МИНИСТЕРСТВО ВЫСШЕГО И СРЕДНЕГО СПЕЦИАЛЬНОГО ОБРАЗОВАНИЯ РЕСПУБЛИКИ УЗБЕКИСТАН

ТАШКЕНТСКИЙ АРХИТЕКТУРНО-СТРОИТЕЛЬНЫЙ ИНСТИТУТ

У.А.Шадманова, М.А.Заидова, Ф.С.Исламова

ИНФОРМАТИКА И ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

Учебное пособие

ЧАСТЬ-I

ТАШКЕНТ - 2015

УДК: 681.518

Авторы: У.А.Шадманова, М.А.Заидова, Ф.С.Исламова.

Учебное пособие «Информатика и информационные технологии», часть-I.

В данном учебном пособие рассмотрены основные категории аппаратных и программных средств вычислительной техники. Указаны базовые принципы построения архитектур вычислительных систем. Приведены основы работы с распространенными программными продуктами. Рассмотрены основные средства, приемы и методы программирования.

Книга предназначена для студентов технических вузов, изучающих информационные технологии в рамках дисциплины «Информатика», для преподавательского состава, для слушателей учреждений системы повышения квалификации и для лиц изучающих средства вычислительной техники самостоятельно.

Этим учебным пособием, также могут пользоваться специалисты данной отрасли и слушатели курсов повышения квалификации и переподготовки кадров.

Рецензенты:

Аманов А.К.- к.ф-м.н., доцент, зав. кафедрой «Математика и информатика», Акад. лицей при ТАСИ.

Назиров Ш.А.- зав.каф. «Информатики» ТУИТ.

Рекомендовано Научно-методическим Советом Ташкентского архитектурно-строительного института под № 1 от 8 сентября 2015 года.

ВВЕДЕНИЕ

Значение информатики и информационных технологий в жизни общества приобретают все большее значение. Поэтому использование информационных технологий становится одним из основных приоритетов государственной политики Узбекистана. Во исполнение Указа Президента страны И.А. Каримова «О дальнейшем развитии компьютеризации и внедрении информационно-коммуникационных технологий» от 30 мая 2002 года принимаются действенные меры по развитию информационных технологий. С этой целью разработана и реализуется «Программа развития компьютеризации и информационно-коммуникационных технологий на 2002-2010 годы», утвержденная постановлением Кабинетом Министров Республики Узбекистан №200 от 6 июня 2002 года. А также принят 11 февраля 2004 года Закон Республики Узбекистан «Об информатизации». Приоритетность применения информационных технологий для развития как основной фактор прогресса во всех секторах страны, признана и в совместном протоколе сотрудничества Программы развития ООН (ПРООН) и Правительства Республики Узбекистан на период 2000-2004 годы. В целях дальнейшего внедрения и развития информационно-коммуникационных технологий в республике 21 марта 2012 года Президентом Республики Узбекистан И.А.Каримовым принято Постановление «О мерах по дальнейшему внедрению и развитию современных информационно-коммуникационных технологий». Поэтому в последние годы возрастает потребность в квалифицированных специалистах в области информационных технологий и требования к уровню их подготовки.

Такой специалист должен обладать теоретическими знаниями и практическими навыками по использованию современных информационных технологий, включающих технические и программные средства обработки информации, средства телекоммуникаций, а также

знаниями в области алгоритмизации вычислительных процессов для решения задач и создания программ на языке программирования. Дисциплина «Информатика» предусматривает рассмотрение современных операционных систем, программ обработки текстовой и табличной информации, использование компьютерной графики, а также работу в локальной сети и в сети Internet.

Процесс изучения дисциплины «Информатика» предусматривает использование современных информационных технологий. Так, обязательным является использование новейших версий операционной системы WINDOWS, текстовых редакторов, графических редакторов, электронных таблиц, языков программирования. Кроме того, изучение дисциплины предполагает работу, как в локальной сети, так и в сети Интернет.

Кроме использования новых информационных технологий, при изучении дисциплины широко используются и новые педагогические технологии обучения. Это решение ситуационных задач, работа в группах, моделирование процессов, подготовка презентаций, исследование проблем и пр.

Данное учебное пособие охватывает материал курса «Информатика и информационные технологии», для направления бакалавриата всех вузов.

Основная цель обучения дисциплины заключается в том, чтобы дать студентам фундаментальные теоретические знания по работе с техническими и программными средствами вычислительной техники.

Основные задачи обучения дисциплины заключаются в следующем:

- ознакомить студентов с возможностями существующих технических и программных средств систем обработки информации и тенденций их развития;

- дать основные сведения о современных операционных системах;

- ознакомить с пользовательским интерфейсом WINDOWS;

- раскрыть принципы основ алгоритмизации вычислительных процессов.

Коренное отличие информатики и информационных технологий от других технических дисциплин, изучаемых в высшей школе, состоит в том, что ее предмет изучения меняется ускоренными темпами. Сегодня количество компьютеров в мире превышает 500 миллионов единиц, при этом каждая вычислительная система по-своему уникальна. Найти две системы с одинаковыми аппаратными и программными конфигурациями весьма сложно, и потому для эффективной эксплуатации вычислительной техники от специалистов требуется достаточно широкий уровень знаний и практических навыков.

Вместе с тем, в количественном отношении темп численного роста вычислительных систем заметно превышает темп подготовки специалистов, способных эффективно работать с ними. При этом в среднем один раз в полтора года удваиваются основные технические параметры аппаратных средств, один раз в два-три года меняются поколения программного обеспечения и один раз в пять-семь лет меняется база стандартов, интерфейсов и протоколов.

Таким образом, кардинальным отличием информатики и информационных технологий от других технических дисциплин является тот факт, что ее предметная область изменяется чрезвычайно динамично.

Основная задача информатики состоит в преодолении общечеловеческого кризисного явления, называемого «информационным бумом», путем внедрения средств и методов, автоматизирующих операции данными. Однако даже в собственной предметной области информатики испытывается такой информационный бум, какого не знает ни одна другая область человеческой деятельности.

Анализируя вышеуказанные особенности информатики, авторы данного пособия приходят к выводу, что для преподавания информатики и информационных технологий в сложившихся условиях необходимо

расширенное взаимодействие между учебными программами общетехнических, специальных дисциплин и учебной программой курса информатики и информационных технологий. Основные принципы, вытекающие из такого подхода, включают непрерывность и системность образования, а также раннюю профессиональную ориентацию.

Информатика – одна из немногих общетехнических дисциплин, развивающая такие практические навыки, которые востребуются напрямую и немедленно, сразу после включения молодого специалиста в профессиональную деятельность.

Учебное пособие состоит из двух частей. В первой части содержатся сведения о современном состоянии аппаратных и программных средств вычислительной техники, основы алгоритмизации и программы компьютерной графики.

Во второй части содержатся программы Microsoft Office и компьютерных сетей. Каждая тема завершается списком контрольных вопросов, которые могут обсуждаться на лекционных и практических занятиях.

Исходя из структуры и содержания книги, авторы рассчитывают на то, что она будет полезна следующим категориям читателей:

- студентам технических специальностей вузов, изучающим информатику как самостоятельную дисциплину;
- преподавательскому составу, осуществляющему теоретическую и практическую подготовку студентов по дисциплине «Информатика и информационные технологии»;
- преподавателям иных дисциплин, использующим персональные компьютеры в качестве технического средства обучения и средства подготовки учебно-методических материалов (бумажных и электронных) по своей предметной области.

ГЛАВА I. ОБЩИЕ СВЕДЕНИЯ ОБ «ИНФОРМАТИКЕ И ИНФОРМАЦИОННОЙ ТЕХНОЛОГИИ»

В настоящее время информатика охватывает все сферы нашей жизни и развивается невиданными в истории темпами. В этом причина нашего повышенного интереса к информатике.

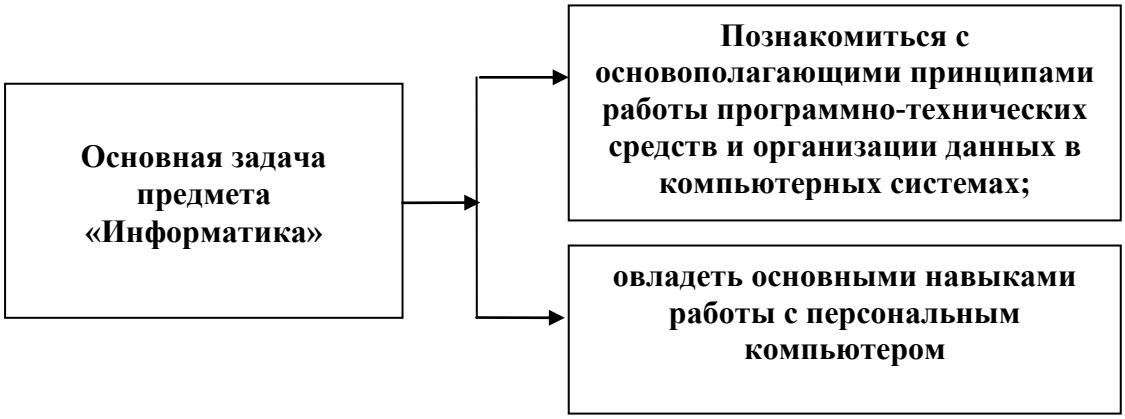
Термин *"информатика"* (франц. *informatique*) происходит от французских слов *information* (информация) и *automatique* (автоматика)
Широко распространён также англоязычный вариант этого термина — *"Computer science"*, что означает буквально *"компьютерная наука"*

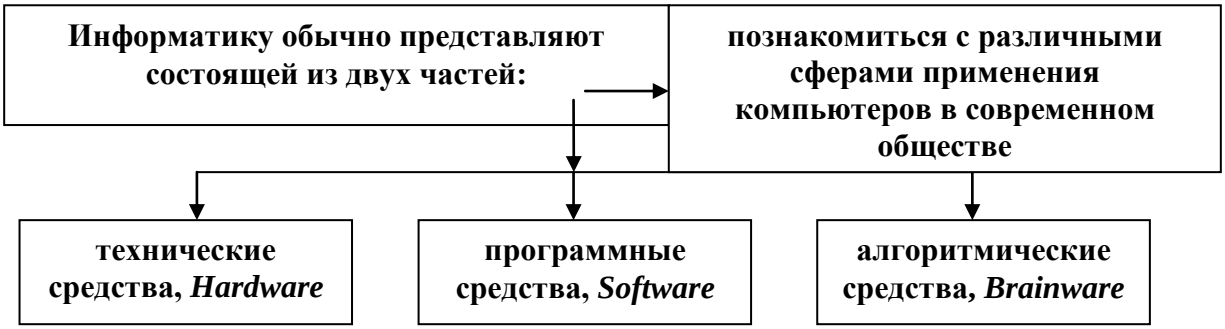
<u>Информатика</u> — это основанная на использовании компьютерной техники дисциплина, изучающая структуру и общие свойства информации, а также закономерности и методы её создания, хранения, поиска, преобразования, передачи и применения в различных сферах человеческой деятельности.

Основные направления Информатики:

- **разработка вычислительных систем и программного обеспечения**;

- **теория информации**, изучающая процессы, связанные с передачей, приёмом, преобразованием и хранением информации;
- **методы искусственного интеллекта**, позволяющие создавать программы для решения задач, требующих определённых интеллектуальных усилий при выполнении их человеком (логический вывод, обучение, понимание речи, визуальное восприятие, игры и др.);
- **системный анализ**, заключающийся в анализе назначения проектируемой системы и в установлении требований, которым она должна отвечать;
- **методы машинной графики, анимации, средства мультимедиа**;
- **средства телекоммуникации;**
- **разнообразные приложения**.





Помимо этих двух общепринятых ветвей информатики выделяют ещё одну существенную ветвь — ***алгоритмические средства***. Для неё российский академик А.А. Дородницин предложил название ***Brainware*** (от англ. *brain* — интеллект). **Эта ветвь связана с разработкой алгоритмов и изучением методов и приёмов их построения**.

Информатизация общества — организованный социально-экономический и научно-технический процесс создания оптимальных условий для удовлетворения информационных потребностей и реализации прав граждан, органов государственной власти, органов местного самоуправления организаций, общественных объединений на основе формирования и использования информационных ресурсов

Цель информатизации — улучшение качества жизни людей за счет увеличения производительности и облегчения условий их труда.

В Законе Республики Узбекистан «Об информатизации» применяются следующие основные понятия:

- «*информатизация* - организационный социально-экономический и научно-технический процесс создания условий для удовлетворения потребностей юридических и физических лиц в информации с использованием информационных ресурсов, информационных технологий и информационных систем;

- *информационный ресурс* - информация, банк данных, база данных в электронной форме в составе информационной системы;

- *информационная технология* - совокупность методов, устройств, способов и процессов, используемых для сбора, хранения, поиска, обработки и распространения информации;

- *информационная система* - организационно упорядоченная совокупность информационных ресурсов, информационных технологий и средств связи, позволяющая осуществлять сбор, хранение, поиск, обработку и пользование информацией».

Технология - от греческого (techno) означает искусство, мастерство, учение, ремесло. А это значит процесс.

Процесс это совокупность действий, который приводит к намеченной цели. Человек сам выбирает стратегию, чтобы осуществить процесс. И это осуществляется совокупностью методов и способов.

Раньше мощь государства зависела от материальной производственной технологии.

Ресурсы → **Технология производства** → Продукт

Задача материальной производственной технологии удовлетворить потребность человека или системы своим производственным продуктом.

Развитие технологии, появление вычислительной техники повысило производительность, и сейчас мощь страны зависит от производства информации.

→ Инф.ресурсы **Информационные технологии** → Инф. продукт

Информационные ресурсы - это идеи человечества и указания по их реализации, накопленные в форме, позволяющей их воспроизводство.

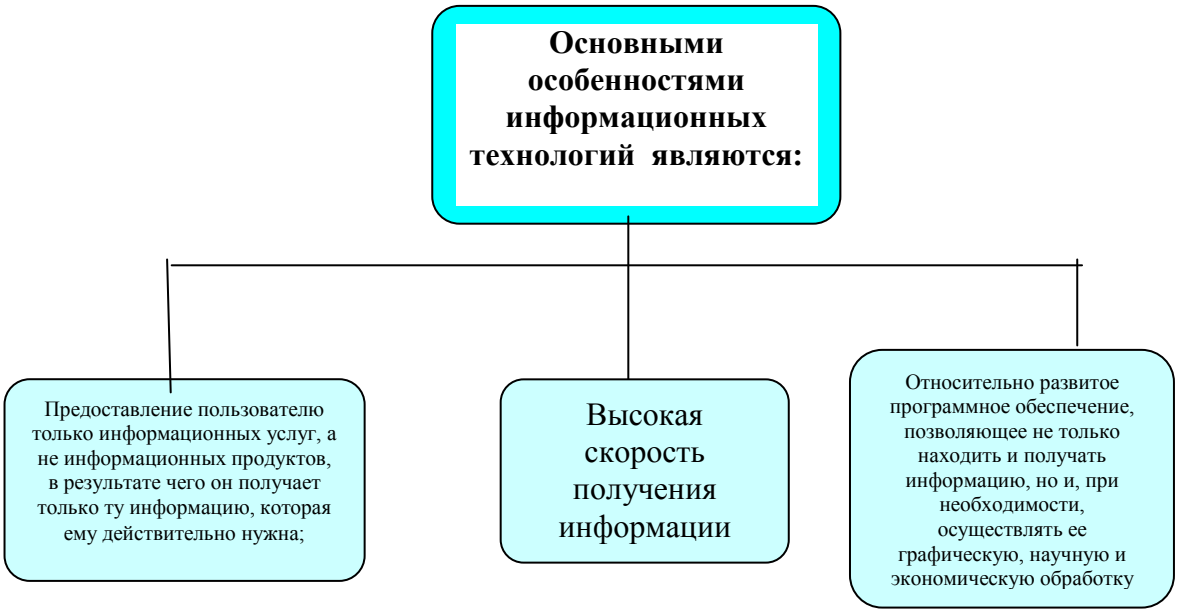
Это книги, статьи, патенты, диссертации, научно-исследовательская и опытно конструкторская документация, технические переводы, данные о передовом производственном опыте и др.

Информационные ресурсы (в отличие от других видов ресурсов - трудовых, энергетических, минеральных и т.д.) тем быстрее растут, чем больше их расходуют.

Информационная технология - это изучение методов создания, методы обработки, методы передачи, методы хранения, методы поиска информации с помощью современных аппаратных средств и программных продуктов.

Основная цель информационной технологии производство такой полезной информации, на основе которой можно будет принимать наиболее обоснованные управленческие или иные решения по выполнению, какого-либо действия (информацию, чтобы человек анализировал и по этому поводу принимал решение).

Информационные технологии предполагают предоставление пользователю не только информационного продукта, но и средств доступа к нему (средства поиска, обработки, представления и т.п.), эти средства позволяют пользователю не только ознакомиться с содержанием компьютерной информации, но и получить информацию (документ) в объеме и формате, которые адекватны именно его потребностям.



Роль информационных технологий в развитии общества состоит в ускорении процессов получения, распространения и использования обществом новых знаний.

Внедрение персонального компьютера в информационную сферу и применение телекоммуникационных средств связи определили новый этап развития информационной технологии.

Новая информационная технология - информационная технология с дружеским интерфейсом работы пользователя, использующая компьютеры и телекоммуникационные средства.

Прилагательное «компьютерная» подчеркивает, что основным техническим средством ее реализации является компьютер.

В понятие новой информационной технологии включены также новые информационные технологии, которые обеспечивают передачу информации разными средствами, а именно – телефон, телеграф, телекоммуникации, факс и др.

Три основных принципа новой (компьютерной) информационной технологии:

- Интерактивный (диалоговый) режим работы с компьютером.
- Интегрированность (стыковка, взаимосвязь) с другими программными продуктами.
- Гибкость процесса изменения, как данных, так и постановок задач.

Составляющие информационной технологии

Состав информационных технологий:

- этапы, где реализуются сравнительно длительные технологические процессы;
- операции, в результате выполнения которых будет создан конкретный объект на определенном уровне программной среды;
- действия - совокупность стандартной для каждой программы среды приемов работы, приводящих к выполнению поставленной в соответствующей операции цели;
- элементарные операции по управлению мышью и клавиатурой.

Этапы развития информационной технологии.

Рассмотрим этапы развития информационной технологии по признаку – виды инструментария технологии.

1й этап (до второй половины XIX века)	**-«ручная» информационная технология**
2й этап (с конца XIX века)	-«механическая» технология
3й этап (60-40е гг. XX века)	-«электрическая» технология
4й этап (с начала 70х гг.)	-«электронная» технология
5й этап (с середины 80х гг.)	-«компьютерная» («новая») технология

1.Информационная технология обработки данных, предназначенных для решения хорошо структурированных задач, по которым имеются необходимые входные данные и известны алгоритмы и другие стандартные процедуры их обработки	Основными компонентами информационной технологии обработки данных являются: • Сбор данных; • Обработка данных; • Хранение данных; • Создание отчетов (документов).
2. Информационная технология используется при худшей структурированности решаемых задач по сравнению с предыдущей	Основными компонентами информационной технологии управления являются: • База данных; • Формирование управленческих отчетов (регулярных или специальных).
3. Автоматизация офиса - это организация и поддержка коммуникационных процессов как внутри организации, так и с внешней средой на базе компьютерных сетей и других современных средств передачи и работы с информацией	Основными компонентами автоматизации офиса являются: • База данных; • Текстовый процессор; • Электронная почта; • Аудиопочта; • Табличный процессор; • Электронный календарь; • Компьютерные конференции; • Телеконференции; • Хранение изображения; • Видеотекст; • Управленческие программы; • Аудиоконференции; • Видеоконференции; • Факсимильная связь.
4.Информационная технология поддержки принятия решений - качественно новый метод организации взаимодействия человека и компьютера	При этом выработка решения происходит в результате итерационного процесса, в котором участвуют: • Система поддержки принятия решений в роли вычислительного звена и объекта управления; • Человек, как управляющее звено, задающее входные данные и оценивающее полученный результат вычислений на компьютере. Окончание итерационного процесса происходит по воле человека. Основными компонентами информационной технологии поддержки принятия решения являются: • База данных; • База моделей; • Система управлений базой данных; • Система управлений базой моделей; • Система управлений интерфейсом.

5.Информационная технология экспертных систем основана на использовании искусственного интеллекта. Экспертные системы позволяют менеджеру и специалисту получать консультации экспертов по любым проблемам, о которых этими системами накоплены знания	Основными компонентами информационной технологии экспертных систем являются: • Интерфейс пользователя; • База знаний; • Интерпретатор; • Модуль создания системы.

Виды информационной технологии.

Существуют следующие виды информационной технологии:

1.1 Представление информации на компьютере

Термин ***"информация"*** происходит от латинского слова "***informatio***", что означает ***сведения, разъяснения, изложение.***

Информация — сведения об объектах и явлениях окружающей среды, их параметрах, свойствах и состоянии, которые воспринимают информационные системы (живые организмы, управляющие машины и др.) в процессе жизнедеятельности и работы.

Применительно к компьютерной обработке данных под информацией понимают некоторую последовательность символических обозначений (букв, цифр, закодированных графических образов и звуков и т.п.), несущую смысловую нагрузку и представленную в понятном компьютеру виде. Каждый новый символ в такой последовательности символов увеличивает информационный объём сообщения.

Информация может существовать в самых разнообразных формах:

• в виде текстов, рисунков, чертежей, фотографий;
• в виде световых или звуковых сигналов;
• в виде радиоволн;
• в виде электрических и нервных импульсов;
• в виде магнитных записей;
• в виде жестов и мимики;
• в виде запахов и вкусовых ощущений;
• в виде хромосом, посредством которых передаются по наследству признаки и свойства организмов и т.д.

Предметы, процессы, явления материального или нематериального свойства, рассматриваемые с точки зрения их информационных свойств, называются информационными объектами.

Передача информации.

Информация передаётся в виде **сообщений** от некоторого **источника** информации к её **приёмнику** посредством **канала связи** между ними. Источник посылает **передаваемое сообщение**, которое **кодируется в передаваемый сигнал**. Этот сигнал посылается по **каналу связи**. В результате в приёмнике появляется **принимаемый сигнал**, который **декодируется** и становится **принимаемым сообщением**.

канал связи

ИСТОЧНИК ——————————□ **ПРИЁМНИК**

Примеры:

1. *Сообщение, содержащее информацию о прогнозе погоды, передаётся приёмнику* (телезрителю) *от источника* — специалиста-метеоролога *посредством канала связи* — телевизионной передающей аппаратуры и телевизора.

2. *Живое существо при помощи своих органов чувств* (глаз, ухо, кожа, язык и т.д.) *воспринимает информацию из внешнего мира*, *перерабатывает её* в определенную последовательность нервных импульсов, *передает* импульсы по нервным волокнам, *хранит* в памяти в виде состояния нейронных структур мозга, *воспроизводит* в виде звуковых сигналов, движений и т.п., *использует* в процессе своей жизнедеятельности.

Передача информации по каналам связи часто сопровождается воздействием **помех**, вызывающих **искажение и потерю информации**.

Измерение информации.

В качестве единицы информации условились принять один бит (*англ. bit — **bi**nary, digi**t*** — двоичная цифра).

Бит в теории информации — количество информации, необходимое для различения двух равновероятных сообщений. А в вычислительной технике битом называют наименьшую "порцию" памяти, необходимую для хранения одного из двух знаков "0" и "1", используемых для внутримашинного представления данных и команд.

Бит — слишком мелкая единица измерения. На практике чаще применяется более крупная единица — ***байт***, равная **восьми битам**. Именно **восемь битов требуется для того, чтобы закодировать любой из 256 символов алфавита клавиатуры компьютера ($256=2^8$)**.

