИ (OSNAP)	Точное совмещение указываемых точек с характерными точками объектов	ЦЕН	к центру дуги или круга
		КОН	к ближайшей конечной точке отрезка или дуги
		ТВС	к точке вставки текста, блока, формы
		ПЕР	к точке пересечения отрезков, дуг, окружностей
		СЕР	к середине дуги, отрезка
		БЛИ	к ближайшей видимой точке дуги, круга, отрезка или точке
		УЗЕ	к узлу (точке)
		НИЧ	ничего (Откл)
		НОР	нормаль к дуге, отрезку или кругу
		КВА	к квадранту дуги или круга
		БЫС	быстрый режим (возвращается не ближайшая к перекрестью, а первая найденная точка)
		КАС	касательная к дуге или окружности

ПРОДОЛЖИ ('RESUM)	Продолжает прерванный процесс выполнения пакета		
ОДЕЛИ (DIVIDE)	Делит объект на заданное число частей, помечая их маркерами	Б	Вместо маркера вставляется указанный блок
ПСК (UCS)	Определяет или видоизменяет текущую ПСК	У	Удаляет одну или более записанную систему
		О	Устанавливает ПСК с тем же направлением выдавливания, что и у указанного объекта
		Н	Перемещает точку отсчета текущей ПСК
		П	Делает предыдущую ПСК текущей
		З	Заменяет текущую ПСК на записанную ранее
		С	Сохраняет текущую ПСК
		В	Поворачивает текущую ПСК, совмещая направление взгляда с направлением оси Z
		М	Устанавливает текущую ПСК, эквивалентную МСК
		X	Поворачивает текущую ПСК вокруг ее оси X
		Y	Поворачивает текущую ПСК вокруг ее оси Y
		Z	Поворачивает текущую ПСК вокруг ее оси Z
		ZO	Задание ПСК по точке отсчета и положительному направлению оси Z
		3	Задание ПСК по трем точкам: началу отсчета, точке на положительном направлении оси X и аналогичной точке на оси Y
		?	Список имен сохраненных ПСК
РАЗМЕР (DIM)	Выход в режим проставления размеров, позволяющий дополнить рисунок размерными и выносными линиями		

РАЗМЕР1 (DIM1)	Вход в режим проставления одного размера, затем выход в обычный командный режим		
РАЗМЕТЬ (MEASURE)	Ставит метки на выбранном объекте через указанные интервалы	Б	Вместо метки будет вставлен указанный блок
РАЗОРВИ (BREAK)	Стирает часть объекта или разрывает его на две части	П	Переопределяет первую указанную точку
РАСТЯНИ (STRETH)	Позволяет растягивать объект, перемещая одну из его частей, но не разрывая его		
РАСЧЛЕНИ (EXPLODE)	Расчленяет блок или полилинию на составляющие элементы, не изменяя геометрии объекта, но удаляя определение блока или полилинии		
РЕГЕН (REGEN)	Регенерирует изображение в текущем видовом экране		
РЕГЕНАВТО (REGENAUTO)	Определяет, будет ли произведена регенерация, обусловленная выполнением другой команды	В	Автоматическая регенерация без сообщений. Перед регенерацией в зоне подсказок появляется запрос
СВОЙСТВА (CHPROP)	Изменяет свойства выбранных объектов	Ц Т СЛ В	Цвет Тип линии Слой Высота
СЕТКА (GRID)	Изображает на экране сетку из точек с заданным расстоянием между ними	В О Ш А число числоX	Включает видимость сетки Отключает видимость сетки Устанавливает интервал равным шагу привязки Устанавливает отличные друг от друга интервалы по осям Устанавливает интервал сетки (0=шаг привязки) Задает интервал сетки кратным шагу привязки