Широко используются также ещё **более крупные производные единицы информации**:

- **1 Килобайт (Кбайт) = 1024 байт = 2^{10} байт.**
- **1 Мегабайт (Мбайт) = 1024 Кбайт = 2^{20} байт.**
- **1 Гигабайт (Гбайт) = 1024 Мбайт = 2^{30} байт.**

В последнее время в связи с увеличением объёмов обрабатываемой информации входят в употребление такие производные единицы, как:

- **1 Терабайт (Тбайт) = 1024 Гбайт = 2^{40} байт.**
- **1 Петабайт (Пбайт) = 1024 Тбайт = 2^{50} байт.**

За единицу информации можно было бы выбрать количество информации, необходимое для различения, например, десяти равновероятных сообщений. Это будет не двоичная (**бит**), а десятичная (**дит**) единица информации.

Информацию можно:

• создавать; • передавать; • воспринимать; • использовать; • запоминать; • принимать; • копировать;	• формализовать; • распространять; • преобразовывать; • комбинировать; • обрабатывать; • делить на части; • упрощать;	• собирать; • хранить; • искать; • измерять; • разрушать; • и др. •

Все эти процессы, связанные с определенными операциями над информацией, называются информационными процессами.

Свойства информации:

• достоверность; • полнота; • ценность; • своевременность;	• понятность; • доступность; • краткость; • и др.

Информация достоверна, если она отражает истинное положение дел. Недостоверная информация может привести к неправильному пониманию или принятию неправильных решений.

Достоверная информация со временем может стать недостоверной, так как она обладает свойством устаревать, то есть перестаёт отражать истинное положение дел.

Информация полна, если её достаточно для понимания и принятия решений. Как неполная, так и избыточная информация сдерживает принятие решений или может повлечь ошибки.

Точность информации определяется степенью ее близости к реальному состоянию объекта, процесса, явления и т.п.

Ценность информации зависит от того, насколько она важна для решения задачи, а также от того, насколько в дальнейшем она найдёт применение в каких-либо видах деятельности человека.

Только своевременно полученная информация может принести ожидаемую пользу. Одинаково нежелательны, как преждевременная подача информации (когда она ещё не может быть усвоена), так и её задержка.

Если ценная и своевременная информация выражена непонятным образом, она может стать бесполезной.

Информация становится понятной, если она выражена языком, на котором говорят те, кому предназначена эта информация.

Информация должна преподноситься в доступной (по уровню восприятия) форме. Поэтому одни и те же вопросы по-разному излагаются в школьных учебниках и научных изданиях.

Информацию по одному и тому же вопросу можно изложить кратко (сжато, без несущественных деталей) или пространно (подробно, многословно). Краткость информации необходима в справочниках, энциклопедиях, учебниках, всевозможных инструкциях.

1.2. Обработка информации

Обработка информации – получение одних информационных объектов из других информационных объектов путем выполнения некоторых алгоритмов.

Средства обработки информации — это всевозможные устройства и системы, созданные человечеством, и в первую очередь, компьютер — универсальная машина для обработки информации.

Компьютеры обрабатывают информацию путем выполнения некоторых алгоритмов.

Живые организмы и растения обрабатывают информацию с помощью своих органов и систем.

Вопросы для закрепления знаний:

1. Дать понятие предмету информатика.
2. Основная задача предмета информатики.
3. Развитие информатики.
4. Составные части информатики.
5. Цель информатизации.
6. Как следует понимать новую информационную технологию?
7. Что такое инструментарий информационной технологии?
8. Какова история развития информационной технологии?
9. Перечислите основные компоненты информационных технологий поддержки принятия решения.
10. Перечислите основные компоненты информационных технологий экспертных систем.

11. Что такое информация?
12. В каком виде существует информация?
13. Как передаётся информация?
14. Как измеряется количество информации?
15. Что можно делать с информацией?
16. Какими свойствами обладает информация?
17. Что такое обработка информации?

ГЛАВА II. КОМПЬЮТЕРЫ

2.1.Классификация компьютеров

Компьютер **(англ. *computer* — вычислитель) представляет собой программируемое электронное устройство, способное обрабатывать данные и производить вычисления, а также выполнять другие задачи манипулирования символами.**

Существует два основных класса компьютеров:

- ***цифровые компьютеры***, обрабатывающие данные в виде числовых двоичных кодов;
- ***аналоговые компьютеры***, обрабатывающие непрерывно меняющиеся физические величины (электрическое напряжение, время и т.д.), которые являются аналогами вычисляемых величин.

Любая компьютерная программа представляет собой последовательность отдельных команд.

Команда — это описание операции, которую должен выполнить компьютер. Как правило, у команды есть свой *код* (условное обозначение), *исходные данные* (операнды) и *результат*.

Совокупность команд, выполняемых данным компьютером, называется *системой команд* этого компьютера.

Как устроен компьютер?

Разнообразие современных компьютеров очень велико. Но их структуры основаны на **общих логических принципах**, позволяющих выделить в любом компьютере следующие **главные устройства**:

- **память** (запоминающее устройство, ЗУ), состоящую из перенумерованных ячеек;
- **процессор**, включающий в себя **устройство управления** (УУ) и **арифметико-логическое устройство** (АЛУ);
- **устройство ввода**;

- **устройство вывода**.

Эти устройства соединены ***каналами связи***, по которым передается информация.

Основные устройства компьютера и связи между ними представлены на схеме (рис.2.1.1). Жирными стрелками показаны пути и направления движения информации, а простыми стрелками — пути и направления передачи управляющих сигналов.

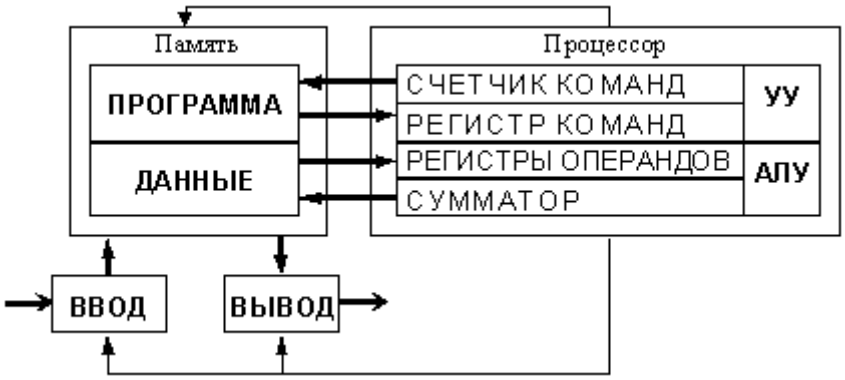


Рис.2.1.1. Общая схема компьютера.

Функции памяти:

- ***приём информации*** из других устройств;
- ***запоминание информации***;
- ***выдача информации*** по запросу в другие устройства машины.

Функции процессора:

- ***обработка данных по заданной программе*** путем выполнения арифметических и логических операций;
- ***программное управление работой устройств*** компьютера.

Та часть процессора, которая выполняет команды, называется *арифметико-логическим устройством* (АЛУ), а другая его часть, выполняющая функции управления устройствами, называется *устройством управления* (УУ).

Обычно эти два устройства выделяются чисто условно, ***конструктивно они не разделены***.

В составе процессора имеется ряд специализированных дополнительных ячеек памяти, называемых ***регистрами*.**

Регистр выполняет функцию кратковременного хранения числа или команды. Над содержимым некоторых регистров специальные электронные схемы могут выполнять некоторые манипуляции. Например, "вырезать" отдельные части команды для последующего их использования или выполнять определенные арифметические операции над числами.

Основным элементом регистра является электронная схема, называемая ***триггером*,** которая способна хранить одну двоичную цифру (*разряд*).

Регистр представляет собой совокупность триггеров, связанных друг с другом определённым образом общей системой управления.

Существует несколько типов регистров, отличающихся видом выполняемых операций.

Некоторые важные регистры имеют свои названия, например:

- **сумматор** — регистр АЛУ, участвующий в выполнении каждой операции (принцип его работы рассмотрен в разделе 5.8);
- **счетчик команд** — регистр УУ, содержимое которого соответствует адресу очередной выполняемой команды; служит для автоматической выборки программы из последовательных ячеек памяти;
- **регистр команд** — регистр УУ для хранения кода команды на период времени, необходимый для ее выполнения. Часть его разрядов используется для хранения ***кода операции***, остальные — для хранения ***кодов адресов операндов***.

На каких принципах построены компьютеры?

В основу построения подавляющего большинства компьютеров положены следующие общие принципы, сформулированные в 1945 г. американским ученым **Джоном фон Нейманом**.

1. Принцип программного управления. Из него следует, что программа состоит из набора команд, которые выполняются процессором автоматически друг за другом в определенной последовательности.

2. Принцип однородности памяти. Программы и данные хранятся в одной и той же памяти. Поэтому компьютер не различает, что хранится в данной ячейке памяти — число, текст или команда. Над командами можно выполнять такие же действия, как и над данными.

3. Принцип адресности. Структурно основная память состоит из перенумерованных ячеек; процессору в произвольный момент времени доступна любая ячейка.

Рис. 2.1.2. Джон фон Нейман, 1945 г.

Компьютеры, построенные на этих принципах, относятся к типу фон-неймановских. Но существуют компьютеры, принципиально отличающиеся от фон-неймановских. Для них, например, может *не выполняться принцип программного управления*, т.е. они могут работать без “счетчика команд”, указывающего текущую выполняемую команду программы. Для обращения к какой-либо переменной, хранящейся в памяти, этим компьютерам *не обязательно давать ей имя*. Такие компьютеры называются не-фон-неймановскими.

По каким критериям классифицируют компьютеры?

Существуют различные классификации компьютерной техники:
по этапам развития (по поколениям);
по архитектуре;
по производительности;
по условиям эксплуатации;
по количеству процессоров;
по потребительским свойствам и т.д.

Четких границ между классами компьютеров не существует. По мере совершенствования структур и технологии производства, появляются новые классы компьютеров, границы существующих классов существенно изменяются.

На чем основана классификация по поколениям?

Деление компьютерной техники на поколения — весьма условная, нестрогая классификация вычислительных систем по степени развития аппаратных и программных средств, а также способов общения с компьютером.

Идея делить машины на поколения вызвана к жизни тем, что за время короткой истории своего развития компьютерная техника проделала большую эволюцию, как в смысле **элементной базы** (*лампы, транзисторы, микросхемы* и др.), так и в смысле **изменения её структуры, появления новых возможностей, расширения областей применения и характера использования.**

Какие компьютеры относятся в первому поколению?

К **первому поколению** обычно относят машины, созданные на рубеже 50-х годов. В их схемах использовались **электронные лампы**. Эти компьютеры были **огромными, неудобными и слишком дорогими машинами**. Лампы потребляли огромное количество электроэнергии и выделяли много тепла.

Быстродействие порядка 10-20 тысяч операций в секунду.

Программы для этих машин писались **на языке конкретной машины**.

Отечественные машины первого поколения: МЭСМ (малая электронная счётная машина), БЭСМ, Стрела, Урал, М-20.

Какие компьютеры относятся ко второму поколению?

Второе поколение компьютерной техники — машины, сконструированные примерно в 1955-65 гг. Характеризуются

использованием в них, как **электронных ламп**, так и **дискретных транзисторных логических элементов**.

Их оперативная память была построена на магнитных сердечниках. В это время стал расширяться диапазон применяемого оборудования ввода-вывода, появились высокопроизводительные **устройства для работы с магнитными лентами, магнитные барабаны и первые магнитные диски**.

Быстродействие — до сотен тысяч операций в секунду, **ёмкость памяти** — до нескольких десятков тысяч слов.

Появились так называемые **языки высокого уровня**, средства которых допускают описание всей необходимой последовательности вычислительных действий **в наглядном, легко воспринимаемом виде**.

Программа, написанная на алгоритмическом языке, непонятна компьютеру, воспринимающему только язык своих собственных команд. Поэтому специальные программы, которые называются **трансляторами**, переводят программу с языка высокого уровня на машинный язык.

Появился широкий набор библиотечных программ для решения разнообразных математических задач. Появились **мониторные системы**, управляющие режимом трансляции и исполнения программ. Из мониторных систем в дальнейшем выросли современные операционные системы.

Операционная система **— важнейшая часть программного обеспечения компьютера, предназначенная для автоматизации планирования и организации процесса обработки программ, ввода-вывода и управления данными, распределения ресурсов, подготовки и отладки программ, других вспомогательных операций обслуживания.**

Таким образом, ***операционная система*** ***является программным расширением устройства управления компьютера***.

В чем особенности компьютеров третьего поколения?

Машины третьего поколения созданы примерно после 60-х годов. Возможно, наиболее важным критерием различия машин второго и третьего поколений является критерий, основанный на понятии архитектуры.

Машины третьего поколения — это семейства машин с единой архитектурой, т.е. программно совместимых. В качестве элементной базы в них используются интегральные схемы, которые также называются микросхемами.

Машины третьего поколения имеют развитые операционные системы. Примеры машин третьего поколения — семейства IBM-360, IBM-370, ЕС ЭВМ (Единая система ЭВМ), СМ ЭВМ (Семейство малых ЭВМ) и др.

Быстродействие машин внутри семейства изменяется от нескольких десятков тысяч до миллионов операций в секунду. Ёмкость оперативной памяти достигает нескольких сотен тысяч слов.

Что характерно для машин четвёртого поколения?

Четвёртое поколение — это теперешнее поколение компьютерной техники, разработанное после 1970 года.

Наиболее важный в концептуальном отношении критерий, это эффективное использование современных высокоуровневых языков и упрощение процесса программирования для конечного пользователя.

В аппаратурном отношении для них характерно широкое использование ***интегральных схем*** в качестве элементной базы, а также наличие быстродействующих запоминающих устройств с произвольной выборкой ёмкостью в десятки мегабайт.

С точки зрения структуры машины этого поколения представляют собой ***многопроцессорные*** и ***многомашинные комплексы***, работающие на общую память и общее поле внешних устройств. Быстродействие составляет до нескольких десятков миллионов операций в секунду, ёмкость оперативной памяти порядка 1 - 64 Мбайт.

Для них характерны:

- применение персональных компьютеров;
- телекоммуникационная обработка данных;
- компьютерные сети;
- широкое применение систем управления базами данных;
- элементы интеллектуального поведения систем обработки данных и устройств.

Какими должны быть компьютеры пятого поколения?

Разработка последующих поколений компьютеров производится на основе **больших интегральных схем повышенной степени интеграции**, использования оптоэлектронных принципов (**лазеры**, **голография**).

Развитие идет также по пути *"интеллектуализации"* компьютеров, устранения барьера между человеком и компьютером. Компьютеры будут способны воспринимать информацию с рукописного или печатного текста, с бланков, с человеческого голоса, узнавать пользователя по голосу, осуществлять перевод с одного языка на другой.

В компьютерах пятого поколения произойдёт качественный переход от обработки *данных* к обработке *знаний*.

Архитектура компьютеров будущего поколения будет содержать два основных блока. Один из них — это **традиционный** компьютер. Но теперь он лишён связи с пользователем. Эту связь осуществляет блок, называемый термином **"интеллектуальный интерфейс"**. Его задача — понять текст, написанный на естественном языке и содержащий условие задачи, и перевести его в работающую программу для компьютера.

На какие типы делятся компьютеры по условиям эксплуатации?

По условиям эксплуатации компьютеры делятся на два типа:

- офисные (универсальные);
- специальные.

Офисные предназначены для решения широкого класса задач при нормальных условиях эксплуатации.

Специальные компьютеры служат для решения более узкого класса задач или даже одной задачи, требующей многократного решения, и функционируют в особых условиях эксплуатации.

На какие типы делятся компьютеры по производительности и характеру использования?

По производительности и характеру использования компьютеры можно условно подразделить на:

- микрокомпьютеры, в том числе — персональные компьютеры;
- миникомпьютеры;
- мэйнфреймы (универсальные компьютеры);
- суперкомпьютеры.

Микрокомпьютеры — это компьютеры, в которых центральный процессор выполнен в виде микропроцессора.

Микрокомпьютеры представляют собой инструменты для решения разнообразных сложных задач. Их микропроцессоры с каждым годом увеличивают мощность, а периферийные устройства — эффективность. Быстродействие — порядка 1 - 10 миллионов операций в сек.

Персональные компьютеры (ПК) — это микрокомпьютеры универсального назначения, рассчитанные на одного пользователя и управляемые одним человеком.

Миникомпьютерами и **суперминикомпьютерами** называются машины, конструктивно выполненные в одной стойке, т.е. занимающие объём порядка половины кубометра. Сейчас компьютеры этого класса вымирают, уступая место микрокомпьютерам.

Мэйнфреймы предназначены для решения широкого класса научно-технических задач и являются сложными и дорогими машинами. Их целесообразно применять в больших системах при наличии не менее 200 - 300 рабочих мест.

Суперкомпьютеры — это очень мощные компьютеры с производительностью свыше 100 мегафлопов (1 мегафлоп — миллион

операций с плавающей точкой в секунду). Они называются **сверхбыстродействующими**. Эти машины представляют собой **многопроцессорные** и (или) **многомашинные** комплексы, работающие на общую память и общее поле внешних устройств. Различают суперкомпьютеры *среднего класса*, класса *выше среднего* и *переднего края* (*high end*).

Архитектура суперкомпьютеров основана на идеях ***параллелизма*** и ***конвейеризации вычислений***.

Суперкомпьютеры используются для решения сложных и больших научных задач (метеорология, гидродинамика и т. п.), в управлении, разведке, в качестве централизованных хранилищ информации и т.д.

Элементная база — микросхемы сверхвысокой степени интеграции.

Какие существуют типы портативных компьютеров?

Laptop (наколенник, от ***lap*** — колено и ***top*** — поверх). По размерам близок к обычному портфелю. По основным характеристикам (быстродействие, память) примерно соответствует настольным ПК. Сейчас компьютеры этого типа уступают место ещё меньшим.

Notebook (блокнот, записная книжка). По размерам он ближе к книге крупного формата. Имеет вес около 3 кг. Помещается в портфель-дипломат. Для связи с офисом его обычно комплектуют *модемом*. Ноутбуки зачастую снабжают *приводами CD-ROM*.

Palmtop (наладонник) — самые маленькие современные персональные компьютеры. Умещаются на ладони. Магнитные диски в них заменяет энергонезависимая электронная память. Нет и накопителей на дисках — обмен информацией с обычными компьютерами идет линиям связи. Если Palmtop дополнить набором деловых программ, записанных в его постоянную память, получится ***персональный цифровой помощник*** (*Personal Digital Assistant*).

Основные устройства компьютера следующее:

♦ Монитор (дисплей, экран)- служит для изображения текстовой и графической информации;

♦ Клавиатура – позволяет вводить символы в компьютер;

♦ Системный блок, располагаются все основные узлы компьютера.

Дополнительные устройства компьютера: принтеры, манипуляторы, сканер, ксерокопильный аппарат, колонки, модем, звуковая карта и т.д.

2.2.Составные части компьютеров, их функции и назначения

Построение персональных компьютеров. Основные устройства компьютера следующее:

♦ Монитор (дисплей, экран)- служит для изображения текстовой и графической информации;

♦ Клавиатура – позволяет вводить символы в компьютер;

♦ Системный блок, располагаются все основные узлы компьютера.

поговорим о мониторе

Мониторы

монитор = дисплей = экран = "очко" = "фонарь"

Электронно-лучевые
- с теневой маской
- с щелевой маской
- с апертурной решеткой

Жидкокристаллические (LCD display)

Газо-плазменные (plasma display)

Сенсорные экраны (sensor display)

:пективные display (FED-, LEP-, LED-, ЭЛ-,

:ернила, нанотрубки, плоскопанельные, DLP-

Вращающиеся проекторы

Монитор - устройство визуального отображения информации (в виде текста, таблиц, рисунков, чертежей и др.). Разрешение (resolution, разрешающая способность монитора) - это разрешающая способность монитора и частота его кадровой развертки. Разрешение - это количество точек по горизонтали и по вертикали на экране монитора. Чем выше разрешение, тем более детальным может быть изображение на экране. Минимальный элемент изображения на экране (точка) называется пикселем- от английского «picture element»:

- EGA 640 x 350 выводит прямоугольные пиксели.
- VGA 640 x 480 , SVGA квадратные пиксели.

Монитор работает под управлением специального аппаратного устройства – видеоадаптера, который предусматривает два возможных режима – текстовый и графический. Мониторы бывают цветные и монохромные.

Системный блок.

В системном блоке располагаются все основные узлы компьютера.

1. *Микропроцессор* – «мозг» машины, который выполняет поступающие на его вход команды, а именно: проводит вычисления и дирижирует работой остальных элементов компьютера. В компьютерах используется микропроцессоры фирмы Intel, AMD, Cyrix, IBM и др. Микропроцессоры фирмы Intel, таковы Intel – 8088, 80286,80386,80486, Pentium, Pentium Pro.

Одинаковые модели микропроцессоров могут иметь разную тактовую частоту – чем выше тактовая частота, тем выше производительность и цена микропроцессора. Тактовая частота измеряется в мегагерцах (мГц). Например, микропроцессоры Pentium выпускаются с тактовой частотой от 75 до 200 мГц. Тактовая частота указывается вслед за моделью микропроцессора (Pentium/75мГц).

Тактовая частота указывает скорость выполнения элементарных операций внутри микропроцессора.

Сопроцессор используется в тех случаях, когда на компьютере приходится выполнять много математических вычислений, над вещественными числами, 80486 DX, Pentium/133мГц, Pentium Pro/150мГц микропроцессорах существует сопроцессор встроенный.

2. *Память* служит для хранения информации и программы. Память делится на след. виды:

- постоянная;
- оперативная;
- внешняя;
- сверхоперативная.

Оперативная память, предназначенная для временного хранения программ и данных, а также для постоянного хранения встроенного блока операционной системы. Обычно называют RAM (random access memory, то есть память с произвольным доступом).

Сверхоперативная память (КЭШ память) располагается между микропроцессором и оперативной памятью и хранит копии наиболее часто используемых участков оперативной памяти.

Постоянная память. BIOS (Basic Input-Output System-базовая система ввода-вывода) в которую данные занесены при ее изготовлении. Как правило, эти данные не могут быть изменены, выполняемые на компьютере программы могут только их считывать. Такой вид памяти обычно называют ROM (read only memory, или память только для чтения), или ПЗУ (постоянное запоминающее устройство)

Контроллеры это электронные схемы, управляющие различными устройствами компьютера.

Шины – магистрали передачи данных между оперативной памятью и контроллерами (шина ISA, PCI).

Расширяя функциональные возможности компьютера к системному блоку можно подключать следующие дополнительные устройства:

- Манипуляторы – устройство облегчающие ввод информации в компьютер.

- Манипуляторы (мышь, джойстик, трекбол и др.) — это специальные устройства, которые используются для управления курсором.

Принтер **— печатающее устройство. Осуществляет вывод из компьютера закодированной информации в виде печатных копий текста или графики.**

Существуют тысячи наименований принтеров. Но основных видов принтеров три: матричные, лазерные и струйные.

Плоттер **(графопостроитель) — устройство, которое чертит графики, рисунки или диаграммы под управлением компьютера.**

Плоттеры используются для получения сложных конструкторских чертежей, архитектурных планов, географических и метеорологических карт, деловых схем. **Плоттеры рисуют изображения с помощью пера. Роликовые плоттеры** прокручивают бумагу под пером, а **планшетные плоттеры** перемещают перо через всю поверхность горизонтально лежащей бумаги.

Сканер — устройство для ввода в компьютер графических изображений. Создает оцифрованное изображение документа и помещает его в память компьютера.

Если принтеры выводят информацию из компьютера, то сканеры, наоборот, **переносят информацию с бумажных документов в память компьютера.** Существуют **ручные сканеры**, которые прокатывают по поверхности документа рукой, и **планшетные сканеры**, по внешнему виду напоминающие копировальные машины.

Модем — устройство для передачи компьютерных данных на большие расстояния по телефонным линиям связи.

Модем обеспечивает преобразование цифровых сигналов компьютера в переменный ток частоты звукового диапазона — этот процесс называется ***модуляцией***, **а также обратное преобразование,** которое называется ***демодуляцией***. Отсюда название устройства: ***модем*** — ***мо****дулятор/****дем****одулятор*. Модемы бывают ***внешние***, выполненные в виде отдельного устройства, и ***внутренние,*** представляющие собой электронную плату, устанавливаемую внутри компьютера. Почти все модемы поддерживают и функции факсов. ***Факс*** **— это устройство факсимильной передачи изображения по телефонной сети**. Название *"факс"* произошло от слова *"факсимиле"* (лат. *fac simile* — сделай подобное), означающее точное воспроизведение графического оригинала (подписи, документа и т.д.) средствами печати. Модем, который может передавать и получать данные как факс, называется *факс-модемом*:

- стример – для хранения данных на магнитной ленте;

- звуковая карта – для воспроизведения и записи звуков;

-магнитооптические съемные диски – применяются для резервирования данных и для хранения редко используемых данных флэшки – для хранения данных.

Стример (англ. tape streamer) — устройство для резервного копирования больших объёмов информации. В качестве носителя здесь применяются кассеты с магнитной лентой ёмкостью 1 - 2 Гбайта и больше.

Клавиатура служит для ввода информации на компьютер. Чтобы быстро освоить кнопки на клавиатуре легче его разделить на 4 части:

1.Функциональные клавиши F1 – F12.

2.Основная (алфавитно-цифровая) клавиша.

3.Служебные клавиши.

4.Вспомогательные клавиши. Некоторые из клавиш:

Сигнал о завершении ввода. Нажатие кнопки ОК.

Завершение программы. Сигнал на отмену операции.

Home — Перевод курсора к началу строки.

End — Перевод курсора к концу строки.

PgUp — Перемещение на «страницу» вверх.

PgDn — Перемещение на «страницу» вниз.

Delete — Удаление символа, под которым находится курсор.

Insert — Переключение режима вставки и замещение при вводе информации в текстовых редакторах.

Back Space — Удаление символа слева от курсора.

Print Screen — Пересылка графической копии экрана.

Break — Служит для остановки программы в буфер обмена

Tab — При редактировании текстов обычно используется для перехода к следующей позиции табуляции. Переключение между полями.

Ctrl, Alt, Shift — Предназначены для изменения назначений других клавиш.

Scroll Lock — Режим блокировки прокрутки.

Num Lock — Режим блокировки цифр.

Caps Lock — Режим прописных букв.

Shift – Alt
Shift – Ctrl
} С помощью комбинации клавиш можно изменить алфавит.

Ctrl – Alt – Del — Завершает задачу и вызывает “Диспетчер задач”.

Вопросы для закрепления знаний:

1. Что такое компьютер?
2. Какие классы компьютеров существуют?
3. Дать определение следующим понятиям: команда, регистр, триггер.
4. Как устроен компьютер?
5. Функции памяти, какие памяти бывают на компьютерах?
6. Функции процессора. Из каких устройств состоит процессор?
7. На каких принципах построены компьютеры? Фон-неймановские компьютеры.
8. По каким критериям классифицируют компьютеры?
9. В чём основана классификация по поколениям? Поколения компьютеров.
10. Начало эволюции компьютеров.
11. Основные устройства компьютеров, их функции и назначение.

ГЛАВА III. ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ КОМПЬЮТЕРОВ

3.1. Этапы решения задачи на компьютере

Программирование (programming) - теоретическая и практическая деятельность, связанная с созданием программ. Решение задач на компьютере включает в себя следующие основные этапы, часть из которых осуществляется без участия компьютера.

1. Постановка задачи:

- сбор информации о задаче;
- формулировка условия задачи;
- определение конечных целей решения задачи;
- определение формы выдачи результатов;
- описание данных (их типов, диапазонов величин, структуры и т.п.).

2. Анализ и исследование задачи, модели:

- анализ существующих аналогов;
- анализ технических и программных средств;
- разработка математической модели;
- разработка структур данных.

3. Разработка алгоритма:

- выбор метода проектирования алгоритма;
- выбор формы записи алгоритма (блок-схемы, псевдокод и др.);
- выбор тестов и метода тестирования;
- проектирование алгоритма.

4. Программирование:

- выбор языка программирования;
- уточнение способов организации данных;
- запись алгоритма на выбранном языке программирования.

5. Тестирование и отладка:

- синтаксическая отладка;

- отладка семантики и логической структуры;
- тестовые расчеты и анализ результатов тестирования;
- совершенствование программы.

6. Анализ результатов решения задачи и уточнение в случае необходимости математической модели с повторным выполнением этапов 2-5.

7. Сопровождение программы:

- доработка программы для решения конкретных задач;
- составление документации к решенной задаче, к математической модели, к алгоритму, к программе, к набору тестов, к использованию.

3.2. Жизненный цикл программного продукта

Все программы можно разбить на два класса по характеру использования:

1. утилитарные программы — предназначены для удовлетворения нужд их разработчиков, программы «для себя»;
2. программные продукты — предназначены для удовлетворения потребностей пользователей, широкого распространения и продажи.

Программный продукт должен быть соответствующим образом подготовлен к эксплуатации, иметь необходимую техническую документацию, предоставлять сервис и гарантию надежной работы программы, иметь товарный знак изготовителя. Только при таких условиях созданный программный комплекс может быть назван **программным продуктом.**

Программный продукт имеет несколько качественных характеристик:

- алгоритмическая сложность;
- полнота функций обработки;
- объём файлов программ;

- требования к операционной системе и техническим средствам обработки со стороны программного средства;

- объём дисковой памяти;

- размер оперативной памяти.

Показатели качества должны содержать следующие аспекты:

- насколько хорошо можно использовать программный продукт (просто, надёжно, эффективно);

- насколько легко эксплуатировать программный продукт;

- можно ли использовать программный продукт при изменении условия его применения.

В условиях существования рынка программных продуктов важными характеристиками являются стоимость, количество продаж, время нахождения на рынке, известность фирмы-производителя и самой программы, наличие на рынке программных продуктов аналогичного назначения. Программный продукт любого вида характеризуется жизненным циклом, состоящим из отдельных этапов.

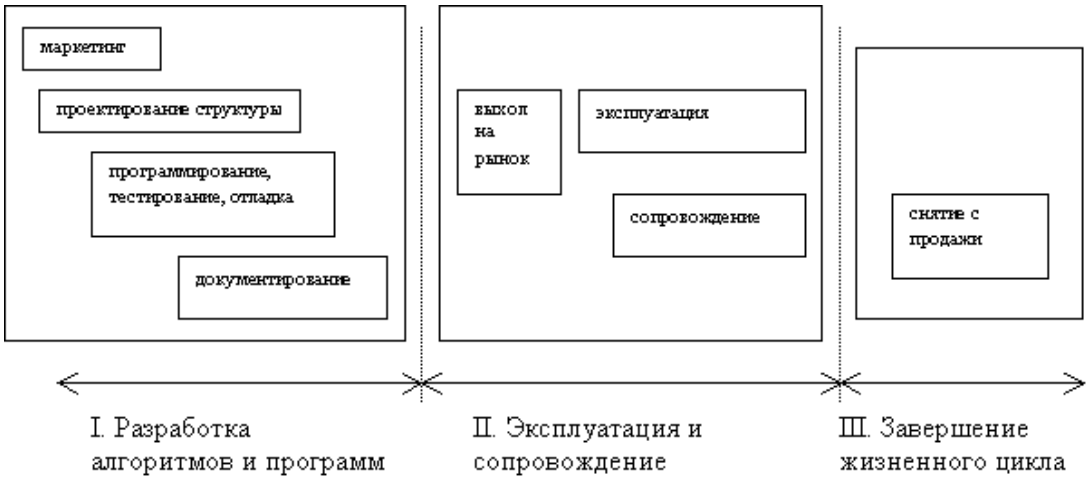


Рис.3.2.1. Жизненный цикл программного продукта.

Маркетинг предназначен для изучения требований к создаваемому программному продукту (технических, программных, пользовательских). Изучаются также существующие аналоги и продукты-конкуренты.

Оцениваются необходимые для разработки материальные, трудовые и финансовые ресурсы, а также устанавливаются примерные сроки разработки.

Проектирование структуры — алгоритмизация процесса обработки данных, детализация функций, разработка архитектурного проекта, выбор методов и средств создания программ.

Программирование, тестирование и отладка — основной этап работы по разработке программного средства. Часто отдельные работы этого этапа ведутся параллельно, что позволяет сократить общее время разработки.

Документирование — обязательный вид работы. Документация должна содержать необходимые сведения по установке, обеспечению надёжной работы продукта, справочное пособие для пользователя, демонстрационные версии, примеры документов, создаваемых при помощи данного программного продукта, обучающие программы.

Выход программного продукта на рынок связан с организацией продаж массовому пользователю. Здесь применяются стандартные методы — реклама, увеличение числа каналов реализации, создание дилерской и дистрибьюторской сети, гибкая ценовая политика.

Эксплуатация и сопровождение идут, как правило, параллельно. В процессе эксплуатации могут выявляться ошибки, и устранение этих ошибок ведётся в режиме сопровождения, то есть оказание сервисной помощи, обеспечение новыми версиями программ, организация «горячих телефонных линий» для консультаций.

Снятие программного продукта с продажи и отказ от его сопровождения происходит, как правило, в случае изменения технической политики фирмы-изготовителя, неэффективности работы программного продукта, наличия в нём неустранимых ошибок, отсутствие спроса. Длительность жизненного цикла разных программных продуктов неодинакова.

Для большинства современных программ его длительность составляет 2-3 года. Хотя часто встречаются на компьютерах и давно снятые с производства программные продукты.

Под *программным обеспечением* (Software) понимается совокупность программ, выполняемых вычислительной системой.

Программное обеспечение – *неотъемлемая часть компьютерной системы*. Оно является логическим продолжением технических средств.

Сам по себе компьютер не обладает знаниями ни в одной области применения. Все эти знания сосредоточены в выполняемых на компьютерах программах.

Программное обеспечение современных компьютеров включает миллионы программ — от игровых до научных.

3.3. Классификация программного обеспечения

В первом приближении все программы, работающие на компьютере, можно условно разделить на **три категории:**

1. **Прикладные программы**, непосредственно обеспечивающие выполнение необходимых пользователям работ.

2. **Системные программы**, выполняющие различные вспомогательные функции, например:

- управление ресурсами компьютера;
- создание копий используемой информации;
- проверка работоспособности устройств компьютера;
- выдача справочной информации о компьютере и др.

3. **Инструментальные программные системы (системы программирования)**, облегчающие процесс создания новых программ для компьютера.

Категории программного обеспечения.

Кроме того, появились нетрадиционные программы, классифицировать которые по устоявшимся критериям очень трудно, а то и просто невозможно, как, например, программа — *электронный собеседник.*

Системы программирования

Система программирования — это система для разработки новых программ на конкретном языке программирования.

Современные системы программирования обычно предоставляют пользователям **мощные и удобные средства разработки программ.** В них входят:

- компилятор или интерпретатор;
- интегрированная среда разработки;
- средства создания и редактирования текстов программ;
- обширные библиотеки стандартных программ и функций;
- отладочные программы, т.е. программы, помогающие находить и устранять ошибки в программе;
- "дружественная" к пользователю диалоговая среда;
- многооконный режим работы;
- мощные графические библиотеки; утилиты для работы с библиотеками;
- встроенный ассемблер;
- встроенная справочная служба;
- другие специфические особенности.

Популярные системы программирования – ***Turbo Basic***, ***Quick Basic***, ***Turbo Pascal***, ***Turbo C***.

В последнее время получили распространение системы программирования, ориентированные на создание *Windows-приложений*.

Инструментальные программы

Инструментальные программные средства — это программы, которые используются в ходе разработки, корректировки или развития других прикладных или системных программ.

По своему назначению они близки системам программирования. К инструментальным программам, например, относятся:

- редакторы;
- средства компоновки программ;
- отладочные программы, т.е. программы, помогающие находить и устранять ошибки в программе;
- вспомогательные программы, реализующие часто используемые системные действия;
- графические пакеты программ и т.п.

Инструментальные программные средства могут оказать помощь на всех стадиях разработки ПО.

Прикладные программы

Прикладная программа — это любая конкретная программа, способствующая решению какой-либо задачи в пределах данной проблемной области.

Какова роль и назначение системных программ?

Системные программы **выполняются вместе с прикладными и служат для управления ресурсами компьютера — центральным процессором, памятью, вводом-выводом.**

Это программы общего пользования, которые **предназначены для всех пользователей компьютера.** Системное программное обеспечение разрабатывается так, чтобы компьютер мог эффективно выполнять прикладные программы.

Среди десятков тысяч системных программ особое место занимают **операционные системы**, которые обеспечивают управление **ресурсами компьютера** с целью их эффективного использования.

Важными классами системных программ являются также программы вспомогательного назначения — **утилиты** (лат. ***utilitas*** — польза). Они либо **расширяют и дополняют соответствующие возможности операционной системы**, либо **решают самостоятельные важные задачи.** Кратко опишем некоторые разновидности утилит:

- **программы контроля, тестирования и диагностики**, которые используются для проверки правильности функционирования устройств компьютера и для обнаружения неисправностей в процессе эксплуатации; указывают причину и место неисправности;
- **программы-драйверы**, которые расширяют возможности операционной системы по управлению устройствами ввода-вывода, оперативной памятью и т.д.; с помощью драйверов возможно подключение к компьютеру новых устройств или нестандартное использование имеющихся;
- **программы-упаковщики** (архиваторы), которые позволяют записывать информацию на дисках более плотно, а также объединять копии нескольких файлов в один архивный файл;
- **антивирусные программы**, предназначенные для предотвращения заражения компьютерными вирусами и ликвидации последствий заражения вирусами.

3.4. Операционные системы

Операционная система — это комплекс взаимосвязанных системных программ, назначение которого — организовать взаимодействие пользователя с компьютером и выполнение всех других программ.

Операционная система выполняет роль связующего звена между аппаратурой компьютера, с одной стороны, и выполняемыми программами, а также пользователем, с другой стороны.

Операционная система обычно хранится во внешней памяти компьютера — *на диске*. При включении компьютера она считывается с дисковой памяти и размещается в *ОЗУ*. Этот процесс называется ***загрузкой операционной системы.***

В функции операционной системы входит:

- осуществление диалога с пользователем;
- ввод-вывод и управление данными;
- планирование и организация процесса обработки программ;
- распределение ресурсов (оперативной памяти и КЭШа, процессора, внешних устройств);
- запуск программ на выполнение;
- всевозможные вспомогательные операции обслуживания;
- передача информации между различными внутренними устройствами;
- программная поддержка работы периферийных устройств (дисплея, клавиатуры, дисковых накопителей, принтера и др.).

В зависимости от количества одновременно обрабатываемых задач и числа пользователей, которых могут обслуживать ОС, различают четыре основных класса операционных систем:

1. Однопользовательские однозадачные, которые поддерживают одну клавиатуру и могут работать только с одной (в данный момент) задачей.

2. Однопользовательские однозадачные с фоновой печатью, которые позволяют помимо основной задачи запускать одну дополнительную задачу, ориентированную, как правило, на вывод информации на печать. Это ускоряет работу при выдаче больших объёмов информации на печать.

3. **Однопользовательские многозадачные**, которые обеспечивают одному пользователю параллельную обработку нескольких задач. Например, к одному компьютеру можно подключить несколько принтеров, каждый из которых будет работать на "свою" задачу.

4. **Многопользовательские многозадачные**, позволяющие на одном компьютере запускать несколько задач нескольким пользователям. Эти ОС очень сложны и требуют значительных машинных ресурсов.

ГЛАВА IV. ОПЕРАЦИОННАЯ СИСТЕМА WINDOWS

4.1. Версии операционной системы

При создании Windows 95 фирма Microsoft в полной мере реализовала объектно-ориентированный подход. Поскольку именно он лег в основу новой операционной системы, вначале скажем несколько слов о том, что такое ориентация на объекты.

Понятие «объектно-ориентированный» возникло в программировании сравнительно недавно. Когда вычислительная мощность машин была невысока, о создании объектно-ориентированных систем не могло быть и речи. Основой всего был программный код. Программисты записывали последовательности команд для выполнения тех или иных действий над данными, которые оформлялись в модули и процедуры. Для работы с каждым объектом создавалась своя процедура.

Объекты, их свойства и методы.

Постепенно с увеличением производительности вычислительных систем процедурный подход начал заменяться объектным. На первое место выдвинулся объект, а не код, который его обрабатывает. На уровне пользователя объектный подход выражается в том, что интерфейс представляет собой подобие реального мира, а работа с машиной сводится к действиям с привычными объектами. Так, папки можно открыть, убрать в портфель, документы — просмотреть, исправить, переложить с одного места на другое, выбросить в корзину, факс или письмо — отправить адресату и т. д. Понятие объекта оказалось настолько широким, что до сих пор не получило строгого определения.

Объект, как и в реальном мире, обладает различными свойствами. Программист или пользователь может изменять не все свойства объектов, а только некоторые из них. Можно изменить имя объекта, но нельзя изменить объем свободного места на диске, который также является его свойством.

Свойства первого типа в языках программирования носят название read/write (для чтения и записи), а свойства второго — read only (только для чтения).

Метод — это способ воздействия на объект. Методы позволяют создавать и удалять объекты, а также изменять их свойства. Например, для того чтобы нарисовать на экране точку, линию или плоскую фигуру, составляются разные последовательности кодов или программы. Пользователь, однако, применяет для отображения этих объектов один метод Draw( ), который содержит коды для отображения всех объектов, с которыми он работает. За такое удобство приходится платить тем, что объектно-ориентированные системы могут работать только на достаточно мощных вычислительных установках.

Процедурный подход в ранних ОС.

До настоящего времени во всех операционных системах преобладал процедурный подход. Для того чтобы произвести в системе какое-либо действие, пользователь должен был вызвать соответствующую программу (процедуру) и передать ей определенные параметры, например, имя обрабатываемого файла. Программа выполняла над файлом указанные действия и заканчивала работу. При этом пользователь в первую очередь имел дело с задачей обработки документа, а затем уже с самим документом. В давние времена, когда ЭВМ не были персональными, пользователь описывал действия, которые должна была выполнить задача, на некоем странном языке, называемом языком управления заданиями (JCL—Job Control Language).

С появлением терминала язык управления заданиями упростился и постепенно превратился в командную строку, однако на первом месте все равно находилась процедура обработки документа, а сам документ играл вспомогательную роль. Следующим этапом упрощения работы с машиной стал создание различного рода операционных оболочек (сначала текстовых), которые «спрятали» от пользователя командную строку DOS.

Ввод последовательности символов, из которой состоит команда операционной системы, свелся к нажатию одной функциональной клавиши или щелчку мыши. Самой распространенной из таких «надстроек» над операционной системой стала оболочка Norton Commander,

Однако основным «инструментом» пользователя все еще оставалась клавиатура. Качественный переход произошел после того, как появились графические оболочки. Теперь пользователь в основном работает с устройством указания, таким как мышь, трекбол или планшет, а не с клавиатурой (разумеется, это не относится к работе внутри самих приложений, например, в текстовых редакторах). Ему не нужно помнить почти никаких команд операционной системы. Для того чтобы запустить приложение, достаточно щелкнуть мышью на его изображении или на «значке» (автор предпочитает называть его пиктограммой).

От процедурного подхода к объектно-ориентированному.

В начале 90-х гг. процедурный подход все еще преобладает, однако намечаются и некоторые признаки объектно-ориентированного. Например, уже в Windows 3.0 можно поставить в соответствие конкретному документу приложение для его обработки. Тогда же появился метод объектного связывания и встраивания (OLE), позволяющий щелчком на изображении объекта неявно запустить приложение, которое его обрабатывает, а после окончания обработки вернуться в предыдущее приложение. С OLE тесно связан так называемый метод редактирования документов «на месте» (in-place). Если в документ встроен объект, который должен обрабатываться конкретным приложением, то при щелчке на этом объекте нужное приложение неявным образом запускается, причем в рабочем поле не изменяется ничего, кроме панелей инструментов. Например, если в тексте, который обрабатывается в редакторе Microsoft Word, есть таблица, созданная в редакторе Microsoft Excel, то при щелчке на ней произойдет замена *панелей инстр*ументов Excel. Пользователь может обрабатывать документ совсем другим приложением, даже не подозревая об этом.

Еще один механизм, который упростил работу и приблизил эру объектно-ориентированного подхода, называется «Drag & Drop», что в буквальном переводе означает «перетащить-и-оставить». Работая этим методом, вы щелкаете кнопкой мыши (как правило, левой) на изображении объекта, перемещаете его по экрану при нажатой кнопке и отпускаете кнопку, когда указатель окажется в нужном месте экрана. Таким образом, процедуры копирования, перемещения и удаления стали объектно-ориентированными.

Что делал пользователь, когда ему нужно было удалить файлы в Операционной системе MS-DOS? Он запускал процедуру удаления файлов, передавая их имена в качестве параметров:

Del FILEI.TXT FILE2TXT.

Это действие ничем не напоминает реальный мир, в котором вы просто выбрасываете ненужные

Бумаги в мусорную корзину. На первом месте для нас стоит объект (бумага), над которым выполняется процедуры (переноса в мусорную корзину), R операционных оболочках, которые работают под управлением Windows 3.1, такое действие уже реализовано как объектно-ориентированное — с помощью механизма «Draw & Drop». Например, в оболочке Norton Desktop можно схватить мышью файл и перенести его на изображение мусорной корзины. Этого достаточно для удаления файла. Так работа на персональном компьютере все больше напоминает манипуляции с объектами в реальном мире.

Выбор показателей и параметров для оценки ОС.

- *Windows 95 — объектно-ориентированная ОС.*
- *Windows 95—полноценная операционная система.*
- *Использование стандарта Plug & Play.*
- *32-разрядная ОС защищенного режима.*
- *Приоритетная многозадачность.*

- *Многопоточность.*
- *Спулер печати.*
- *32-разрядные устанавливаемые файловые системы.*
- *Средства удаленного доступа.*
- *Возможности работы с мультимедиа.*
- *Поддержка приложений MS-DOS.*
- *Поддержка длинных имен файлов.*
- *Интерфейс пользователя.*
- *Работа с памятью.*

Сравнительная оценка ОС ПВЭМ по выбранным показателям Windows 95 по сравнению с Windows 3.1

Принципиальная новизна операционной системы Windows 95 состоит именно в том, что концепция объектно-ориентированного подхода реализована в ней наиболее полно.

Windows 95 — объектно-ориентированная ОС.

Объектно-ориентированный подход реализуется через модель рабочего стола. Windows 95 обходится без привычного в Windows 3.1 ДИСПЕТЧЕРА ПРОГРАММ (PROGRAM MANAGER). Пользователь работает с задачами и приложениями так же, как с документами на своем письменном столе.

Это удобно для людей, которые первый раз увидели компьютер, но создает некоторые трудности «переходного периода» для тех, кто привык считать программу основой всего сущего в машине.

Итак, одно из главных отличий Windows 95 от Windows 3.1 (и от подавляющего большинства других операционных систем) состоит в том, что основной упор в ней делается на документ, а программа, задача, приложение или программный код вообще рассматриваются только как инструмент для работы с документом.

Windows 95—полноценная операционная система.

Другая принципиальная особенность Windows 95 состоит в том, что она, в отличие от Windows 3.1. является «настоящей» операционной системой (а не операционной оболочкой, выполняемой под управлением MS-DOS). Под словом «настоящая» мы подразумеваем то, что при включении машины сразу выполняется загрузка Windows 95. Для пользователя это оборачивается некоторыми неудобствами. Он должен привыкнуть к тому, что прежде чем выключить машину, нужно корректно завершить работу с Windows 95, поскольку новая операционная система создает буфера в оперативной памяти, и их содержимое должно быть сброшено на диск.

Использование стандарта Plug & Play.