СКРОЙ (HIDE)	Удаляет скрытые линии на трехмерном изображении объекта		
СЛАЙД (VSLIDE)	Изображает на экране содержимое ранее созданного слад-файла	имя	Показывает на экране содержимое файла "имя"
		*имя	Загружает слайд-файл для употребления следующей командой СЛАЙД
СЛЕДИ (DRAGMODE)	Включает и отключает режим динамического отслеживания, применяющийся при выполнении некоторых команд	В	Ввод требований "СЛЕДИ" при необходимости
		О	Игнорирование требования "СЛЕДИ"
		А	Устанавливает автоматический режим: отслеживание включается без запроса
СЛОЙ (LAYER)	Создает в текущем рисунке именованные слои и присваивает им цвет и тип линии	Ц имя	Присваивает указанному слою цвет "имя"
		З а,б	Замораживает слои "а" и "б"
		Т t	Присваивает указанному слою тип линии "t"
		С а	Делает текущим слой "а"; если такого слоя не существует, создает его
		Н а.б	Создает новые слои "а" и "б"
		В а.б	Включает видимость слоев "а" и "б"
		О а.б	Отключает видимость слоев "а" и "б"
		У а	Устанавливает текущим существующий слой "а"
		Р а.б	Размораживает слои "а" и "б"
		?	Выдает на экран список определенных в рисунке слоев, их состояние, цвет и тип линии
СОПРЯГИ (FILLET)	Построение сопряжения двух отрезков, дуг или кругов заданного радиуса	ПОЛ	Сопряжение между собой отдельных сегментов одной полилинии
		РАД	Установка радиуса сопряжения
СОТРИ (ERASE)	Удаляет примитивы из рисунка		

СОХРАНИ (SAVE)	Записывает сделанные в сеансе редактирования изменения в файл, не выходя из графического редактора		
СПИСОК (LIST)	Выводит на экран информацию о выбранном объекте, содержащуюся в базе данных рисунка		
СТАТУС (STATUS)	Выводит на экран текущую информацию о рисунке и режимах		
СТИЛЬ (STYLE)	Создание новых гарнитур шрифта, пользователю предоставляется возможность выбрать шрифт, его направление, наклон и степень растяжения	?	Выводит на экран список гарнитур, определенных в рисунке
ТЕКСТ (TEXT)	Рисует на экране текстовые символы заданного размера и начертания	ВПИ	Текст заданной гарнитуры, вписанный между двумя точками; высота определяется автоматически
		Ц	Горизонтальное центрирование строки текста
		ВЫ	Строка текста заданной высоты выравнивается между двумя точками; соответствующая степень растяжения/сжатия вычисляется автоматически
		С	Горизонтальное и вертикальное центрирование строки текста
		ВПР	Строка текста выравнивается по правому краю
		Г	Выбор гарнитуры шрифта
'ТЕКСТЭКР ('TEXTSCR)	При одноэкранной конфи-гурации переключает в текстовый режим монитор; команда употребляется в пакетных файлах и при создании меню		

ТЗРЕНИЯ (VPOINT)	Выбор в пространстве точки зрения на объект	П	Определение точки зрения заданием двух углов поворота
		RETURN	Выбор точки зрения поворотом тройки осей координат
		x,y,z	Задание точки зрения ее координатами
ТИПЛИН (LINETYPE)	Определение типа линии как последовательности штрихов, точек и пробелов; загрузка его из библиотеки и присвоение последующим нарисованным объектам	?	Вывод на экран списка имен типов линий, содержащихся в библиотеке
		С	Создание определения типа линии
		З	Загрузка определения типа линии
		У	Установка типа линии текущим для последующего рисования
ТОЧКА (POINT)	Рисует отдельные точки		
УДАЛИ (PURGE)	Удаляет из рисунка ненужные блоки, формы, гарнитуры шрифтов, слои, блоки и типы линий	В	Удаляет все ненужные поименованные объекты
		Б	Удаляет ненужные блоки
		С	Удаляет ненужные слои
		Т	Удаляет ненужные типы линий
		Ф	Удаляет ненужные формы
		Г	Удаляет ненужные атрибуты
УДЛИНИ (EXTEND)	Удлиняет отрезок, дугу или полилинию до пересечения с другим объектом		
УРОВЕНЬ (ELEV)	Устанавливает уровень и высоту для вновь создаваемых объектов		
'УСТПЕРЕМ ('SETVAR)	Дает возможность вывести на экран или изменить значения системных переменных		