Подход к аппаратному обеспечению также кардинальным образом изменился. Теперь система использует стандарт Plug & Play (переводится как «включил-и-работай», произносится чаще всего как «плаг-н-плэй»), что облегчает и максимально автоматизирует процесс добавления новых периферийных устройств. Стандарт Plug & Play — это совместная разработка фирм Intel и Microsoft. Основная его идея заключается в том, что каждое устройство, соответствующее этому стандарту, сообщает о себе определенную информацию, благодаря которой операционная система выполняет автоматическую конфигурацию периферийных устройств и разрешает аппаратные конфликты. Стандарту Plug & Play должен в первую очередь удовлетворять BIOS материнской платы и, разумеется, периферийные устройства. Таким образом, операционная система обеспечивает автоматическое подключение и конфигурирование устройств, соответствующих требованиям стандарта Plug and Play, поддерживает совместимость с устаревшими устройствами и создает динамическую среду для подключения и отключения мобильных компонентов.

32-разрядная ОС защищенного режима.

MS-DOS была чисто 16-разрядной операционной системой и работала в реальном режиме процессора. В версиях Windows 3.1 часть кода была 16-разрядной, а часть — 32-разрядной. Windows 3.0 поддерживала реальный режим работы процессора, при разработке версии 3.1 было решено отказаться от его поддержки. Windows 95 является 32-разрядной операционной системой, которая работает только в защищенном режиме процессора. Ядро, включающее управление памятью и диспетчеризацию процессов, содержит только 32-разрядный код. Это уменьшает издержки и ускоряет работу. Только некоторые модули имеют 16-разрядный код для совместимости с режимом MS-DOS. Windows 95 32-разрядный код используется везде, где только возможно, что позволяет обеспечить повышенную надежность и отказоустойчивость системы. Помимо этого, для совместимости с устаревшими приложениями и драйверами используется и 16-разрядный код.

Приоритетная многозадачность.

В отличие от предыдущих версий, Windows 95 поддерживает приоритетную многозадачность (preemptive multitasking) и параллельные процессы (multithreading). В Windows.3.0 существовала так называемая «вытесняющая многозадачность» (non-preemptive multitasking), при которой за распределение процессорного времени отвечало приложение. Система выполняла задачу до тех пор, пока приложение «добровольно» не отдавало процессор. В Windows 95 за распределение времени процессора отвечает ядро системы, что обеспечивает нормальную работу фоновых задач.

Многопоточность.

Windows 95 поддерживает многопоточность - технологию, которая позволяет соответствующим образом осуществлять многозадачное выполнение своих собственных процессов.

Спулер печати.

Спулер печати кардинально переработан по сравнению с Windows 3+. Теперь параллельно с печатью можно делать что-либо еще (в старой оболочке можно было или печатать, или работать). Спулер печати также стал теперь 32-разрядным.

32-разрядные устанавливаемые файловые системы.

Эта часть операционной системы стала гораздо более производительной, чем аналогичные компоненты Windows 3+. Для жестких дисков используются виртуальные таблицы распределения файлов (VFAT), а для компакт-дисков — новая файловая система CDFS (CD-ROM File System). При этом имена файлов могут содержать до 255 знаков, включая пробелы и специальные символы (совместимость со старой файловой системой сохранена, хотя и несколько искусственным путем.. Теперь в большинстве случаев не требуется модуль MSCDEX EXE, выполнявший преобразование файловой системы стандарта ISO-9660 (компакт-диска) к файловой системе MS-DOS.

Устанавливаемая файловая система, которая отображает файловую структуру удаленной машины на сетевой диск рабочей станции, называется сетевым редиректором. Сетевые редиректоры для протоколов IPX/SPX и NetBEU также используют 32-разрядный код. Протокол NetBEU применяется при работе Windows 3.1, а IPX/ SPX—для связи с машинами, на которых установлена Windows NT,

Средства удаленного доступа.

Windows 95, в отличие от большинства операционных систем для персональных компьютеров, с самого начала создавалась для работы в сети, благодаря чему возможность совместного использования файлов и устройств полностью интегрирована в интерфейс пользователя Windows 95. В Windows 95 вы можете получить доступ к сети без установки сетевого адаптера! Его заменят модем и специальный протокол PPP («от-точки-к-точке», или «point-to-point protocol»).

В этом случае скорость работы ограничена скоростью вашего модема. Система предоставляет развитые программные средства для доступа к сетям Internet, Microsoft Network, America Online и другим аналогичным службам.

Возможности работы с мультимедиа.

Современную операционную систему сложно представить себе без средств мультимедиа. Для работы с аудио- и видеофайлами различных форматов в составе Windows 95 имеется набор кодеков — эффективных программных средств сжатия и распаковки этих файлов и преобразования их форматов для вывода на различные устройства мультимедиа (слово «кодер» является сокращением слов «кодер-декодер», так же, как «модем» — сокращение от слов «модулятор-демодулятор»). При воспроизведении файла система запускает тот кодер, с помощью которого файл был создан. Драйверы звуковых карт используют 32-разрядный код, но в тех случаях, когда система не может распознать карту, применяется 16-разрядный драйвер реального режима, который поставляется вместе с картой. При работе 32-разрядного драйвера защищенного режима драйвер реального режима автоматически отключается. При установке компакт-диска в устройство считывания система пытается распознать его формат и запустить соответствующее приложение для его воспроизведения. Если установлен диск формата ISO-9660 (программный), то Windows 95 ищет файл с именем AUTO-RUN.INF u выполняет его. Это механизм получил название Spin & Grin. Значительно переработан код, который отвечает за обработку изображений. поэтому качество воспроизведения файлов AVI сильно возросло по сравнению с Windows 3+, а скорость их воспроизведения теперь почти не зависит от выбранного масштаба изображения. Встроенные возможности работы со звуком, видео и компакт-дисками дадут новый толчок развитию приложений мультимедиа. Windows 95 - это первая версия Windows, которая бросает вызов MS-Dos в сфере поддержки игрового программного обеспечения.

Поддержка приложений MS-DOS.

Windows 95 занимает меньше места в основной памяти, так что теперь вы можете запускать многие из тех программ MS-DOS, которые не работали под управлением Windows 3.0. Программы, которые и сейчас не будут помещаться в память, можно запескать в *режиме эмуляции* MS-DOS. Переключаясь в этот режим, Windows 95 завершает все работающие приложения, а потом удаляет из памяти и саму себя, оставляя лишь маленький загрузочный модуль. Закончив работать с программой

Поддержка длинных имен файлов.

Вы сможете забыть об ограничениях на длину имени файла в системах Windows 3.0 и MS-DOS.В Windows 95 имена файлов могут иметь длину до 255 символов.

Интерфейс пользователя.

Благодаря новому интерфейсу в Windows 95, по сравнению с Windows 3.0. гораздо проще запускать программы, открывать и сохранять документы, работать с дисками и сетевыми серверами.

Работа с памятью.

Windows 95 автоматически освобождает всю память, отведенную приложению, после того, как оно заканчивает работу. В Windows 3.0 некорректно написанные приложения нередко освобождали не всю запрошенную ими память. Время от времени памяти оказывалось настолько мало, что единственным выходом оставался перезапуск системы (а иногда и перезагрузка машины). Такая неприятность носит название «утечка памяти» («memory leak») и случается с программными произведениями даже известнейших фирм. При завершении приложения в Windows 95 вся память, занимаемая им, освобождается автоматически, и таких проблем не возникает.

Перспективы развития ОС ПВЭМ Windows NT.

На данный момент мировая компьютерная индустрия развивается очень стремительно.

Производительность систем возрастает, а следовательно возрастают возможности обработки больших объёмов данных. Операционные системы класса MS-DOSa уже не справляются с таким потоком данных и не могут целиком использовать ресурсы современных компьютеров. Поэтому в последнее время происходит переход на более мощные и наиболее совершенные операционные системы класса UNIX , примером которых и является Windows NT, выпущенная корпорацией Microsoft.

Задачи, поставленные при создании Windows NT.

Система Windows NT не является дальнейшим развитием ранее существовавших продуктов . Её архитектура создавалась с нуля с учётом предъявляемых к современной операционной системе требований . Особенности новой системы ,разработанной на основе этих требований, перечислены ниже .

- Стремясь обеспечить *совместимость* (compatible) новой операционной системы,разработчики Windows NT сохранили привычный интерфеис Windows и реализовали поддержку существующих файловых систем (таких ,как FAT) и различных приложений (написанных для MS - Dos ,OS/2 1.x ,Windows 3.x и POSIX).Разработчики также включили в состав Windows NT средства работы с различными сетевыми средствами.

- Достигнута *переносимость* (portability) системы, которая может теперь работать как на CISC , так и на RISC - процессорах. К CISC относятся Intel - совместимые процессоры 80386 и выше; RISC представлены системами с процессорами MIPS R4000 , Digital Alpha AXP и Pentium серии P54 и выше.

- *Масштабируемость* (scalability) означает, что Windows NT не привязана к однопроцессорной архитектуре компьютеров, а способна полностью использовать возможности , предоставляемые симметричными мультипроцессорными системами .В настоящее время Windows NT может функционировать на компьютерах с числом процессоров от 1 до 32 . Кроме того, в случае усложнения стоящих перед пользователями задач и

расширения предъявляемых к компьютерной среде требований, Windows NT позволяет легко добавлять более мощные и производительные серверы, и рабочие станции к корпоративной сети. Дополнительные преимущества даёт использование единой среды разработки и для серверов, и для рабочих станций.

- Windows NT имеет однородную *систему безопасности* (security) удовлетворяющую спецификациям правительства США и соответствующую стандарту безопастности B2. В корпоративной среде критическим приложениям обеспечивается полностью изолированное окружение.

- *Распределённая обработка* (distributed processing) означает, что Windows NT имеет встроенные в систему сетевые возможности. Windows NT также позволяет обеспечить связь с различными типами хост - компьютеров благодаря поддержке разнообразных транспортных протоколов и использованию средств “клиент-сервер” высокого уровня ,включая именованные каналы ,вызовы удалённых процедур (RPC - remote procedure call) и Windows - сокеты.

- *Надёжность и отказоустойчивость* (reliability and robustness) обеспечивают архитектурными особенностями, которые защищают прикладные программы от повреждения друг другом и операционной системой. Windows NT использует отказоустойчивую структурированную обработку особых ситуаций на всех архитектурных уровнях, которая включает восстанавливаемую файловую систему NTFS и обеспечивает защиту с помощью встроенной системы безопасности и усовершенствованных методов управления памятью.

- Возможности *локализации* (allocation) представляют средства для работы во многих странах мира на национальных языках, что достигается применением стандарта ISO Unicod (разработан международной организацией по стандартизации).

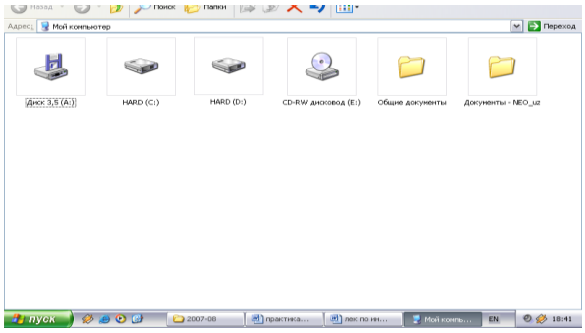
• Благодаря модульному построению системы обеспечивается *расширямость* (insibility) Windows NT, что, как будет показано в следующем разделе, позволяет гибко осуществлять добавление новых модулей на различные уровни операционной системы.

4.2. Основные элементы WINDOWS

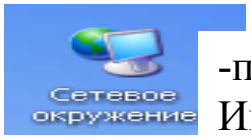
Рабочим столом (desktop) называют всю поверхность экрана во время работы оболочки Windows.

Основные элементы рабочего стола:

-располагаются ресурсы подсоединения объектом, устройства со съемными носителями, жесткий диск, файлы хранящиеся на этом компьютере;



-папка, в которой находятся документы созданные пользователем;



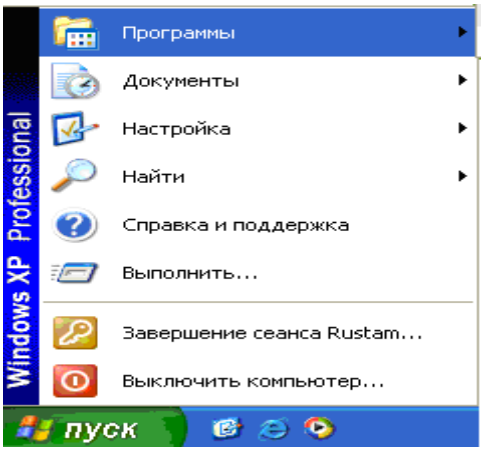
-помощь при регистрации и подключении к Интернету;

- отправляются файлы, папки, значки и другие легко уничтожаемые объекты.

Просмотр веб-страниц.

Для управления работой нескольких программ удобно использовать панель задач, расположенную в нижней части рабочего стола Windows XP.

В левой части панели расположена кнопка **Пуск** (Start), с которой вы уже знакомы. Правее этой кнопки обычно расположена панель инструментов почти с такими же значками, как и на рабочем столе Windows



Быстрый запуск (Quick Launch) и предназначена для запуска популярных и часто используемых программ. В правой части панели задач располагается панель индикации. На ней отображается текущее время и значки различных запущенных программ.

Чтобы создать папку сперва выбираем место, затем нажимаем правую кнопку мыши открывается контекстное меню, где выбираем СОЗДАТЬ - ПАПКУ – задаём имя – ОК. Меню – это просто список некоторых объектов, из которого вам необходимо сделать выбор.

В Windows мы различаем 3 типа меню:

- *Главное меню* открывается при нажатии кнопки ПУСК. Её задача запустить программы, открыть документы, настройка, находить нужные файлы и т.д.

- *Контекстное меню* открывается при нажатии правой части мышки, это зависимое меню.

- *Диалоговое меню* это программное меню.

Пиктограмма – это небольшая цветная картинка (иконка), которая представляет на экране дисплея некоторую программу, окно, функцию, файл и т.п. Пиктограмма часто имеет название или пояснительный текст.

Linux также **Ли́нукс** — общее название Unix-подобных операционных систем, основанных на одноимённом ядре. Ядро Linux и обычно использующиеся вместе с ним компоненты создаются и распространяются в соответствии с моделью разработки свободного и открытого программного обеспечения. Поэтому общее название не подразумевает какой-либо единой «официальной» комплектации Linux; они обычно распространяются (часто бесплатно) в виде различных готовых дистрибутивов, имеющих свой набор прикладных программ и уже настроенных под конкретные нужды пользователя. Для сравнения, аналогом дистрибутива у Microsoft в одном смысле является «линейка Windows NT», в другом — продукт (версия, выпуск) из этой линейки «Windows XP Professional x64 Edition».

На начальном этапе Linux бесплатно разрабатывался только энтузиастами-добровольцами, но с успехом Linux и его массовым коммерческим использованием дорабатывать ОС и вносить свой вклад стали и компании, со временем став значительной силой. Всё ПО [*прояснить*] по-прежнему бесплатно доступно по свободным лицензиям. В 2008 году расчёты показывали, что для того чтобы «с нуля» разработать систему, аналогичную Fedora 9, потребовалось бы затратить 10,8 млрд долл.

Совокупная стоимость ядра Linux оценена в более чем 1 млрд евро (около 1,4 млрд долл.). Только за 2008 год ценность ядра Linux увеличилась

на 225 млн. евро. В системе Linux воплощён труд в эквиваленте 73 тыс. человеко-лет. В настоящее время системы Linux лидируют на рынках смартфонов (Android занимает 64,1 % рынка), интернет-серверов (60 %), самых мощных суперкомпьютеров (93,8 %), а также, согласно Linux Foundation, в дата-центрах и на предприятиях, занимают половину рынка встраиваемых систем, имеют значительную долю рынка нетбуков (32 % на 2009 год). На рынке домашних компьютеров Linux прочно занимает 3 место (по разным данным, от 1 до 5 %). Согласно исследованию Goldman Sachs, в целом, рыночная доля Linux среди электронных устройств составляет около 42 %. Linux 2 С тех пор как ядро Linux было создано для x86-ПК, оно было портировано на множество платформ и процессоров, включая x86-64, PowerPC и ARM. Linux работает в роутерах, телевизорах и игровых приставках.

ОС на ядре продолжают быстро совершенствоваться (например, новая версия ядра выпускается каждые 2-3 месяца, с 2005 года в разработке ядра принимают участие более 7800 разработчиков из более чем 800 различных компаний) и набирать популярность (за 9 месяцев с мая 2011 по январь 2012 доля Linux выросла на 64 %).

Согласно distrowatch, наиболее популярными дистрибутивами являются: deb-based (Debian, Mint, Ubuntu), RPM-based (RedHat, Fedora, Mageia, OpenSUSE), source-based (Slackware, Gentoo).

Собственные дистрибутивы Linux выпускаются различными компаниями и энтузиастами со всего мира, в том числе, например, из России и Украины.

UNIX (читается ю́никс) — семейство переносимых, многозадачных и многопользовательских операционных систем.

Первая система UNIX была разработана в 1969 году в подразделении Bell Labs компании AT&T.

С тех пор было создано большое количество различных UNIX-систем. Юридически лишь некоторые из них имеют полное право называться

«UNIX»; остальные же, хотя и используют сходные концепции и технологии, объединяются термином «UNIX-подобные» (англ. UNIX-like). Для краткости в данной статье под UNIX-системами подразумеваются как истинные UNIX, так и UNIX-подобные ОС.

Некоторые отличительные признаки UNIX-систем включают в себя:

• использование простых текстовых файлов для настройки и управления системой;

• широкое применение утилит, запускаемых в командной строке;

•взаимодействие с пользователем посредством виртуального устройства — терминала;

• представление физических и виртуальных устройств и некоторых средств межпроцессового взаимодействия как файлов;

• использование конвейеров из нескольких программ, каждая из которых выполняет одну задачу.

В настоящее время UNIX-системы используются в основном на серверах, а также как встроенные системы для различного оборудования. На рынке ОС для рабочих станций и домашнего применения лидером является Microsoft Windows, UNIX занимает только второе (Mac OS X), третье (GNU/Linux) и многие последующие места.

UNIX-системы имеют большую историческую важность, поскольку благодаря им распространились некоторые популярные сегодня концепции и подходы в области ОС и программного обеспечения. Также, в ходе разработки Unix-систем был создан язык Си.

Среди примеров известных UNIX-подобных операционных систем: BSD, Solaris, Linux, Android, MeeGo, NeXTSTEP, Mac OS X.

Первые версии UNIX были написаны на ассемблере и не имели встроенного компилятора с языком высокого уровня. Примерно в 1969 году Кен Томпсон при содействии Денниса Ритчи разработал и реализовал язык Би (B), представлявший собой упрощённый (для реализации на миникомпьютерах) вариант разработанного в 1966 языка BCPL. Би, как и

BCPL, был интерпретируемым языком. В 1972 году была выпущена вторая редакция UNIX, переписанная на языке Би. В 1969—1973 годах на основе Би был разработан компилируемый язык, получивший название Си (C).

В 1973 году вышла третья редакция UNIX, со встроенным компилятором языка Си. 15 октября того же года появилась четвёртая редакция, с переписанным на Си системным ядром (в духе системы Multics, также написанной на языке высокого уровня ПЛ/1), а в 1975 — пятая редакция, полностью переписанная на Си.

С 1974 года UNIX стал распространяться среди университетов и академических учреждений. С 1975 года началось появление новых версий, разработанных за пределами Bell Labs, и рост популярности системы. В том же 1975 году Bell Labs выпустила шестую редакцию, известную по широко разошедшимся комментариям Джона Лайонса.

К 1978 году система была установлена более чем на 600 машинах, прежде всего, в университетах. Седьмая редакция была последней единой версией UNIX. Именно в ней появился близкий к современному интерпретатор командной строки Bourne shell.

Вопросы для закрепления знаний:

1.Что представляет с собой ОС Windows?

2. Поколение Windows, преимущества и недостатки ОС?

3.Что такое рабочий стол?

4. Элементы рабочего стола.

5. Панель задач Windows.

6. Сколько вида меню Windows?

7. Работа с окнами.

8. Как создаются папки в Windows?

9. Работа файлами.

ГЛАВА V. ОСНОВЫ АЛГОРИТМИЗАЦИИ

Термин «алгоритм» своим происхождением обязан имени узбекского математика Ал-Хорезми, который ещё в IX-м веке сформулировал правила выполнения четырёх арифметических действий. Появившееся несколько позже слово «алгоритм» связано с именем древнегреческого математика Евклида, назвавшего так сформулированные им правила нахождения наибольшего общего делителя двух чисел. Многие годы понятие «алгоритм» использовалась математиками для описания правил решения математических задач. После появления вычислительной техники алгоритм началось использоваться при создании программы для машин.

Раньше чтобы получить результат какой-то задачи составляли алгоритм и программу, вводили в компьютер и получали ответ. Если программа составлена правильно, и не было никаких ошибок, тогда ответ получали быстро, если программа составлена неправильно, то ждали ответ долго.

Сейчас все по другому, графическая операционная система позволяет, не прибегая программированию, использовать готовый продукт.

В форме различных инструкций и правил, алгоритмы сопровождают человека во всей его жизни. Каждый из нас, не замечая этого, ежедневно решает задачи, для описания которых используется тот или иной алгоритм. Например, алгоритмом является кулинарный рецепт, инструкция по включению и эксплуатации различных электробытовых приборов, использование телефона-автомата, приготовление чашки чая, медицинские рекомендации и т.д. Каждый алгоритм создается конкретным автором (человеком или группой людей).

Чтобы заставить компьютер решить какую- либо задачу, необходимо, прежде всего, разработать алгоритм решения, выполнить следующие основные работы:

1) Разработать описание задачи.
2) Выбрать математический метод решения задачи.
3) Описать алгоритм (составить блок-схему алгоритма).

4) Составить программу на алгоритмическом языке.

5) Ввести программу в память компьютера.

6) Отладить программу, а затем протестировать ее.

7) Решить задачу на компьютере.

8) Оформить документацию программы.

Алгоритм - это конечная последовательность однозначных предписаний, исполнение которых позволяет с помощью конечного числа шагов получить решение задач, однозначно определяемые исходными данными.

Алгоритм должен обладать следующими свойствами:

1.	***Определенностью*** (детерминированностью)-	требует, чтобы каждая команда алгоритма была понятна исполнителю. Это означает, что сколько бы раз мы не применяли определенный алгоритм к одним и тем же исходным данным, каждый раз мы должны получать один и тот же результат
2.	***Массовостью*** (универсальностью)-	то есть возможностью использовать данный алгоритм не только для решения данной конкретной задачи (с ее конкретными исходными данными), но и для решения всех других однотипных задач (с другими исходными данными)
3.	***Результативностью*** (сходимостью)-	то есть неизбежностью получения конечного результата (решения) после выполнения определенного (не бесконечного) количества шагов вычисления

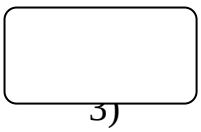
Устанавливаемая алгоритмом последовательность действий задается *словесной, графической форме и в виде формул.*

Блок-схема алгоритма - это алгоритм решения задачи, в виде графических обозначений с примечаниями.

1)

-процесс, проводятся вычисления или присваиваются значения.

2)

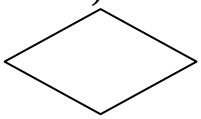


-обозначает начало и конец. Начало обработки информации или конец.

3)

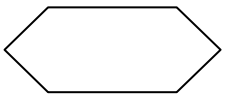
-ввод, вывод информации.

4)



- проверка условия. Логический блок, по условиям выполнения или невыполнения проводится разветвление вычислительного процесса.

5)



- циклический алгоритм.

6)

-вывод документа на печать (или сканирование- ввод с бумаги).

7)

- соединение потока информации.

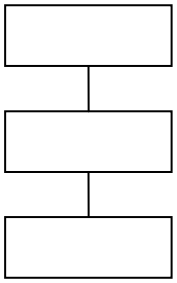
8) _____ - связь между блоками.

9)

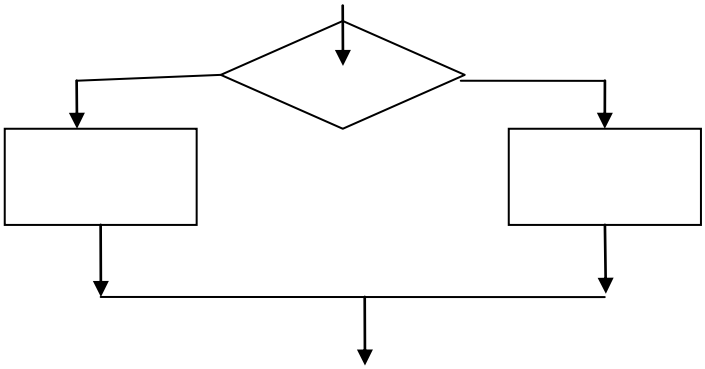
-стандартные блоки. Использование заранее разработанного алгоритма или программы.

В основном алгоритм подразделяется на 3 вида: *линейный, разветвляющий, циклический.*

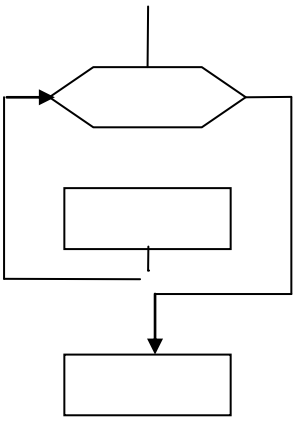
1) Линейный алгоритм. Процесс выполняется последовательно.



2)Разветвляющий алгоритм. Процесс вычисления зависит от условия. По условию выполняется та или иная задача.



1) Циклический алгоритм. Процесс повторяется несколько раз с другими значениями.



Пример 1. Вычислить уравнение:

$$У= 2*X^2 +X-1 \text{ где } X=1.$$

Используем для этого примера словесный алгоритм, алгоритм в виде формул и блок-схемы.

Алгоритм словесный и в виде формул имеет след. вид:

1. Начало алгоритма ПРИМЕР1.

2. Ввод значения X.

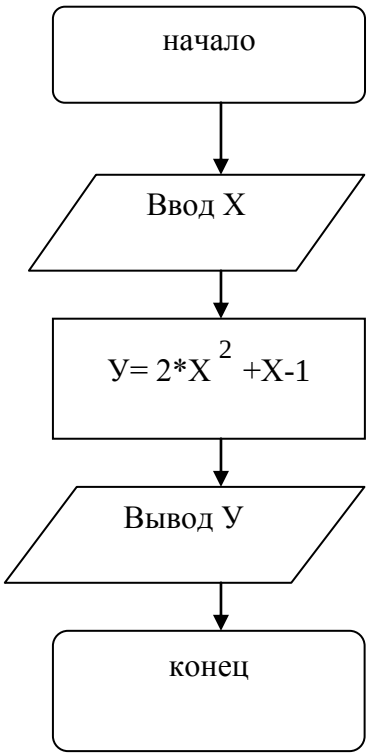
3. Вычисление У.

$У=2*X^2+X-1$

4. Вывод значения У.

5. Конец.

Алгоритм в виде блок-схемы для этого примера имеет вид:



Пример 2:

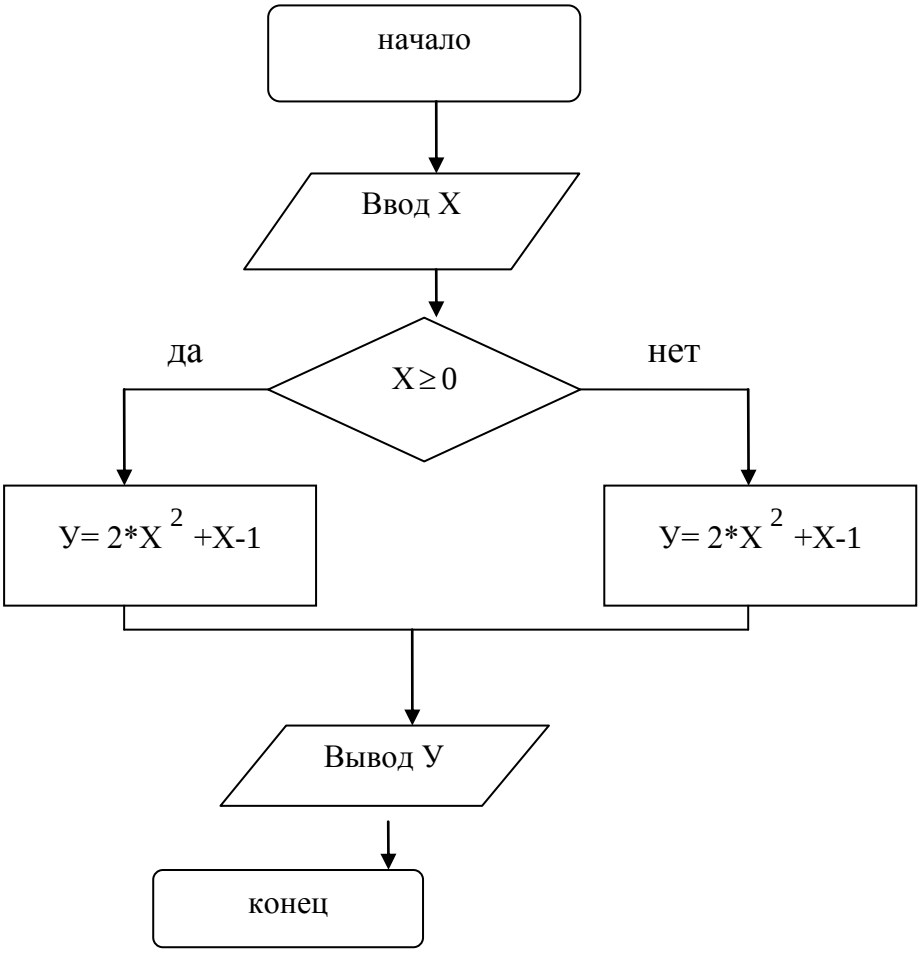
$$У=\begin{cases} X, \text{ если } X\geq 0 \\ -X, \text{ если } X<0 \end{cases} \quad \text{где: } X=5$$

Алгоритм словесный и в виде формул имеет след. вид:

1. Начало алгоритма ПРИМЕР2.
2. Ввод значения X.
3. Проверка условия если $X\geq 0$ то переходи к пункту 6 иначе.
4. Вычисление У= -X.

5. Идти к пункту 7.
6. Вычисление У=Х.
7. Вывод значения У.
8. Конец.

Алгоритм в виде блок-схемы для этого примера имеет вид:



Пример 3. Вычислить уравнение.

$$У= 2*X^2 +X-1, \text{ где: } X=[1,10] \text{ шагом } h=1.$$

Используем для этого примера словесный алгоритм, алгоритм в виде формул и блок-схемы.

Алгоритм словесный и в виде формул имеет след. вид:

1. Начало алгоритма ПРИМЕР3.
2. Циклический процесс Х=1,10,1.
3. Вычисление У.

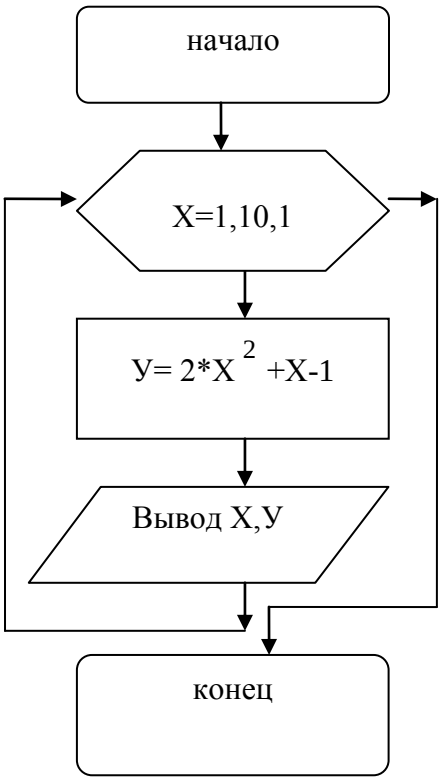
$У=2*X^2+X-1$.

4. Вывод значения Х,У.

5. Возврат к 2 до тех пор пока Х не достигнет конечного шага.

6. Конец.

Алгоритм в виде блок-схемы для этого примера имеет вид:



Вопросы для закрепления знаний:

1. Что такое алгоритм?

2. Какими свойствами обладает алгоритм?

3. В каком виде представляется алгоритм?

4. Какие бывают типы алгоритмов?

5. Как создаются линейные алгоритмы?

6. Как строится разветвляющий алгоритм?

7. Создание циклического алгоритма.

8. Какие блоки используются для создания алгоритма?

9. Какие машинные языки используются для решения алгоритмов?

5.1. Программирование на языке С++. Состав и алфавит языка

Создатель С++(С с классами) - Бьярн Страуструп, сотрудник знаменитой AT&TBell Labs, где были разработаны операционная система UNIX и язык С, придумал один из самых сложных языков программирования (в начале 1980 годов). Собственно С/С++является самым популярным языком для создания программ любой сложности и на любой платформе. Самая стабильная и надёжная ОС UNIX была изначально полностью написана на С, с небольшим добавлением ассемблерного кода. Все основные программы для этой ОС, в том числе популярнейший в Internet веб-сервер Apache, также написаны на С. Ядро ОС Windows также использует API, написанные на этом языке.

В качестве базового языка для С++ был выбран С, потому что он:

- многоцелевой, лаконичный и относительно низкого уровня;
- отвечает большинству задач системного программирования;
- пригоден в среде программирования UNIX.

Популярность С/С++ связана с его расширяемостью, свободой написания программ, поддержкой со стороны крупнейших компаний, выпустивших свои компиляторы, и конечно же с принципом модульности и его воплощением в виде миллиардов строк кода, пригодного для повторного использования с минимальными изменениями.

С++ применяется в очень широком спектре задач, связанных с современными технологиями. На нём строятся ядра операционных систем. Практически все современные программные продукты в той или иной степени имеют под собой основу на языке С++. Компанией Microsoft была разработана библиотека MFC (Microsoft Foundation Classes), которая используется при создании графических приложений на С++ с помощью среды Visual C++. В ней реализованы классы всех стандартных элементов управления Windows. Borland выпустила альтернативную среду разработки

– C++ Builder, практически представляющую собой тот же Delphi, в котором Object Pascal заменён на C++.

Любой язык программирования имеет свой состав, то есть набор символов и их сочетаний, которые можно использовать в программе на этом языке. Подобно обыкновенному человеческому языку, в языке программирования можно выделить основные элементы – лексемы, выражения и операторы. Все они записываются при помощи алфавита языка.

Дадим определение элементам языка:

- *лексема* – минимальная (неделимая) конструкция языка, которая имеет самостоятельный смысл, то есть её нельзя разбить на более мелкие значащие элементы;

- *выражение* – последовательность операндов и знаков операций, задающая правило вычисления некоторого выражения;

- *оператор* (инструкция) – последовательность выражений и лексем, используемая для обозначения какого-либо действия, производимого над объектами программы. В частности, различают простой и составной операторы.

При написании программ используются те символы, которые включены в алфавит языка, алфавит C++ включает в себя:

- прописные и строчные латинские буквы и знак подчеркивания;
- цифры от 0 до 9;
- специальные знаки:

“ { }, | [ ] ( ) + - / % * . \ ‘ : ? < = > ! & # ~ ; ^

- пробельные символы: пробел, символы табуляции, символы перехода на новую строку.

Из символов алфавита формируются лексемы языка: идентификаторы, ключевые слова, знаки операций, константы и разделители.

Вопросы для закрепления знаний:

1. Каково назначение алгоритмического языка C^{++}?
2. Что такое лексема?
3. Каковы основные элементы языка?
4. Что включает в себя алфавит C^{++}?

5.2.1. Структура программы на языке C++

Программа на C++ состоит из функций, описаний, и директив препроцессора.

Функция – участок кода, оформленный как блок, в котором производятся те или иные действия. Можно рассматривать функции, как некоторые «кирпичики», из которых строится тело программы. Каждая функция обладает, как и математический её аналог, неким значением, то есть величиной одного из допустимых типов, которую она возвращает, будучи вызванной из некоторого места в программе. Выполнение программы начинается с функции main, которая обязательно должна присутствовать в теле программы.

В простейшем случае функция задаётся так:

тип_возвращаемого_значения *имя (/параметры/)*

{

здесь следуют операторы, составляющие тело функции

}.

Функция main также соответствует этому определению, поэтому в простейшем случае программа может выглядеть так:

Int main ( ) **{**

return 0**;**

}.

Заголовок говорит о том, что функция main не принимает никаких параметров (так как список в круглых скобках отсутствует) и возвращает в

вызвавшую её программу значение типа **int**, то есть целое число. Оператор **return** указывает, какое именно значение возвращает функция, в данном случае ноль. Если вместо **int** написать в заголовке **void**, то программа будет выглядеть следующем образом:

```
Void main(void)  {
return;      }.
```

В скобках указан не пустой список, а void. Это совершенно аналогичный способ указать программе, что функция не принимает параметров. Так как возвращать какое-то значение она также не должна, в операторе return ничего не указано, функция просто что-то делает и передает управление вызвавшей ее программе. Функция типа void аналогична процедуре Паскаля.

Но вышеописанная программа совершенно ничего не делает. Попробуем добавить в нее некоторый работоспособный код.

```
# include <iostream.h>
int main()  {
int a, b, c;
a=3; // переменная a получает значение 3
b=2; // переменная b получает значение 2
  c= a + b; // c=6
cout<<”a * b  = “ <<c; // выводится надпись “a * b = 6”;
return 0;
}
```

Самая первая строка содержит директиву препроцессору. Перед тем как передать программу компилятору, исходный код обрабатывается препроцессором и все директивы обрабатываются соответственно их смыслу. Например директива # include <iostream.h> указывает препроцессору вставить в это место всё содержание заголовочного файла, указанного её параметром, в нашем случае это файл <iostream.h>,

содержащий ввод-вывод потоком. Именно его подключение позволяет использовать функции cin и cout.

Рассмотрим лексемы подробнее

Идентификатор – имя объекта в программе, состоящее из латинских букв, цифр и знака подчеркивания, идентификатор не может начинаться с цифры и содержать пробелы. Не допускается также соединение идентификатора с одним из ключевых слов языка.

Правильный идентификатор	**Неправильный идентификатор**
MyVar	myVar (пробел в имени)
Casel	case (совпадает с ключевым словом)
Numl	Lnum (начинается с цифры)

В C и C++ имеет значение регистр идентификатора, то есть идентификаторы, отличающиеся регистром букв будут восприниматься компилятором как совершенно разные.

Ключевое слово – это тоже идентификатор, но имеющий для компилятора особое значение, например, обозначающее одну из конструкций языка. Они предназначены только для использования в том контексте, в котором они определены.

Знак операции – это один или несколько символов, определяющих действие над своими операндами, например, знак операции «+» мы привыкли рассматривать как сложение или как признак положительного числа. Операции подразделяются на унарные, бинарные и тернарные по количеству своих операндов. Каждый знак, кроме [ ], ( ) и ?: является отдельной лексемой.

Константы – это величины, которые не могут изменять свое значение в программе. Они подразделяются на целые, вещественные, символьные и строковые константы.

Целые константы:

- Десятичные: последовательность десятичных цифр, которые НЕ начинается с нуля (сам нуль – исключение).

- Восьмеричные нуль, за которым следует восьмеричные цифры (от 0 до 7).

Обозначают собой восьмеричные числа.

- Шестнадцатеричные: 0x или 0X, за которым следуют шестнадцатеричные цифры (от 0 до 9 и буквы A, B, C, D, E, F);

Вещественные константы представляют собой дробные числа, или, употребляя компьютерную нотацию, числа с плавающей точкой (это название связано со способом представления таких чисел в компьютере). Вещественные константы могут быть представлены в двух видах: десятичный, например, 6.67, 3.14159 и т. д., и экспоненциальный. В экспоненциальном виде вещественное число делится на две части, мантиссу и порядок. Мантисса записывается слева от знака экспоненты (с или E), порядок – справа от него. Значение вещественной константы определяется как произведение мантиссы на указанную в порядке степень числа 10. Примеры вещественных констант в экспоненциальной форме записи: 0.3E3, 1.7E4. Заметьте, что везде для отделения дробной части используется *точка, а не запятая.*

Символьные константы – это просто символ, заключенный в одинарные кавычки, например, ‘c’. Символ ‘\’ (обратная косая черта, или бэкслэш), будучи употреблен в символьной константе (равно как и в строке) имеет специальное значение, например ‘\n’ обозначает перевод строки, а ‘\t’ – горизонтальную табуляцию. Также с помощью ‘\’ можно вывести на экран символ по его коду (\015, \0xA2). Если требуется указать сам символ ‘\’, он просто удваивается – ‘\’.

Строковые константы представляют собой набор символов в двойных кавычках “Васил\n” – содержит строку “Васил” и символ перехода на следующую строку.

5.2.2. Типы данных

Тип данных определяет их внутреннее представление, то есть количество памяти, которое им необходимо выделить и то, как собственно они будут в этой памяти представлены. Кроме того, тип данных задает множество значений, которые способны принимать его величины и операции, которые с ними допустимо выполнять.

В C/C++ различают *основные и составные типы данных*.

Встроенные типы данных

В C++ различают следующие *основные типы переменных:*

- **Int** (целый);
- **char** (символьный);
- **bool** (логический, введен в стандарт C++ позже и не поддерживается старыми компиляторами);
- **float** (вещественный);
- **double** (тоже вещественный, но с удвоенной точностью представления).

Первые три типа называют *целочисленными,* а последние два – *типами с плавающей точкой.* Ко всем типам можно применять так называемые спецификаторы, их насчитывается четыре:

- short (короткий);
- long (длинный);
- signed (знаковый);
- unsigned (беззнаковый).

Пример: unsignedlongint – длинное целое, могущее принимать лишь положительные значения.

Целый тип (int) предназначен для хранения целых чисел. Его размер, вообще говоря не определен в стандарте C++, он зависит от компьютера и используемого компилятора.

Для 16-битных процессоров тип int занимает 2 байта, для 32-битных – 4 байта. Спецификатор short указывает компилятору отвести под хранение переменной 2 байта, а спецификатор long – 4 байта. На 32-битных компьютерах типы int и longint эквивалентны.

Символьный тип (char). Как правило, под величину типа char отводится 1 байт. Этого достаточно, чтобы отобразить любой символ из таблицы ASCII, так как в ней все допустимые символы соответствуют номерам от 0 до 255.

Логический тип (bool). Величины этого типа могут принимать только 2 значения true (истина) или false (ложь). Значение false по внутреннему представлению равно нулю, любое другое значение интерпретируется как true.

Типы с плавающей точкой (float, double и long double). Типы данных с плавающей точкой хранятся в памяти иначе, чем целое. Их внутреннее представление состоит из двух частей – мантиссы и порядка. Обычно под величину типа float выделяется 4 байта, в которых один двоичный разряд отводится под знак мантиссы, 8 разрядов под порядок и 23 под мантиссу. Величины типа double (с удвоенной точностью) занимают в памяти 8 байт, под порядок отводится 11 разрядов, а под мантиссу – 52 разряда. Спецификатор long показывает, что требуется выделить под величину 10 байтов памяти.

Диапазоны значений простых типов данных (таблица)

Тип	***Диапазон значений***	***Размер в байтах***
Bool	true и false	1
Signed char	-128…127	1
Unsigned char	0…255	1
Signed short int	-32768…32767	2
Unsigned short int	0…65535	2
Signed long int	2147483648…2147483647	4
Unsigned long int	0…4 294 967 295	4
Float	3.4e-38…3.4e+38	4
Double	1.7e-308…1.7e+308	8
Long double	3.4e-4932…3.4e+4932	10

В С++ предусмотрена операция sizeof, возвращающая размер типа в байтах. Ее нужно использовать в случае необходимости узнать размер типа int, так как он зависит от архитектуры компьютера и используемой ОС. Выражение sizeof (int) вернет для MS-DOS 2, а для Win32 4.

Говоря о многообразии типов данных в С++, нужно упомянуть также тип void, или *пустой тип.* Void используется для указания того, что функция не возвращает в программу никакого значения.

Вопросы для закрепления знаний:

1. Какова структура программы на С++?
2. Что такое идентификатор? Какие требования ставятся к нему?
3. Как записывают операторы?
4. Расскажите о директивах препроцессору.
5. Укажите названия стандартных математических функций.
6. Диапазоны значений простых типов данных.
7. Перечислите простые типы данных.
8. Как объявляются типы переменных?
9. Перечислите спецификаторы типов.
10. Что означает тип void?

5.2.3. Переменные, операции, выражения в С++

Переменная – это область в памяти компьютера, на которую можно ссылаться по имени. Переменная имеет своё имя и значение. Её отличие от константы в том, что значение переменной можно изменить в ходе программы. Достаточно просто присвоить ей другое подходящее значение.

Тип, как уже говорилось ранее, характеризует объём памяти, выделяемой под переменную и диапазон ее значений. Так как при использовании переменной должен быть известен ее тип, каждая переменная должна быть *описана,* прежде чем хотя бы раз встретиться в тексте программы.

Частный случай синтаксиса описания был рассмотрен в разделе «Типы данных». В общем же виде оператор описания выглядит так:

[класс памяти] [const] тип имя [инициализатор];

Здесь и далее в квадратных скобках указываются элементы, которые могут быть опущены.

Инициализироваться переменная может двумя способами: первый – после знака равенства, например:

Int a = 10 ;

и второй - в круглых скобках после имени переменной:

int a (10) .

В одном операторе объявления могут объявляться сразу несколько переменных, тогда они указываются через запятую, причем каждая должна инициализироваться отдельно.

Int a = 3,b = 2 ,c (1).

Если заявленный тип переменной не совпадает с типом присваиваемого при инициализации значения, происходит преобразование типа.

Переменные в программе нужны для хранения данных. С помощью *операций* осуществляется необходимое программисту изменение переменных, обработка данных. Операции подразделяются на три категории по количеству операндов: унарные (один операнд), бинарные (два операнда) и тернарные (три операнда, в С и С++ всего одна тернарная операция).

В приведенной здесь таблице указаны все операции языка С++.

Операция	**Описание**
Унарные операции	
++	увеличение на 1
--	уменьшение на 1
sizeof	размер
~	поразрядное отрицание
!	логическое отрицание
-	арифметическое отрицание(унарный минус)
+	унарный плюс
&	взятие адреса

*	раз адресация
new	выделение памяти
delete	освобождение памяти
(type)	преобразование типа
Бинарные операции	
*	умножение
/	деление
%	остаток от деления
+	сложение
-	вычитание
<<	сдвиг влево
>>	сдвиг вправо
<	меньше
<=	меньше или равно
>	больше
>=	больше или равно
= =	равно
!=	не равно
&	поразрядная конъюнкция (И)
^	поразрядное исключающее ИЛИ
|	поразрядная дизъюнкция (ИЛИ)
&&	логическое И
||	логическое ИЛИ
=	присваивание
*=	умножение с присваиванием
/=	деление с присваиванием
%=	остаток от деления с присваиванием
+=	сложение с присваиванием
-=	вычитание с присваиванием
<<=	сдвиг влево с присваиванием
>>=	сдвиг вправо с присваиванием
&=	поразрядное И с присваиванием
|=	поразрядное ИЛИ с присваиванием
^=	поразрядное исключающее ИЛИ с присваиванием
.	последовательное вычисление
Тернарная операция	
?:	условная операция (тернарная)

Операций очень много, и некоторые поначалу могут показаться очень непонятными. Попытаемся пояснить некоторые на примерах.

Самая частая операция – присваивание. Хотя операций присваивания в С++ около десятка, цель у них одна – переменная слева от знака присваивания получает значение выражения справа от знака.

```
# include <iostream.h>
int main()  {
```

```
inta, b, c;
a=3; // переменная a получает значение 3
b=2; // переменная b получает значение 2
  c= a + b; // c=6
cout<<”a * b  = “ <<c; // выводится надпись “a * b = 6”;
return 0;
}.
```

В этой программе, кроме оператора присваивания, впервые были использованы комментарии. *Комментарий* – это последовательность любых символов, заключенная или в парные символы / * * / или следующая после // и до конца строки. Комментарии полностью игнорируются компилятором. Их основное предназначение – сопровождение текста программы пояснениями и временное исключение некоторых фрагментов (этот процесс называется «закомментировать»).