ФАЙЛЫ (FILES)	Осуществляет работу с файлами на диске		
ФАСКА (CHAMFER)	Снимает фаску с угла, образованного пересечением двух отрезков	Д ПОЛ	Устанавливает длину фасок Снимает фаску с полилинии
ФИГУРА (SOLID)	Рисует закрашенные многоугольники		
ФИЛЬМ (FILMROOL)	Создает файл для дальнейшей его загрузки в АВТОШЕЙД		
ФОРМА (SHAPE)	Рисует ранее определенные формы	?	Список имен доступных форм
ЦВЕТ (COLOUR)	Установка цвета примитива	число имя ПОБ ПОС	Устанавливает цвет объекта; Устанавливает цвет объекта; стандартное имя цвета Присваивает цвет "ПОБлоку" Присваивает цвет "ПОСлою"
ЧЕРТИ (PLOT)	Вывод рисунка на плоттер		
ШАГ (SNAP)	Устанавливает шаг для ввода точек с помощью дигитайзера, что облегчает указание и делает его точным	число В О А П Т	Устанавливает шаг привязки Включает режим ШАГ Отключает режим ШАГ Устанавливает различный шаг по разным осям Поворачивает сетку шаговой привязки Выбор типа изометрического стиля
ШТРИХ (HATCH)	Осуществляет штриховку и заполнение областей	имя С ?	Использует для штриховки образец, имя которого указано Использует для штриховки простой образец, создаваемый пользователем Выводит на экран названия имеющихся образцов

ЭКСПОРТА (DXFOUT)	Записывает файл обмена рисунками на диск	Д	Записывает файл в двоичном формате
		О	Только выбранные объекты
ЭКСПОРТИ (IGESOUT)	Записывает на диск обменный файл в формате IGES		
ЭЛЛИПС (ELLIPSE)	Рисует эллипсы	Ц	По центру
		П	По эксцентриситету указанием угла поворота вокруг второй оси
		И	Рисует круг на текущей изометрической плоскости
ЭСКИЗ (SKETCH)	Выполнение эскизов "от руки"	Р	Продолжает эскизную линию, начиная с конечной точки предыдущей линии
		С	Стирает временные (незаписанные) эскизные линии
		П	Поднимает/опускает перо
		О	Выход из команды с отменой временных линий
		З	Запись временных линий без выхода из команды
		Х	Запись временных линий с последующим выходом из команды
		.	Рисует линию до указанной устройством указания точки
3-ГРАНЬ (3DFACE)	Рисует участок трехмерной плоскости	Н	Делает указанный край невидимым
3-ПОЛИ (3DPOLY)	Создание трехмерных полилиний	З	Замыкает открытые полилинии
		О	Отменяет (удаляет) последний нарисованный сегмент
		RETURN	Выход из команды
3-СЕТЬ (3DMESH)	Задает трехмерную многоугольную сеть указанием ее размера (в терминах M,N) и положения каждой ее вершины		

ПРЕДМЕТНЫЙ УКАЗАТЕЛЬ

А

Б

В

Г

Д

Е

З

И

К

Л

М

Н

О

П

Р

С

Т

У

Ф

Ц

Ч

Ш

Э

ОГЛАВЛЕНИЕ

"ДИАЛОГ-МИФИ" ПРЕДЛАГАЕТ

ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ АВТОЛИСП В СИСТЕМЕ САПР АВТОКАД.
С. Гладков

Учебно-справочное пособие по программированию. Ориентировано на все категории пользователей. В нем описывется структура языка, распределение памяти и обращение к графической базе данных Автокада, дается методика написания и примеры программ. Рассмотрены вопросы написания макроопределений, меню Автокада и настройки рабочей среды. Содержит тематический справочник и предметный указатель функций Автолиспа. По дополнительному запросу поставляется дискета, расширяющая примеры, данные в книге.

Объем книги около 120 стр. Планируемая цена 10 руб.

ПРОГРАММИРОВАНИЕ В MICROSOFT WINDOWS.
Для системных программистов, аспирантов, студентов вузов.
С. Гладков, Г. Фролов

Учебно-справочное пособие по созданию программ ("приложений"), работающих под управлением операционной оболочки MS-Windows 2 x, 3 x Рассматривается положение MS-Windows среди системного программного обеспечения, ее внутренняя структура Обсуждаются вопросы установки среды программирования, механизмы взаимодействия приложения с операционной оболочкой, алгоритм создания программы. Раскрываются аспекты программирования в Windows: структура программы, работа с экраном, клавиатурой, мышью, графика, шрифты и пр. Полный предметный указатель позволяет использовать книгу как справочник функций MS-Windows. Дополнительно можно преобрести дискету с примерами программ.

Объем книги около 400 стр.

БИБЛИОТЕКА СИСТЕМНОГО ПРОГРАММИСТА.
Для системных программистов, аспирантов, студентов вузов.
А. Фролов, Г. Фролов

Том 1. ОПЕРАЦИОННАЯ СИСТЕМА MS-DOS. В 3 книгах, книги 1 и 2.