Использованная в приведенной выше программе операция – называется операцией простого присваивания. Она присваивает стоящей слева переменной значение справа от знака равенства. Остальные операции присваивания называются сложными: в их формировании участвуют и правая и левая величины.

Например, в результате выполнения оператора a + = 2, вначале значение переменной a будет увеличено на 2 и затем присвоено той же переменной a, что эквивалентно оператору a = a + 2, но записывается короче и выполняется быстрее.

Другие примеры сложного присваивания:

```
b *= 4;          // b = b *4;
c += 2*b;       // c = c + 2*b;
a %= 3;          // a = a % 3.
```

Наряду с присваиванием в предыдущих операторах используются и арифметические операции +, -, *, /, смысл которых аналогичен

математическому. Хотя следует учитывать, что для операндов целого типа операция / является целочисленным делением.

Операция % обозначает взятие остатка от деления двух величин, например 15 % 4 = 3. Смысл поразрядных и сдвиговых операций присваивания станет ясен после рассмотрения соответствующих им простых операций.

Важный момент: в C++ результатом операции присваивания является присвоенное значение, то есть возможны следующие операторы:

```
cout << a = b + c << "\n";
a = b = 2.
```

В первом случае будет произведено присваивание одной переменной суммы двух других и последующий вывод этой суммы, а во втором значение 2 сначала будет присвоено переменной b, а затем и переменной a. В этом случае необходимо учитывать *ассоциативность* операции, то есть направление её выполнения. У всех операций присваивания ассоциативности справа налево.

Математические функции

C++ унаследовал из C стандартные математические функции, описание которых находится в заголовочных файлах <math.h> (<cmath>). Они позволяют получить абсолютное значение (abs, fabs), округленное число (ceil, floor), квадратный корень(sqrt), степень (pow), значения тригонометрических функций (sin, cos, tan,sinh, cosh, tanh, asin, acos, atan, atan2), экспоненту (exp), логарифм (log, loglO), дробную и целую части числа (modf), остаток от деления (fmod) и другие.

Ошибки индицируются установкой еггпо из <errno.h> (<сеггпо>) в значение EDOMдля ошибок, связанных с выходом из области определения, и ERANGE для ошибоквыхода за пределы диапазона.

Выражения

Из констант, переменных, разделителей и знаков операций можно конструировать выражения.

Каждое выражение представляет собой правило вычисления нового значения. Если выражение формирует целое или вещественное число, то оно называется арифметическим. Пара арифметических выражений, объединенная операцией сравнения, называется отношением. Если отношение имеет ненулевое значение, то оно – истинно, иначе – ложно.

Приоритеты операций в выражениях

Ранг	**Операции**
1	( ) [ ] -> .
2	! ~ - ++ -- & * (тип) sizeof тип( )
3	* / % (мультипликативные бинарные)
	+ - (аддитивные бинарные)
5	<<>> (поразрядного сдвига)
6	<><= >= (отношения)
7	== != (отношения)
8	& (поразрядная конъюнкция «И»)
9	^ (поразрядное исключающее «ИЛИ»)
10	| (поразрядная дизъюнкция «ИЛИ»)
11	&& (конъюнкция «И»)
12	|| (дизъюнкция «ИЛИ»)
13	?: (условная операция)
14	= *= /= %= -= &= ^= |= <<= >>= (операция присваивания)
15	, (операция запятая)

Вопросы для закрепления знаний:

1. Как определяются переменные?
2. Перечислите унарные операции.
3. Перечислите бинарные операции.
4. Объявите целые переменные и присвойте им значения.
5. Объявите действительные переменные и присвойте им значения.
6. Укажите приоритеты различных операций.
7. Запишите выражение с использованием стандартных функций.

5.2.4. Управляющие конструкции языка C++

Операторы управления работой программы называют управляющими конструкциями программы. К ним относят:

- составные операторы;
- операторы выбора;
- операторы циклов;
- операторы перехода.

Операторы передачи управления (выбора) помогают перейти от выполняющегося в данный момент участка кода к другому, выйти из тела цикла до его завершения**,** или вернуть результат из функции.

Условный оператор if

Условный оператор if предназначен для ветвления процесса вычислений в программе. Блок операторов, следующий за конструкцией if выполняется лишь при истинности некоторого условия. Формат условного оператора:

if (выражение) оператор_ 1 ; [**else**оператор _ 2 ;]

В начале определяется значение *выражения,* оно может иметь арифметический тип, или тип указателя Если выражение равно нулю (то есть не равно true), выполняется первый оператор, а иначе – второй оператор. Часть else можно опустить, тогда, если условие не выполняется, управление переходит к оператору, следующему за первым.

К примеру, в зависимости от года рождения, хранящегося в переменной b__year, выведем сообщение, гласящее, достиг ли пользователь совершеннолетия (текущий год хранится в переменной year);

```
if ( year - b_yer> 18)
cout<< “Вы достигли совершеннолетия“;
else
  cout << “Вам осталось еще“ << 18 -- (y e  a r  --  b __y e a r )  <<
```

<< “ лет до совершеннолетия”.

Допустим, пользователь ввел 1985, а текущий год - 2002. Тогда результатом логического выражения будет false (ложь), и выполнится оператор cout, стоящий в ветви else. Если бы ее не было, управление было бы передано оператору следующему за первым cout.

В том случае, если при выполнении условия требуется выполнить не один, а несколько операторов, их необходимо заключить в фигурные скобки, то есть представить как блок. Введем в программу проверку на случай, если пользователь, достигший совершеннолетия должен ввести некоторый номер:

```
if(year - b_ year>18 ) {
cout<< “Вы достигли совершеннолетия” ;
cin>>num;
      }
else
cout<< “ Вам осталось еще “ << 18 – (year – b_year) <<
<< “ лет до совершеннолетия”;.
```

В качестве оператора_2 может быть указана другая управляющая структура, это позволяет проверять диапазоны:

```
if(x > 0)
{ . . . };
else if (x < 0)
{ . . . };
else
{ . . . }.
```

Структура if является гибким средством алгоритмизации, но она лишь проверяет условие и выполняет операторы в зависимости от результата проверки. В случае же многократного повторения некоторых операций целесообразно использовать циклы.

Оператор switch - переключатель

Оператор switch предназначен для разветвления процесса вычислений на несколько направлений. Его формат:

switch(выражение)

{

caseконстантное_ выражение_1 : [список_операторов_1];

caseконстантное_ выражение_2 : [список_операторов_2];

. . .;

caseконстантное_ выражение_n : [список_операторов_n];

[**default: опеаторы**];

}.

Работает этот оператор следующим образом: вначале вычисляется выражение (оно должно быть целочисленного типа), затем среди конструкций *case* ищется такая, константное выражение которой совпадает со значением выражения в условии. Если такое совпадение найдено, происходит выполнение соответствующего списка операторов. Чтобы организовать выход из структуры *switch* в этих списках используется оператор *break*, осуществляющий выход из самого внутреннего из объемлющих его операторов switch, for, while и do. Если не было найдено ни одного совпадения с константными выражениями, выполняется ветка *default,* которая может и отсутствовать. Для примера использования оператора *switch* в реальной программе, рассмотрим случай тестовой программы, когда за каждый из ответов на тестовый вопрос начисляется разное количество баллов. Допустим, есть четыре варианта ответа (a, b, c, d), введенный ответ содержится в переменной answer типа char, а текущее количество баллов в переменной total:

switch(answer) {

case‘a’, ’A’: total += 2; **break;**

case‘b’, ’B’: total += 1; **break;**

case‘c’, ’C’: total += 4; **break;**

case‘d’, ’D’: **break;** // 0 баллов, сумма не меняется

default: cout<< “Вы ввели неверный ответ!”;

}.

Обратите внимание: структура *switch* может проверять выражение на совпадение только с константами. Если необходимо проверить некоторый диапазон или значение переменной, приходится использовать *ifelseif.*

Вопросы для закрепления знаний:

1. Для чего предназначен оператор условного перехода?

2. Каковы формы записи оператора условного перехода?

3. Когда и как применяется составной оператор?

4. Может ли оператор условного перехода содержать вложенные операторы условного перехода?

5.Нарисуйте и объясните блок-схему выполнения оператора условного перехода.

6. Для чего предназначен оператор выбора (переключатель)?

7. Сколько операторов можно записать после возможного значения переменной?

8. Расскажите об основных логических операциях.

9. Расскажите об операциях отношения.

10. Каков порядок действий в логическом выражении?

Операторы цикла

Организация повторений: операторы цикла: с предусловием, с параметром. Алгоритм вычисления сумм и произведений, вычисление суммы ряда с заданной точностью, вычисление значений многочленов (схема Горнера).

Под оператором цикла понимается указание программе повторять некоторую последовательность операторов заданное количество раз или пока выполняется некоторое условие. Всего в С++ три вида циклов:

- с предусловием (while);

- с постусловием (dowhile);
- с параметром (for).

Любой цикл состоит из тела цикла, то есть тех операторов, которые выполняются несколько раз, начальных установок, модификации параметров цикла и проверки условия продолжения выполнения цикла.

Один проход цикла называется итерацией. Проверка условия выполняется на каждой итерации либо до тела цикла (цикл с предусловием), либо после тела цикла (цикл с постусловием). Отличие этих двух способов в том, что в цикл с предусловием, возможно, не выполнится ни разу, если условие с самого начала окажется ложным, а цикл с постусловием гарантировано выполнится хотя бы один раз.

Цикл завершается, если не выполнится условие его продолжения. При необходимости возможно принудительное завершение всего цикла оператором break или текущей итерации оператором continue.

Теперь рассмотрим синтаксис операторов цикла в C++.

Цикл while (с предусловием).

Формат:

- while(выражение) оператор.

Выполнение оператора начинается с проверки условия в скобках после while. Если условие истинно, выполняется оператор цикла, иначе управление передается оператору, следующему за циклом. Если при первой же проверке условие не выполняется (равно false), то цикл не выполнится ни разу.

Потенциальный источник ошибок, если в теле цикла условие не изменяется, но оно является истинным, цикл будет бесконечным. Это также можно использовать в своих целях, например, объявить цикл:

while (true) { . . . } ;

и организовать выход изнутри цикла с помощью break.

Например, составим программу, которая будет печатать таблицу значений функции

y = x^2 + 2x + 5 в диапазоне пользователем:

```
#include <stdio.h>
int main  ( )
{
floatxn, xk, dx;
printf(“ Введите диапазон и шаг изменения аргумента: “);
scanf(“%f%f%f”, &xn, &xk, &dx);
printf (“  X   |   Y   | \n”);
float x = xn;
while (x <= xk)
{
printf(“ %5.2f  | %5.2f  | \n”, x, x*x + 2 * x + 5);
x += dx;
}
return 0;
}.
```

В скобках после while можно объявлять переменную, ее областью действия в этом случае будет цикл.

Цикл do while (с постусловием).

Формат:

do оператор while выражение.

Цикл dowhile отличается от цикла while только тем, что в нем проверка условия происходит после цикла, то есть он достоверно выполнится хоть раз.

Порядок выполнения данного цикла следующий: вначале выполняется простой или составной оператор, являющейся телом цикла, затем проверяется выражение. Если оно не равно false, тело цикла выполняется еще раз. Цикл завершается, когда выражение становится равным false, или если происходит выход из цикла при помощи оператора передачи управления.

Одним из приложений цикла dowhile является ситуация, когда ввод какой-то величины должен продолжаться пока она не примет определенное значение, например:

```
do {
cout<< “Введите число от 1 до 100 (0 – выход) : “;
cin>>num;
. . . // действия счислом
}
while (num ! = 0);
```

Цикл for (с параметром).

Формат:

for(инициализация; выражение; модификации) оператор.

Если цикл for предназначен для многократного выполнения не одной инструкции, а программного блока, то его общий формат выглядит так

```
for( инициализация; выражение; инкремент)
{.
Последовательность инструкций:
}.
```

Цикл с параметром – один из мощнейших инструментов языка C++. Он позволяет не только повторять последовательность операторов заданное число раз, но и создавать сложные циклы с условиями выхода и прочими продвинутыми возможностями.

Схема работы цикла for следующая:

Выполняется инициализация. Она производится лишь однажды перед началом цикла. Обычно инициализация содержит установку переменной-счетчику начального значения.

Как и в цикле while, проверяется истинность выражения, и если оно равно true, выполняется тело цикла, иначе оно пропускается.

В заключение, производятся операции, описанные в секции модификаций, обычно это изменение управляющей переменной.

Рассмотрим на примерах, какие циклы можно создать, используя for:

for (int n = 1; n<= 10; n++) { . . . }.

Это наиболее очевидное применение for – переменная n принимает все значения от 1 до 10, то есть тело цикла выполнится 10 раз (если n не меняется в теле цикла):

for (int i = 1, j = 100; i != j; i ++, j --) {. . .}.

Это более сложный пример условия. Здесь в секции инициализации начальные значения присваиваются двум переменным, после каждой итерации одна из них увеличивается на единицу, а вторая уменьшается на единицу. Вывод из цикла происходит, когда значение обеих переменных равны, то есть всего будет 100/2 = 50 итераций:

for (int k = 1; ; k++) { . . . }.

В данном случае опущено условие выхода из цикла – на его месте стоит пустой оператор, состоящий только из знака; (точка с запятой). Значит, тело цикла будет выполняться, пока внутри не встретится оператор передачи управления, но в конце каждой итерации значение переменной-счетчика k увеличивается на единицу:

for (int i = 1; i <= 10; a[i] = i, i ++).

В этом, последнем примере, рассматривается случай, когда необходимо проинициализировать массив из десяти элементов некоторыми значениями (в данном случае – порядковыми номерами элементов, о массивах см ниже). Этот цикл не имеет даже тела – все необходимые операции выполняются уже в заголовке, в секции модификаций. После очередной итерации сначала происходит присваивание значения очередному элементу массива, а затем собственно приращение счетчика.

В принципе, конструкцию for всегда можно заменить эквивалентной ей, но несколько более громоздкой while, равно как и dowhile без проблем заменяется на простой while. Но можно привести несколько рекомендаций, когда использовать тот или иной цикл.

Оператор dowhile удобен, когда цикл обязательно должен быть выполнен хотя бы раз (например, в нем есть ввод данных с последующей проверкой правильности ввода).

Оператор for идеален для организации цикла, управляемого счетчиком, но также удобен во многих других случаях.

В остальных циклах можно применять while, особенно если число итераций неизвестно заранее, очевидных параметров цикла не имеется или их удобно модифицировать в конце цикла.

Примеры использования цикла с параметром.

Уменьшение параметра:

```
for( n=10; n>0; n--)
{ оператор}.
```

Изменение шага корректировки:

```
for( n=2; n>60; n+=13)
{ оператор }.
```

Возможность проверять условие отличное от условия, которое налагается на число итераций:

```
for( num=1;num*num*num<216; num++)
{ оператор }.
```

Коррекция может осуществляться не только с помощью сложения или вычитания:

```
for( d=100.0; d<150.0;d*=1.1)
{ <тело цикла>};
for (x=1;y<=75;y=5*(x++)+10)
{ оператор }.
```

Вычисление значений многочленов (схема Горнера).

Многочлен вида

$P=a0 *x^{к}+ a1* x^{к-1} + a2 *x^{к-2} + ...+ак$

с помощью схемы Горнера можно представить в удобном для вычислений виде, а, именно, в виде:

P=(…((a0 * x +a1)* x +a2) * x + …+ак-1) * x +ак.

В качестве примера составим программу для вычисления значения многочлена:

$P=6,25 * x^5+1,83 * x^4 +2,74 * x^3 – 10,5 * x^2 -10 * x +3.$

Для различных значений x, вводимых с клавиатуры.

```
# include<iostream.h>
# include<conio.h>
void main()
{ clrscr();
const float a[6]={6.25,1.83,2.74,-10.5,-10,3};//koefitsienti
int i=0 ;//indekskoefits
floatx,p=1.0;
cout<<"vvedi x";cin>>x;
while(i<6)
{
 p=p*x+a[i]; i++;
}
cout<< "\n mnogochlenpri x= "<<x<<" raven="<<p<<endl;
cout<<"enter ";
getch();
}.
```

Вопросы для закрепления знаний:

1. Как записывается и в каком случае используется:

 а) оператор цикла с параметром;

 б) оператор цикла с предусловием;

 в) оператор цикла с постусловием.

2. Как в алгоритмах представляются циклы? Приведите примеры.
3. Что такое цикл в программировании?
4. Могут ли быть циклы вложенными?
5. Какова структура функций в С++?

6. Как происходит обращение к функции в программе?

7.Как сформировать последовательность натуральных чисел случайным образом?

8. Напишите программу нахождения наибольших, наименьших значений последовательности.

Функции, объявление. Передача параметров в функции

Основной принцип структурного программирования – *составить алгоритм самым рациональным способом.* Для этого необходимо использовать наиболее подходящие управляющие структуры и создавать функции. Функцией называют участок кода, оформленный особым образом. Каждая функция имеет имя и обладает списком параметров. Внутри программы функция вызывается по своему имени с указанием параметров в скобках. При вызове функции выполняется заданная ею последовательность операторов с подстановкой параметров. Такой подход подобен построению программы из своеобразных «кирпичиков», что значительно облегчает последующее редактирование и повторное использование кода.

Использование функции дает возможность разбиению программы на завершенные по смыслу части, каждая из которых решает более мелкую задачу, чем программа в целом. Затем эти части (модули) объединяются и вместе используются для решения основной задачи. Такой подход является обязательным требованием к структурному программированию в целом.

Выполнение любой программы в C++ начинается с функции main.

Еще до функции main может быть объявлен прототип функции. Этот прототип необходим, чтобы компилятор, встретив вызов функции внутри тела программы, смог оценить ее параметры и выдать ошибку, если указано неверное число параметров, или не соблюдены их типы. Само описание функции должно находиться после функции main. Хотя при желании

можно вообще не употреблять прототип, а описать функцию прямо при первом ее объявлении.

Функция может не возвращать никакого значения. Зачастую это необходимо, когда эта функция должна лишь выполнить определенные действия, но не имеет числового эквивалента, который используется в программе. Тогда эту функцию можно описать как возвращающую тип void

В следующем примере объявлена функция, которая принимает два параметра. Затем она печатает свой первый аргумент количество раз, заданное вторым аргументом.

```
# include  <iostream . h>
void print(char,    int);   // прототипфункции
int main()
{
charc;
intn;
cout  «  "Введите  любой  символ:  ";
cin  »  c;
cout  «  "Сколько  ступенек?”; cin  » n;
for  (int  i  =  1;  i  <=  n;   i++)
{
print(c,i );
cout  «  "\n";
}
Return 0;
}
void print(char  c,   int n)
{
for  (int  i -  1;  i  <= n;  i++)
cout  «  c;
return;
```

}.

Для выхода из функции, имеющей тип void, используется оператор return без параметров. В программе функция типа void не может быть ничему присвоена, поэтому обычно вызывается в отдельной строке с указанием параметров, но не в составе выражения.

Все величины, описанные внутри функции, а также ее параметры, являются локальными. Областью их действия является функция. При вызове функции, равно как и при входе в любой блок, в специальной области памяти, называемой стеком, выделяется память под переменные этой функции (блока). При выходе из функции эта память очищается, то есть переменные перестают существовать.

Впрочем, если локальная переменная описана с модификатором static, ее значение сохраняется при выходе из функции.

Пример

```
Void func ( )
{
int a = 0;
static int n =0;
a++;
n++;
return ;
}.
```

В теле этой функции модифицируются две переменные a и n, но только n объявлена с модификатором static. Это означает, что при повторном вызове функции значении, а будет снова равно нулю, а значение n останется таким, каким оно было при предыдущем выходе из функции (1 после первого выхода, 2 - после второго и т. д.).

Глобальные переменные видны во всех функциях, кроме тех, в которых существуют локальные переменные с тем же именем. В таких случаях обратиться к глобальной переменной все же можно, указав,

спецификатор доступа :: (два двоеточия). Стоит избегать неоправданного использования глобальных переменных, так как они могут нарушить модульный принцип программы.

Передача по ссылке и по значению

При вызове функции создается временная копия переменных - аргументов функции. Любые изменения в теле функции не коснутся переменных программы. Но зачастую необходимо, чтобы функция изменяла переданные в нее переменные. Такое случается, когда одна функция должна вернуть в программу несколько значений. Тогда одного возвращаемого значения мало, и необходимо использовать передачу аргументов по ссылке.

В программе это можно записать так:

-void f(int& a, int& b, int&c).

Или эквивалентно:

-void f(int&a, int&b, int&c).

Запись со знаком & (амперсанд) означает, что в функцию передается не копия параметра, а сам параметр, при этом любые изменения, которым он подвергается в функции, будут отражаться на его значении в вызывающей программе.

Пример:

```
#include <iostream.h>
void f(int&, int&, int&);
int main(){
int a = 1, b = 2, c = 3;
cout « "a = " « a « " b = " « b « " c = "
« c « "\n";
cout « "Вызов функции f!" « “\n"; f(a,b,c);
cout « "a – “ « a « “ b = " « b « " c = " « c;
return 0;
}
```

```
void f(int& a, int& b, int& c) {
a *= 2;
b *= 2;
c *= 2;
return;
}.
```

В результате будет выведено две строки. В первой будет 1, 2, 3, а во второй 1, 4, 6, так как функция изменила значения переменных.

Итерационные циклы (Примеры)

Для итерационного цикла должно быть известно условие выполнения цикла.

Задача:

Дана последовательность целых чисел, за которой следует 0. Найти минимальный элемент этой последовательности.

```
#include <iostream.h>
#include <math.h>
void main()
{
inta,min;
cout<<"\nEnter a";
cin>>a;
min=a;
while(a!=0) || for(;a!=0;)
{
cout<<"\nEnter a";
cin>>a;                          // ввод  последовательности
if (a!=0 && a<min)  min=a;  // определение минимального
}
cout<<"\n min="<<min<<"\n";
}.
```

Задача : Найти сумму чисел Фибоначчи, меньших заданного числа Q.

```
# include<iostream.h>
void main()
{
int a=1,b=1, s=2, Q, c;
cout<<"\n Enter Q";
cin>>Q;
if(Q<=0)cout<<"Error in Q";
else
if (Q==1)    // формирование начальных значений
cout<<"\n S=1";
else
{
c=a+b;
while(c<Q) || for(;c!=0;)
{
s+=c;      // формирование суммы
a=b;
b=c;
c=a+b;   // формирование очередных чисел Фибоначчи
}
cout<<"\nS="<<s<<"\n";
}
}.
```

Вложенные циклы

Задача: Напечатать N простых чисел.

```
#include<iostream.h>
void main()
{
int a=1,n,d;
```

```
cout<<"\nEnter N";
cin>>n;
for(int i=0;i<n;) //внешний цикл
{
a++;d=1;
do                  //внутренний цикл
{
d++;
}
while(a%d!=0) ;//конец внутреннего цикла
if(a==d){
cout<<a<<" ";
i++;}
}                   //конец внешнего цикла
}.
```

Вопросы для закрепления знаний:

1. Какова структура функций в С++?

2. Как происходит обращение к функции в программе?

3. Каково назначение оператора return, как определяется тип возвращаемого значения?

4. Напишите фрагмент программы, используя меню, осуществляющей обращение к одной из нескольких функций.

5. Что такое прототип (прообраз, сигнатура) функции, его назначение?

6. Как сформировать последовательность натуральных чисел случайным образом?

ГЛАВА VI. КОМПЬЮТЕРНАЯ ГРАФИКА, ВИДЫ КОМПЬЮТЕРНОЙ ГРАФИКИ

Одно из самых популярных направлений использования компьютеров – это работа с компьютерной графикой. Компьютерная графика появилась достаточно давно, уже в 1960-х годах существовали полноценные программы работы с графикой. Сегодня принято пользоваться терминами «компьютерная графика» и «компьютерная анимация» «Computer graphics»- ввод-вывод, отображение, преобразование и редактирование графических объектов под управлением компьютера «Computer animation» – «оживление» изображений на экране дисплея, синтез динамических изображений на компьютере.

Понятие «компьютерная графика» включая все виды работ со статическими, векторными, растровыми изображениями, «компьютерная анимация» имеет дело с динамически, изменяющимися двумерными и трехмерными изображениями. На любом предприятии время от времени возникает необходимость в подаче рекламных объявлений в газеты и журналы или просто выпуске рекламной листовки или буклета. Крупные фирмы заказывают такую работу в специальном дизайнерским бюро или рекламным агентствам. Малые предприятия, имеющие ограниченный бюджет обходятся собственными силами и доступными программными средствами. В связи с развитием Интернета стала ощутима необходимость широкого графических программных средств. Каждая страница в Интернете оформлена с помощью компьютерной графики. Страница, оформленная без компьютерной графики, не имеет шансов выделяться на фоне широчайшего круга конкурентов и привлечь к себе массовое внимание. Несмотря на то, что для работы с компьютерной графикой существует множество классов программного обеспечения. Различают всего три вида компьютерной графики.

Это:

1. Растровая графика.
2. Векторная графика.
3. Фрактальная графика.

Они отличаются принципами формирования изображения при отображении на экране монитора или при печати на бумаге.

Растровую графику применяют при разработке электронных (мультимедийных) и полиграфических изданий. Иллюстрации, выполненные средствами растровой графики, редко создают вручную с помощью компьютерных программ. Чаще для этой цели используют, сканируют иллюстрации, подготовленные художником на бумаге или фотографии.

В последнее время для ввода растровых изображений в компьютер нашли широкое применение цифровые фото- и видеокамеры. Соответственно, большинство графических редакторов, предназначенных для работы с растровыми иллюстрациями, ориентированы не столько на создание изображений, сколько на их обработку.

Основным элементом растрового изображения является точка. Если изображения экранное, то эта точка называется *пикселем.* В зависимости от того, на какое графическое разрешение экрана настроена операционная система компьютера, на экране могут размещаться изображения, имеющие 640/480 , 800/600 , 1024/768 и более пикселей.

С размером изображения непосредственно связано и его *разрешение*. Этот параметр измеряется в точках на дюйм (dots per inch - dpi). У монитора с диагональю 15 дюймов размер изображения на экране составляет примерно 28/21 см. Зная, что дюйме 25,4 мм можно рассчитать, что при работе монитора в режиме 800/600 пикселов разрешение экранного изображения равно 72 dpi. При печати разрешение должно быть намного выше.

Полиграфическая печать полноценного изображения требует разрешения 200-300dpi. Чем больше количество пикселей в изображении, тем лучше его разрешение на экране и на печати. Основными недостатками этой графики являются большие размеры файлов и невозможность масштабирования изображений без изменения данных.

Векторная графика. Программные средства для работы с векторной графикой предназначены в первую очередь для создания иллюстраций и в меньшей степени для их обработки. Такие средства широко используют в рекламных агентствах, дизайнерских бюро, редакциях и издательствах. Оформительские работы, основанные на применении шрифтов и простейших графических элементов, решаются средствами векторной графики намного проще. Основным элементом изображения векторной графики является линия (при этом неважно прямая это линия или кривая).

В растровой графике тоже существует линия, но там они рассматриваются как комбинации точек. Соответственно, чем длинее растровая линия, тем больше памяти она занимает. В векторной графики объём памяти, занимаемый линией, не зависит от размеров линии, поскольку линия представляется в виде формулы, а точнее говоря, в виде нескольких параметров. Что бы мы ни делали с этой линией, меняются только её параметры, хранящиеся в ячейках памяти. Количество же ячеек остаётся неизменным для любой линии.

Линия - это элементарный объект векторной графики. Всё, что есть в векторной иллюстрации, состоит из линий. Простейшие объекты объединяются в более сложные, например, объект четырехугольник можно рассматривать как четыре связанные линии, а куб – 12 связанных линий, либо как шесть связанные четырехугольников. Из за такого подхода векторную графику часто называют объектно-ориентированной графикой.

На практике средства векторной графики используют не для создания художественных композиции, а для оформительских, чертежных и проектно-конструкторских работ.

Легко решаются вопросы масштабирования. Недостатком является сложное создание художественных изображений.

Фрактальная графика. Программные средства работы с фрактальной графикой предназначены для автоматической генерации изображений путём математических расчётов. Создание фрактальной художественной композиции состоит не в рисовании или оформлении, а в программировании.

Фрактальную графику редко применяют для создания печатных или электронных документов, но её часто используют в развлекательных программах. Фрактальная графика, как и векторная – вычисляемая, но отличается от неё тем, что никакие объекты в памяти компьютера не хранятся. Изображение строится по уравнению, поэтому, ничего, кроме формулы хранить не надо. Изменив коэффициенты в уравнении, можно получить совершенно другую картину. Способность фрактальной графики моделировать образы живой природы вычислительным путём.

6.1. Основные понятия компьютерной графики

Разрешение изображения и его размер

Основными параметрами компьютерного изображения является его *физический размер и разрешение*. От них зависит экранные размеры изображения и размеры отпечатка на бумаге, а также качество изображения.

Разрешения бывают разными: разрешение экрана, печати и изображения. Все эти понятия относятся к разным объектам и не связанными между собой пока не потребуется, какой физический размер имеет картина на экране, отпечаток на бумаге, файл на жёстком диске.

1. Разрешение экрана – это свойство компьютерной системы и операционной системы. Разрешение экрана измеряется в пикселях и определяет размер изображения, которое может поместиться на экране целиком..

2. Разрешение принтера – это свойство принтера, выражающее количество отдельных точек, которые могут быть напечатаны на участке единичной длины. Оно измеряется в единицах dpi (точки на дюйм) и определяет размер изображения при заданном качестве или, наоборот, качество изображения при заданном размере.

3. Разрешение изображения – это свойство самого изображения. Оно измеряется в точках на дюйм и задаётся при создании изображения в графическом редакторе или с помощью сканера. Разрешение изображения связано с физическим размером изображения. Оно измеряется как в пикселях, так и в единицах длины (мм, см, дюйм). Он задается при создании изображения и хранится вместе с файлом.

Цветовое разрешение и цветовые модели.

При работе с цветом используется понятие *цветовое разрешение* (глубина цвета) и *цветовая модель*. Цветовое разрешение определяет метод кодирования цветовой информации, и от него зависит то, сколько цветов на экране может отображаться одновременно. Н-р: выделение одного байта позволяет закодировать 256 различных цветовых оттенков;

Два байта (16 битов) позволяет определить 65536 различных цветов. Этот режим называется High Color. Если для кодирования цвета используется 3байта (24 бита), возможно, одновременное отображение – 16,5млн. цветов. Этот режим называется True Color.

Значит основные режимы:8-разрядный (256 цветов), 16-разрядный (65 тысяч цветов) и 24-разрядный (16,5 млн. цветов).

Большинство цветовых оттенков образуется смешанием основных цветов. Способ разделения цветового оттенка на составляющие компоненты называется *цветовой моделью*.

Существует много различных типов цветовых моделей, но в компьютерной графике, как правило, применяется не более трёх. Эти модели известны под названиями:

1) RGB; 2) CMYK; 3) HSB.

1) Цветовая модель RGB – в этой модели работают мониторы и бытовые телевизоры. Любой цвет считается составляющим из трех основных компонентов – красного (red), зелёного (green), синего (blue).

Эти цвета называются основными. Совмещение трёх компонентов даёт нейтральный цвет (серый), который при большой яркости стремится к белому цвету. Это соответствует тому, что мы наблюдаем на экране монитора, поэтому данную модель применяют всегда, когда готовится изображение, предназначенное для воспроизведения на экране.

Эта модель RGB является аддитивной. Метод получение нового оттенка суммированием яркостей составляющих компонентов называют аддитивным методом. Считается также, что при наложении одного компонента на другой, яркость суммарного цвета увеличивается.

2) Цветовая модель CMYK . Эту модель используют для подготовки печатных изображений, они отличаются тем, что их видят не в проходящем, а в отраженном свете. Чем больше краски положено на бумагу, тем больше света она поглощает и меньше отражает. Совмещение трёх основных красок поглощает почти весь падающий свет, и со стороны изображение выглядит почти черным. Увеличение количество краски приводит не к увеличению визуальной яркости, а, наоборот, к её уменьшению. Поэтому для подготовки печатных изображений используется не аддитивная (суммирующая) модель, а *субтрактивная* (вычитающая) модель. Цветовые компонентами этой модели является не основные цвета, а те которые получаются в результате вычитания основных цветов из белого:

- Голубой (Cyan)=Белый – Красный =Зелёный + Синий;
- Пурпурный(Magenta)=Белый – Зелёный = Красный +Синий;
- Жёлтый (Yellow)= Белый – Синий = Красный + Зелёный.

Эти три цвета называются дополнительными, т.к. дополняют основные цвета до белого.

Существенную трудность в полиграфии представляет чёрный цвет. Теоретически его можно получить совмещением трёх основных и

дополнительных красок, но на практике результат оказывается негодным. Поэтому в цветовую модель CMYK добавлен четвертый компонент – черный (black).

Перед печатью на компьютере, готовые изображения разделяют на 4 составляющих одноцветных изображений. Этот процесс называется цветоотделением.

3) Цветовая модель HSB – удобна для применения в тех графических редакторах, которые ориентированны не на обработку готовых изображений, а на их создание собственноручно. Если модель RGB наиболее удобна для компьютера, а модель CMYK- для типографий, то модель HSB наиболее удобна на человека. Она проста и интуитивно понятна.

В модели HSB тоже три компонента:

Оттенок цвета (Hue)

Насыщенность цвета (Saturation)

Яркость цвета (Brightness).

Регулируя эти три компонента, можно получить столь же много произвольных цветов, как и при работе с другими моделями. Создавая собственные художественные произведения, удобно работать в модели HSB, а по окончании работы его можно преобразовать в модель RGB или CMYK.

Создавая и обрабатывая цвет изображения на компьютере, принято использовать модель RGB, а при выполнении цветоотделения, печати на принтер рисунок преобразовывают в модель CMYK.

Цветовая палитра – это таблица данных, в которой хранится информация о том, каким кодом закодирован тот или иной цвет, эта таблица создается и хранится вместе с графическим файлом.

Индексная палитра – в ней записаны данные о том, к какому индексу цвета (из 256), какой реальный цвет соответствует.

Безопасная палитра – используется в web графике. Термин “Безопасная” связан с тем, что иллюстрации, созданные в такой палитре, могут воспроизводиться без искажений цвета на любой модели компьютера, подключённого к сети.

Вопросы для закрепления знаний:

1. Дать понятия компьютерной графике?
2. Перечислите виды компьютерной графики?
3. Что такое растровая графика?
4. Что такое векторная графика?
5. Что такое фрактальная графика?
6. Дайте основные понятия компьютерной графики?
7. Форматы файлов растровой графики.
8. Цветовые разрешения и цветовые модели.
9. Цветовая модель RGB.
10. Цветовая модель CMYK.
11. Цветовая модель HSB.
12. Дайте определение понятиям: цветовая палитра, индексная палитра, безопасная палитра.

6.2. Средства работы с растровой графикой. Программа ADOBE PHOTOSHOP

Средства создания изображения.

Программы, предназначенные для работы с растровыми изображениями, называют растровыми графическими редакторами. С помощью этих программ создают изображения, выполняют их ретушь и монтаж художественных композиций. Существует множество программ, предназначенных для работы с растровой графикой. Это, например,Fauve Matisse,Paint, ориентированы непосредственно на процесс рисования.

Средства обработки изображения. Есть и другие классы растровых графических редакторов, предназначенные не для создания изображений «с нуля», а для обработки готовых рисунков с целью улучшения их качества и реализации творческих идей. К таким программам, в частности, относятся: Photoshop, Photostyler, Picture Publisher и др.

Исходный материал для обработки на компьютере может быть получен разными путями:

Сканированием цветной иллюстрации загрузкой изображения, созданного в другом редакторе, или вводом изображения от цифровой фото или видеокамеры.

Основа будущего рисунка или его отдельные элементы могут быть созданы и в векторном графическом редакторе, после чего их экспортируют в растровом формате. Есть и новая услуга фото-CD, где запись производится по просьбе клиента фото-снимков.

Средства каталогизации изображения. Программы-катализаторы позволяют просматривать графические файлы множества разных форматов, создавая на HDD удобные альбомы, перемещать и переименовывать файлы, документировать и комментировать иллюстрации. Например: ACDSee 32, Imaging и др.

Форматы файлов растровой графики.

Файлы растровых изображений отличаются многообразием форматов (несколько десятков). У каждого формата есть свои положительные качества. Для операционной системы Windows 9x является **формат Windows Bitmap**. Файлы этого формата имеют расширение .bmp. главное качество этого формата универсальность. Недостатком является большой размер файлов из-за отсутствия сжатия изображения. Для WEB-документов (файлы, Используемые в Интернете), циркулирующих в сети Интернет, очень важен размер файлов, т.к. от него зависит скорость доступа к информации. Поэтому используется два вида графических форматов, обеспечивающих наиболее плотное сжатие.

1) Для хранения мгновенных нерегулярных изображений (фотографий) используют **формат JPEG**, имеющий расширение .jpg. Этот формат отличается тем, что обеспечивает хранение данных с огромной степенью сжатия, но за счёт потери части информации.

2) **Формат GIF** – это самый «плотный» из графических форматов, не имеющих потери информации. Файлы этого формата имеют расширение .gif. В этом формате хранятся и передаются малоцветные изображения, например, рисованные иллюстрации.

Особенностью этого формата является позволяющие создавать необычные эффекты: прозрачность фона и анимация изображения.

Есть и специальный **формат TIFF**, использующийся в полиграфии – обеспечивает не только неплохую степень сжатия, но и возможность сохранять в одном файле дополнительную информацию в невидимых вспомогательных слоях – каналах. Файлы этого формата имеют расширение .tif.

Графический редактор Adobe Photoshop.

Особой популярности среди пользователей является программа Photoshop компании Adobe. Кроме Photoshop компания Adobe предлагает сложные программы для создания проектов с использованием богатой графики, текста, и видео: Например, Acrobat , Frame Maker, Illustrator, Page Maker,PostScript и т.д.

Первоначальный вариант программы Photoshop, разработанный братьями Knoll, был приобретен компанией Adobe у фирмы Barney Scan в 1988 году. После значительного усовершенствования, в 1989 году был выпущен первый коммерческий вариант Adobe Photoshop. За прошедшие годы программа стала мировым стандартом в области обработки растровой графики, особенно с появлением версии Photoshop 4.0, выпущенной в ноябре 1996 года. Сейчас применяется версия 8.0

Программа Photoshop предназначена в первую очередь не для создания, а для обработки иллюстраций.

Стандартный интерфейс, позволяет работать с этой программой любому пользователю. В ней существует ряд графических фильтров, которые позволяют накладывать на изображение различные заготовки. Основным способом обработки изображений служит способ наложения слоёв «друг на друга». Этот способ значительно уменьшает работу пользователя.

Преимущество каждой версии Adobe Photoshop:

1) Третья версия – появилась поддержка слоёв.

2) Пятая версия – многоуровневая отмена неудачных действий и очень полезный инструмент Magnetic Lasso.

3) Версия 5.5 – включена программа Image Ready для работы с WEB-графикой.

4) Седьмая версия – разработчики устроили целый ряд недостатков:

- при обработке больших изображений высокого разрешения, снижен риск «зависания» системы;

- с этой программой при запуске одновременно работают и другие приложения;

- расширены возможности известных пакетов и добавлены новые;

- включены фильтры импорта и экспорта изображений в формате WBMP, который используется в карманных компьютерах (PDA) и т. д.

Наиболее мощным средством для обработки готовых растровых изображений сегодня считается программа Adobe Photoshop.

Основное назначение редактора Adobe Photoshop состоит в ретуши готовых изображений, т.е. доведения изображения до полиграфического качества, и в монтаже композиций из отдельных фрагментов, взятых из отдельных изображений и в применении специальных эффектов, называемых фильтрами.

Основными техническими операциями при работе с изображениями является:

- изменение динамического диапазона (управления яркостью и контрастностью изображения);

- повышение чёткости изображения;

- цветовая коррекция (изменение яркости и контрастности в каналах красной, зеленой, и синей, составляющих цветов);

- отмывка (изменение яркости отдельных фрагментов);

- растушёвка (сглаживание перехода между границами отдельных фрагментов);

- обтравка («вырезание» отдельных фрагментов из общей композиции);

- набивка (восстановление утраченных элементов изображения путем копирования фрагментов с сохранившихся участков);

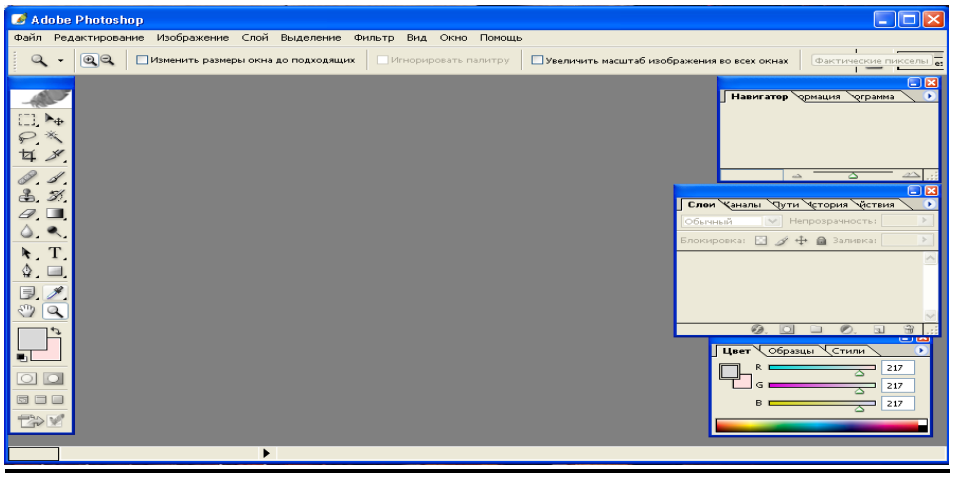
- монтаж (компоновка изображения из фрагментов, скопированных из других изображений или импортированных из других редакторов).

Интерфейс программы.

Основные элементы управления программы сосредоточены в строке меню и панели инструментов и кроме того используется особые диалоговые окна – инструментальные палитры.

Запуск и завершение программы осуществляется стандартным способом. После запуска программы на монитор выходит окно Adobe Photoshop.

<u>Основные элементы окна:</u>



1) Строка заголовков.

2) Строка меню (9 пунктов).

3) Панели инструментов: горизонтальные и вертикальные.

4) Рабочее поле.

5) Инструменты палитры.

6) Строка состояния.

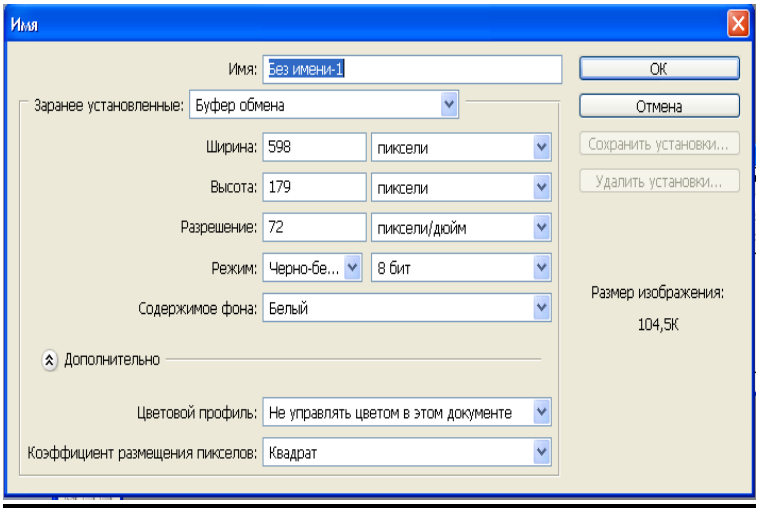
Для создания изображения требуется художественная способность пользователя, а обработка готовых изображений требуют знания технических приемов и обладания практическими навыками.

Поскольку графический редактор Photoshop предназначен в первую очередь не для создания, а обработки иллюстраций, работу с ним обычно начинают с загрузки

Для создания файла выбирают: Файл – Новый. Для открытия файла: Файл – Открыть или Импорт готового изображения.

Импортом называют ввод изображения, полученного сканером, цифровой фотокамерой или др. устройствами ввода.

При создании файла задаются параметры: имя, размер, ширина, высота, разрешение, режим, содержание, цвет, цвет фона, прозрачность. От всех этих параметров зависит размер файла.



6.3. Способы создания и редактирования готовых изображений

Инструменты редактора Adobe Photoshop.

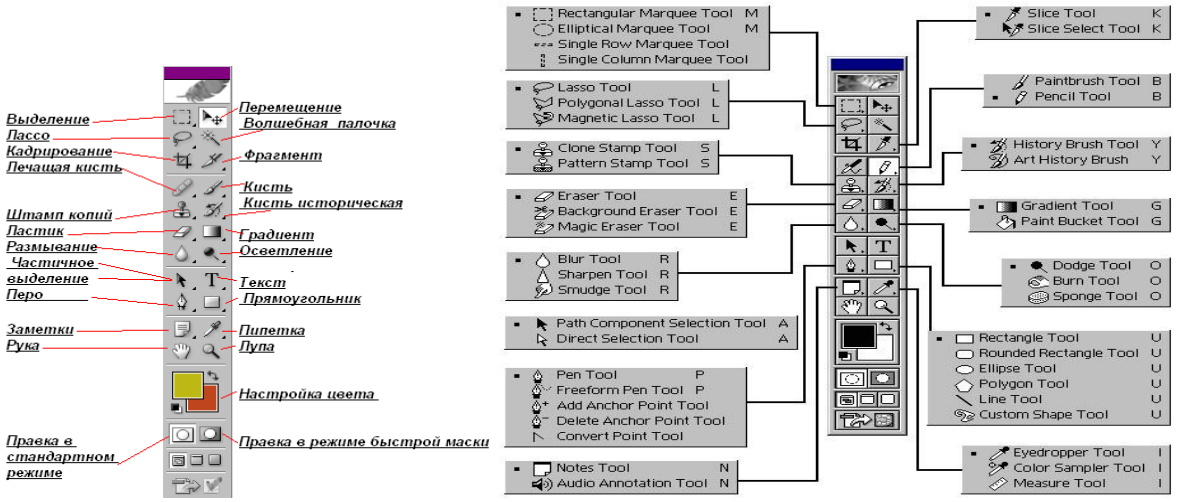
Панели инструментов является основным средством для работы с изображениями. Инструменты объединены в 4 группы. Их особенностью является наличие альтернативных инструментов, которые обозначены специальным треугольником. нажимая правую кнопку мыши откроется линейка с дополнительным инструментом.