Учебно-справочное руководство. От аналогичных изданий отличается более глубоким изложением материала. Описываются детали, которые часто остаются скрытыми даже для опытного программиста: внутренняя структура и организация работы MS-DOS, недокументированные прерывания Разбираются профессиональные приемы работы. Приводятся подробно комментируемые исходные тексты программ.

Том 1. ОПЕРАЦИОННАЯ СИСТЕМА MS-DOS. Книга 3.

Информация о работе с дисками различными способами - от программирования контроллера до высокоуровневых средств, входящих в состав библиотек трансляторов Microsoft C. Подробно рассмотрена логическая структура дисков в MS-DOS. Описаны способы защиты данных от несанкционированного доступа и копирования, защиты от отладки. Приводится много программ на языке C.

Том 2. АППАРАТНОЕ ОБЕСПЕЧЕНИЕ КОМПЬЮТЕРА IBM PC.

Подробно описаны клавиатура, мышь, таймер, часы реального времени, асинхронный адаптер, порт параллельной передачи данных, контроллер прямого доступа к памяти. Большое внимание уделено использованию расширенной и дополнительной памяти. Для описанных устройств приводится методика программирования на всех уровнях - от использования портов ввода/вывода до высокоуровневых средств стандартных библиотек трансляторов Microsoft QuickC 2 5 и C 6 0. Книга содержит большое количество примеров, составленных на языках ассемблера и C

Том 3. ДИСПЛЕЙНЫЕ АДАПТЕРЫ CGA/EGA/VGA.

Подробное описание архитектуры и программирования дисплейных адаптеров CGA/EGA/VGA. Описано использование регистров видеоадаптеров, функций их обслуживания, входящие в состав BIOS Приведены основные графические функции стандартных библиотек трансляторов Microsoft QuickC 2.5 и C 6 0 Книга содержит большое количество примеров, составленных на языках ассемблера и C

Том 4. ИНСТРУМЕНТАЛЬНЫЕ СРЕДСТВА ПРОГРАММИСТА.

Описаны инструментальные средства разработки программного обеспечения Рассмотрены этапы создания и сопровождения программного продукта, описаны утилиты и программные

средства, предназначенные для программистов - QuickC 2.5, C 6.0, MASM 5.1, CodeView и др. Приведены тексты утилиты DMAKE (прилагается дискета), которая предназначена для автоматизирования процесса разработки и сопровождения программных комплексов.

Объем каждого издания около 240 стр. Планируемая цена 20 руб.

ПРАКТИЧЕСКАЯ РАБОТА НА КОМПЬЮТЕРАХ СЕМЕЙСТВА IBM PC В ОПЕРАЦИОННОЙ СРЕДЕ MS-DOS 4.01.

Ю. Максимов, С. Осипов, О. Симоненков.

Учебное пособие предназначено для пользователей, желающих в краткие сроки овладеть навыками работы на персональных компьютерах семейства IBM PC в операционной среде MS-DOS 4.01. Главы пособия, содержащие сведения о персональных компьютерах, файловой системе и основных командах MS-DOS, а также об операционной оболочке MS-DOS Shell, будут полезны для начинающих и для более опытных пользователей. Главы, посвященные операционным системам персональных компьютеров, структуре и функциям операционной системы MS-DOS и командным файлам, ориентированы главным образом на пользователей, уже имеющих навыки работы на персональных компьютерах.

Объем издания 160 стр. Цена 10 руб.

КУРС ПРАКТИЧЕСКОЙ РАБОТЫ С ЭЛЕКТРОННЫМИ ТАБЛИЦАМИ MICROSOFT EXCEL.

С. Зинькевич, И. Моторина, А. Цыганов.

Описание средств пакета Microsoft Excel - самой мощной (среди программных продуктов такого класса) электронной таблицы с деловой графикой и базой данных. Рассматриваются приемы создания документов системы и работы с ними, приемы создания взаимосвязанных электронных таблиц и работы с ними в режиме hot-line. Подробно описаны средства разработки прикладных систем пользователя. Отдельная глава посвящена графическому интерфейсу системы, технике работы с мышью и клавиатурой.

Пособие полезно при разработке прикладных систем, например, автоматизированных рабочих мест для объемно-календарного планирования производства, бухгалтерии и т. д.

Объем книги 248 стр. Цена 10 руб.

С ДЛЯ PC. ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ С ДЛЯ ПЕРСОНАЛЬНЫХ КОМПЬЮТЕРОВ

А. Григорьев

Выпуск 1. Связь С-программ с операционной системой.
Выпуск 2. Время и звук.
Выпуск 3. Стандартный ввод-вывод и прерывания.
Выпуск 4. Графика на экране I.
Выпуск 5. Графика на экране II.
Выпуск 6. Текст и графика на принтере.
Выпуск 7. Клавиатура.
Выпуск 8. Плоттер и диджитайзер.
Выпуск 9. Мышь.

Серия предназначена для широкого круга пользователей персональных компьютеров

В ней рассмотрены вопросы программирования с помощью современных версий компиляторов Microsoft C и QuickC, связь программ с операционной системой персонального компьютера, особенности доступа к переменным окружения. Работа с аппаратными средствами компьютера и периферийными устройствами. Примеры программ, рекомендации по обнаружению часто встречающихся ошибок и их исправлению, примеры использования стандартных библиотечных функций.

Объем каждого выпуска до 80 стр.

Также в 1992 году планируется выпустить следующие книги:

Программирование в среде MS-QuickBasic
Работа в среде MS-Windows

Вы можете сделать предварительный заказ на все эти издания
по адресу: 115409 Москва, ул. Москворечье, 31, корп. 2. Наши телефоны: 324-83-56, 324-30-55.

ГЛАДКОВ Сергей Алексеевич
КРЕЧКО Юрий Андрееевич
МОЛОДЦОВ Константин Ильич
ПОЛИЩУК Владимир Владимирович
СУЧКОВ Геннадий Александрович

КУРС ПРАКТИЧЕСКОЙ РАБОТЫ С СИСТЕМОЙ АВТОКАД 10

Подписано в печать 11.09.91. Печ. л. 18. Уч.-изд. л. 13,1.
Доп. тираж 4000 экз. Заказ 1856.

Акционерное общество "ДИАЛОГ-МИФИ"
115409 Москва, ул. Москворечье, 31, корп. 2

Типография Московского инженерно-физического института
115409 Москва, Каширское ш., 31

КУРСЫ УСКОРЕННОЙ ПОДГОТОВКИ К ПРАКТИЧЕСКОЙ РАБОТЕ НА ПЕРСОНАЛЬНЫХ КОМПЬЮТЕРАХ IBM PC

На курсы ускоренной подготовки к практической работе на персональных компьютерах семейства IBM PC в авторизованном Учебном Центре фирм Borland, Symantec, Autodesk акционерного общества "ДИАЛОГ-МИФИ" принимаются как граждане СССР, так и иностранные граждане без ограничений.

Занятия проводятся в течение одной-двух недель с отрывом от производства. Оплата по наличному и безналичному расчету. Зачисление на курсы слушателей, направляемых организацией, проводится по гарантийному письму. Иногородние обеспечиваются жильем. Каждому слушателю выдается учебное пособие по изучаемому курсу. По курсам, которые имеют авторизацию указанных фирм выдаются сертификаты от имени этих фирм.

ОБУЧЕНИЕ ПРОВОДИТСЯ ПО СЛЕДУЮЩИМ КУРСАМ:

- Практическая работа на компьютерах семейства IBM PC в операционной среде MS-DOS 4.01
- Основы программирования в системе Microsoft Pascal
- Основы программирования в системе Quick Pascal
- Основы программирования в системе Turbo Pascal
- Основы программирования в системе Microsoft QuickC
- Основы программирования в системе Turbo C
- Основы программирования в системе Turbo C++
- Практическая работа с пакетом Paradox
- Основы программирования в системе Microsoft QuickBASIC
- Диалоговая система инженерных и научных расчетов PC-MATLAB
- Практическая работа в многофункциональной среде Microsoft Windows
- Практическая работа с интегрированным пакетом Microsoft Works
- Использование электронных таблиц с деловой графикой и базами данных в системе Microsoft Excel
- Практическая работа с текстовым процессором Microsoft Word 5.0
- Использование универсальной системы автоматизированного проектирования Автокад 10
- Практическая работа с пакетом PCAD
- Локальные и распределенные вычислительные сети
- Использование информационных технологий в медицине

ДИАЛОГ-МИФИ
СССР, 115409 Москва, ул. Москворечье, 31-2
Телефон: (095) 3203088, (095) 3248356
Телефакс: (095) 3243055