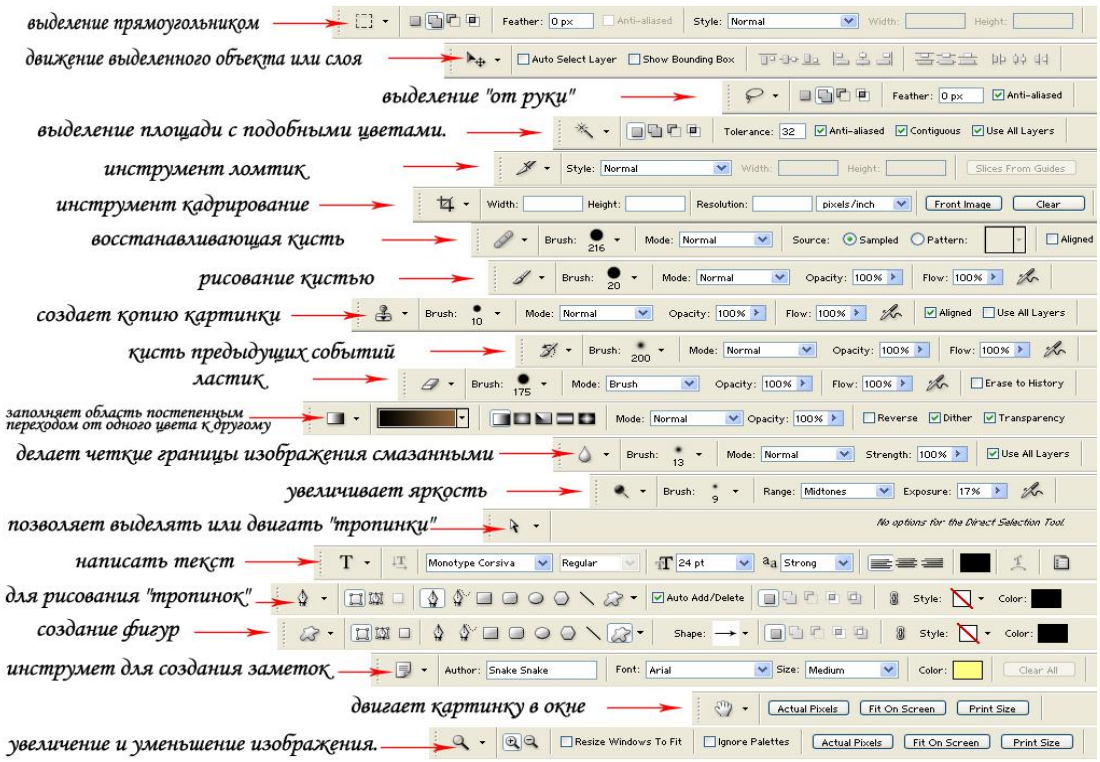
1) Первую группу значков составляют инструменты для работы с объектами. С помощью *выделения и лассо* можно выделять области изображения. *Перемещение* – перемещение выделенной области и их копия. *Волшебная палочка* – служит для автоматического выделения области по признаку цветового подобия. Эти два инструмента *Волшебную палочку и Лассо* применяют для выполнения операции «обтравки» - точной обводки сложных контуров графических объектов.

2) Группа инструментов предназначенных для рисования. *Штамп* применяют для операции набивки, с помощью которой удобно восстанавливать повреждённые участки рисунка, копируя небольшие части изображения с неповрежденных участков. *Палец* имитирует сдвиг сырой краски, и используется для операции размывки. *Размытие/резкость* – позволяют управлять резкостью отдельных участков. *Осветлитель/затемнитель/губка* служат для местной регулировки яркости и цветовой насыщенности. *Губка* – имитирует операцию отмывки. *Заливка, Градиент* – служит для заливки выделенных участков, одним из основных цветов или плавным переходом между цветами.

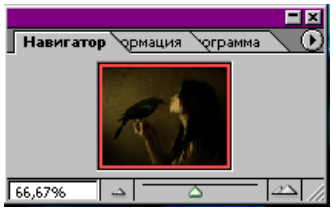
3) Группа инструментов предназначена для создания новых объектов, в том числе и текстовых. *Перо* – создание, редактирование плавных криволинейных контуров. Текст – надписи. Линия – рисование отрезков и прямых. Пипетка – точный выбор цвета.

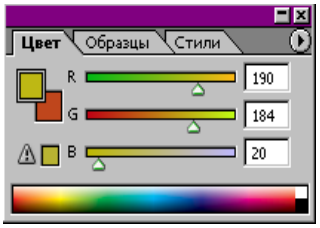
4) Группа инструментов – управление просмотром. *Масштаб* – работа с увеличенными фрагментами рисунка. *Рука* – перемещение рисунка, выходящего за пределы окна программы. *Заметка* – вставка комментариев.

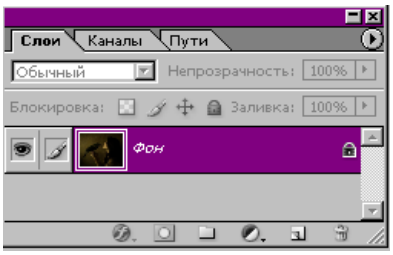


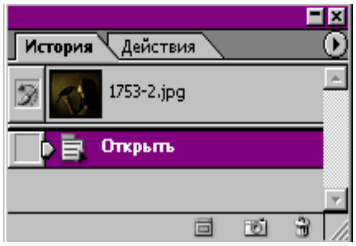


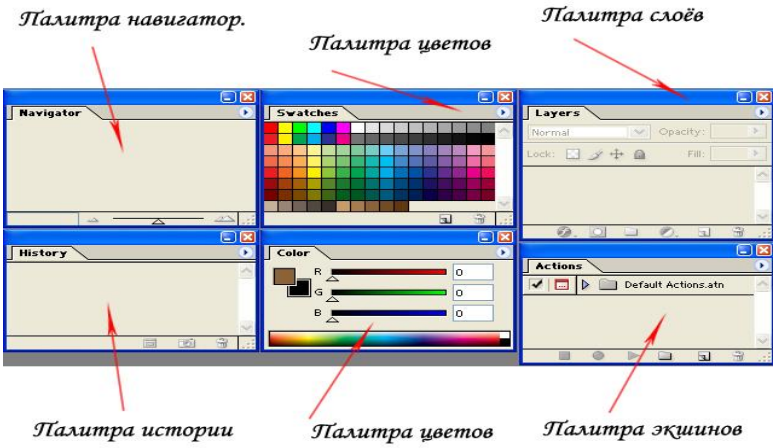
Инструментальные палитры











В графических программах, часто используются диалоговые окна особого вида. Они называются палитрами и имеют некоторые общие элементы управления. Палитры служат для настройки действия основных инструментов и для операций с изображением и его файлом.

Имеется 10 палитр. У каждой палитры своё назначение. Доступ к палитрам осуществляется с помощью меню пункта Окно.

1) Навигатор – позволяет быстро просмотреть фрагменты изображения и изменить масштаб просмотра

2) Инфо – отображает информацию о координатах курсора и цветовых параметрах текущей точки. В зависимости от выбранного инструмента позволяет определять размеры, расстояния, углы поворота.

3) Параметры – показывает названия и текущие настройки выбранного инструмента. С помощью элементов управления палитры можно изменять свойства инструмента.

4) Синтез – отображает цветовые значения текущих цветов переднего и заднего плана.

5) Каталог – содержит набор доступных цветов. Позволяет выбрать цвет переднего и заднего плана, добавить в набор новые и удалить ненужные цвета.

6) Кисти – содержит варианты кистей, используемых для рисования и редактирования. Характеристика кистей сохраняются для каждого инструмента по отдельности.

7) Слои – перечислены все слои изображения, начиная с верхнего. Палитру используют для определения параметров слоев, изменения их порядка и преобразования.

8) Каналы – используются для выделения, создания, дублирования и удаления каналов, определения их параметров, изменения порядка, преобразования каналов в самостоятельные документы и формирования совмещенных изображений из нескольких каналов.

9) Контуры – содержит список всех созданных контуров. Криволинейные контуры при преобразовании их в выделенную область могут использоваться для формирования обтравочных контуров.

10) Операции – можно создавать макрокоманды (последовательности действий, выполняемых с изображением). Их можно записывать, выполнять, редактировать, удалять и сохранять в виде файлов.

Применение эффекта Pattern Overlay

1. На палитре Layers (Слои) дважды щелкните по имени какого-либо слоя.
2. Щелкните по названию эффекта **Pattern Overlay** (Наложение узора).
3. Выполните любое из перечисленных ниже действий (рис. 1).

Выберите значение параметра **Blend Mode** (Режим смешивания).

Теперь отрегулируйте значение параметра **Opacity** (Непрозрачность).

Щелкните по стрелке в поле **Pattern** (Узор) и выберите желаемый образец на всплывающей панели.

Щелкните по кнопке **Snap to Origin** (Привязать к началу координат), чтобы выровнять узор относительно верхнего левого угла изображения. Изменить положение узора можно также, перетащив его в окне изображения.

Рис.6.1. Опции эффекта *Pattern Overlay*.

Рис.6.2. Результат применения эффекта *Pattern Overlay* к объекту слоя типа shape.

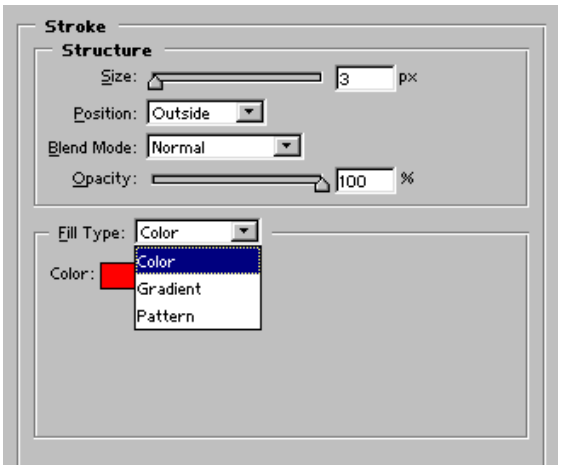


Рис.6.3. Опции эффекта *Stroke*.

Рис.6.4. Результат применения эффекта *Stroke* к объекту слоя фигуры.

Рис.6.5. Исходное изображение.

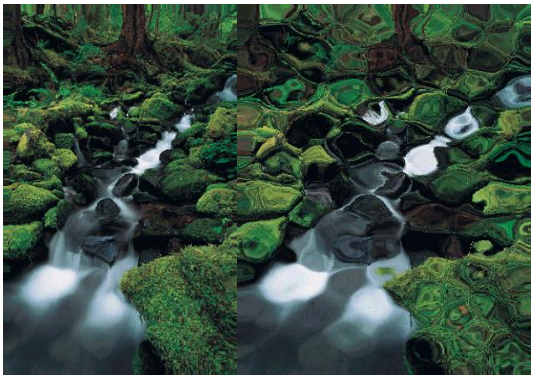


Рис.6.6. Результат заливки изображения дерева, к которому был применен фильтр *Glass* *Стекло),* пикселями предыдущего состояния.

Использование инструмента Magic Eraser.

Инструмент **Magic Eraser** (Волшебный ластик) позволяет стирать пикселы простым щелчком мыши, а не перетаскиванием курсора. Он стирает точки, цвет которых с учетом заданного допуска похож на цвет пиксела, по которому вы щелкнули. Этот инструмент работает так же, как и инструмент **Paint Bucket** (Ведро с краской), за исключением того, что он удаляет, а не добавляет пикселы. Если установить непрозрачность ниже 100%, то с помощью инструмента **Magic Eraser** можно сделать целевые области слоя частично прозрачными.

1. Из всплывающей палитры инструмента **Eraser** (Ластик) выберите инструмент **Magic Eraser** (Волшебный ластик). Активизировать этот инструмент можно также, щелкнув по пиктограмме; iPJ или нажав комбинацию клавиш **Shift+E.**

2. На панели (рис.6.3.7, 6.3.8) введите значение параметра **Tolerance** (Допуск). Чем оно выше, тем шире диапазон цветов, которые будут стерты. Введите маленькое значение допуска, если хотите стирать только цвета, практически идентичные цвету, по которому вы щелкнете. Введите 0, чтобы стереть пикселы только одного цвета.

Рис.6.7. Левая часть панели опций инструмента *Magic Eraser.*

Рис.6.8. Правая часть панели опций инструмента *Magic Eraser.*

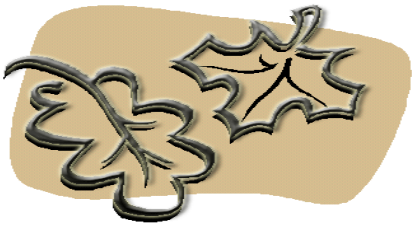


Рис.6.9. Исходное изображение.

Рис.6.10. Результат стирания правого листапри помощи инструмента.
Magic Eraser *с включенной опцией* ***Contiguous.***

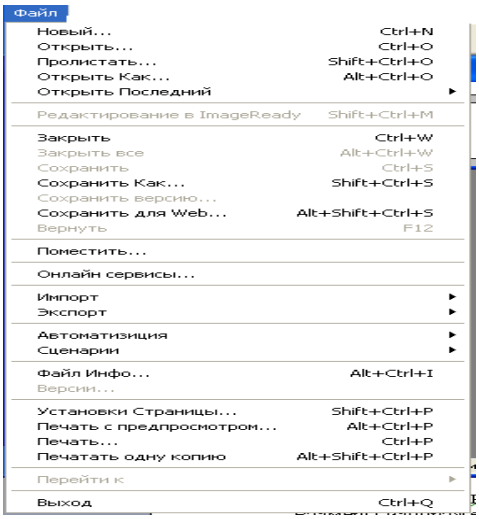
Рис.6.11. Результат стирания правого листа при помощи инструмента.
Magic Eraser *с включенной опцией* ***Contiguous.***

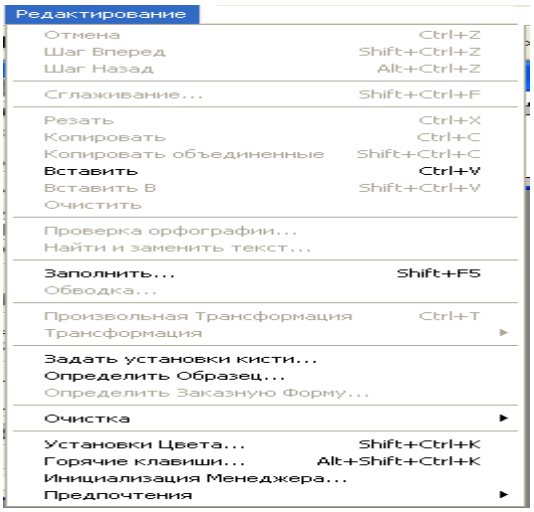
Вопросы для закрепления знаний:

1. Предназначение группы инструментов в программе Photoshop.
2. Инструментальные палитры редактора Photoshop.

6.4. Работа с фильтрами

В меню программы Photoshop 9 пунктов. Каждый пункт меню имеют своё подменю:





Изображение

- Режим ▸
- Регулировки ▸
- Дубликат...
- Применение Изображения...
- Вычисления...
- Размер Изображения...
- Размер Холста...
- Коэффициент размещения пикселов ▸
- Повернуть Холст ▸
- Кадрировать...
- Подрезать...
- Показать Все
- Треппинг...

Слой

- Новый ▸
- Дубликат Слоя...
- Удалить Слой ▸
- Свойства Слоя...
- Эффекты Слоя ▸
- Новая Заливка Слоя ▸
- Новая Установка Слоя ▸
- Изменить Содержание Слоя ▸
- Нстройки Содержания Слоя...
- Тип ▸
- Отрисовка ▸
- Новые Сектора Слоя
- Добавить Маску
- Показать Маску
- Добавить векторную маску
- Включить векторную маску
- Группировать с Предшествующим Ctrl+G
- Разгруппировать Shift+Ctrl+G
- Расположение ▸
- Выравнивание Связей ▸
- Распределение Связей ▸
- Закрыть Все Слои в Наборе...
- Объединить слои Ctrl+E
- Объединить с Видимым Shift+Ctrl+E
- Выполнить сведение
- Матирование ▸

Выделение

- Все Ctrl+A
- Убрать Выделение Ctrl+D
- Выделить Заново Shift+Ctrl+D
- Инвертировать Выделение Shift+Ctrl+I
- Цветовой диапазон...
- Растушевать... Alt+Ctrl+D
- Модифицировать ▸
- Смежные пикселы
- Подобные пикселы
- Преобразовать Выделение
- Загрузить Выделение...
- Сохранить Выделение...

Фильтр

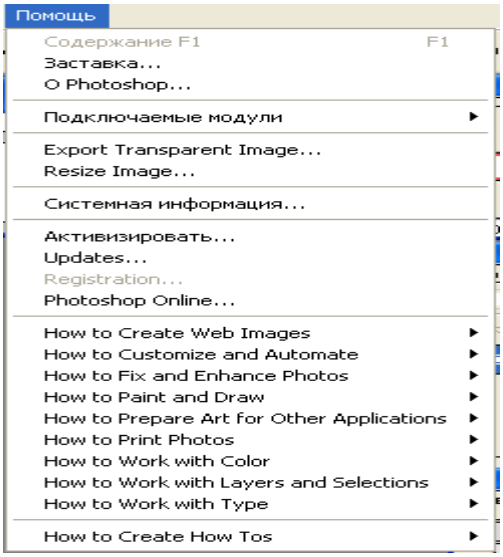
- Последний Фильтр Ctrl+F
- Вычитание... Alt+Ctrl+X
- Галерея фильтров...
- Сжижение... Shift+Ctrl+X
- Создатель узоров... Alt+Shift+Ctrl+X
- Render ▸
- Sketch ▸
- Video ▸
- Пикселизация ▸
- Резкость ▸
- Смазывание ▸
- Стилизация ▸
- Шум ▸
- Другие ▸
- Artistic ▸
- Blur ▸
- Brush Strokes ▸
- Digimarc ▸
- Distort ▸
- Other ▸
- Pixelate ▸
- Stylize ▸
- Texture ▸

Вид

- Установки Просмотра ▸
- Проверка цветов Ctrl+Y
- Предупреждения гаммы Shift+Ctrl+Y
- Подтверждение коэффициента размещения пикселов
- Увеличение Масштаба Ctrl++
- Увеличить Ctrl+-
- Показать во Весь Экран Ctrl+0
- Фактический Размер Alt+Ctrl+0
- Размер Печатного Оттиска
- Вид экрана ▸
- ✔ Дополнения Ctrl+H
- Показать ▸
- ✔ Направляющие Ctrl+R
- ✔ Привязка Shift+Ctrl+;
- Привязка к ▸
- Закрепить Направляющие Alt+Ctrl+;
- Очистить Направляющие
- Новая Направляющая...
- Закрепить Секторы
- Очистить Секторы

Окно

- Упорядочить ▸
- Проект ▸
- Гистограмма
- Действия Alt+F9
- ✔ Инструменты
- Информация F8
- История
- Каналы
- Кисти F5
- ✔ Навигатор
- ✔ Настройки
- Настройки инструментов
- Обозреватель файлов
- Образцы
- Параграф
- Подсчет Слоя
- Пути
- Символ
- ✔ Слои F7
- Стили
- ✔ Цвета F6
- ✔ Строка статуса
- ✔ 1 Без имени-1



1) Файл – Открыть, Обзор, Последние документы, Редактировать Image Ready, Закрыть, Сохранить, Печать, Просмотр, Импортировать, Экспортировать.

2)Редактирование – Отменить, Шаг вперёд, назад, Вырезать, Скопировать, Проверка орфографии, Поиск и Замена текста, Выполнить: заливку, обводку и др.

3)Изображение – Режим, Коррекция, Создать дубликат, Внешний канал, Размеры изображения, Размер холста, Кодировать.

4) Слой: Новый, Удалить слой, Создать дубликат слоя, Параметры и Стиль слоя, Добавить и включить слой маску.

5) Выделение – Всё, Отменить выделение, Выделить снова, Инвертирование, Цветовой диапазон, Модификация и др.

6) Просмотр – Варианты цветопробы, увеличение, Уменьшение, Режим экрана и др.

7) Окно: Упорядочить, Рабочая область, Палитры.

8) Справка – всевозможные справки.

9) Фильтры - это программные средства преобразования изображения.

В Photoshope можно с помощью фильтров получить огромное количество эффектов, начиная от небольшого увеличения резкости и заканчивая всевозможными искажениями.

В пункте Фильтр все допустимые фильтры объединены в группы, согласно производимому эффекту.

Фильтры подменю Artistic

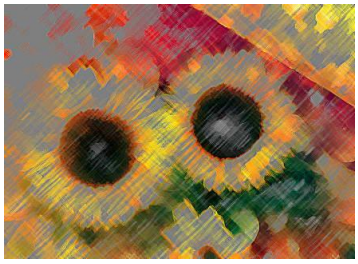


Рис.6.12. Исходное изображение.
Рис.6.13. *Colored Pencil* *(Цветной карандаш).*

Рис.6.14. *Cutout* *(Аппликация).* **Рис.6.15. *Dry Brush*** *(Сухая кисть).*

Рис.6.16. *Film Grain* *(Зернистость фотопленки).* **Рис.6.17. *Fresco*** *(Фреска).*

Рис.6.18. *Neon Glow* *(Неоновый свет*). **Рис.6.19. *Paint Daubs*** *(Масляная живопись).*

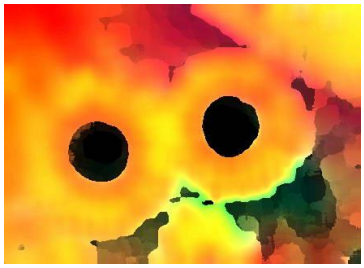


Рис.6.20. ***Palette Knife*** *(Шпатель)*Фильтры подменю Artistic.
Рис.6.21 .*Исходное изображение.*

Рис.6.22. ***Plastic Wrap*** *(Целлофановая упаковка).* **Рис.6.23.** ***Poster Edges****(Плакатные края).*

Рис.6.24. ***Rough Pastels*** *(Пастель).* **Рис.6.25.** ***Smudge Stick*** *(Растушевка.)*

Рис.6.26. ***Sponge*** *(Губка).* **Рис.6.27.** ***Watercolor*** *(Акварель).*

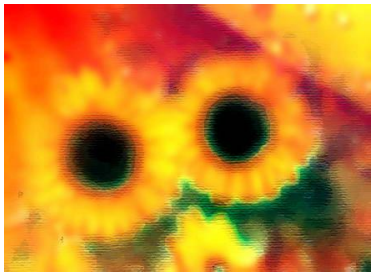


Рис.6.28. ***Underpaying*** *(Грунтовка).*

Фильтры подменю Blur.

Рис.6.29. *Исходное изображение.* **Рис.6.30.** ***Blur More*** *(Размытие плюс).*

Рис.6.31. ***Gaussian Blur*** *(Размытие по Гауссу).* **Рис.6.32.** ***Motion Blur*** *(Размытие в движении).*

Рис.6.33. ***Radial Blur*** *(Радиальное размытие).* **Рис.6.34.** ***Smart Blur*** *(Умное размытие), режим* ***Normal*** *(Нормальный).*

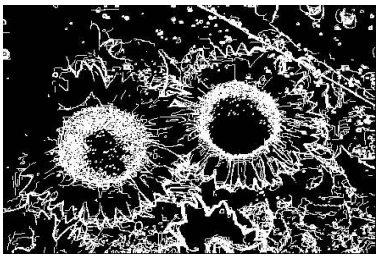


Рис.6.35. ***Smart Blur*** *(Умное размытие), режим* ***Edges Only*** *(Только контуры).*
Рис.6.36. ***Smart Blur*** *(Умное размытие), режим* ***Overlay Edge*** *(Режим наложения).*

Фильтры подменю Brush Strokes

6.5. Средства работы с векторной графикой
Программа COREL DRAW

Векторные графические редакторы применяют для создания графических изображений высокой точности и чёткости – чертежи, схемы, диаграммы, фирменные логотипы. Есть несколько редакторов Adobe Illustrator, Makromedia Freehand, Corel Draw и другие. Все эти программы схожие, основаны на одних и тех же принципах, имеют одинаковый инструмент, приёмы создания изображения и т.д., но и есть различия в некоторых деталях. Это файлы с расширениями .CDR, .CMX, .WMF

Основные понятия векторной графики.

1) Основным объектом векторной графики является *линия*. В некоторых редакторах называют кривой, или же вместо кривой *контур. К*онтуры состоят из сегментов и опорных точек. Каждый сегмент имеет две опорные точки. Свойства опорных точек определяют форму сегментов. Управляя расположением опорных точек, можно управлять формат контуров, составляющих изображение.

2) Каждый контур может иметь две или более *опорных точек*, в некоторых редакторах их называют *узлами*.

3) Элемент контура, заключённый между двумя опорными точками, называется *сегментом контура*. Если контур имеет более двух опорных точек, то он состоит из нескольких сегментов.

Форму контура изменяют перемещением опорных точек, изменяя их свойства, добавляем новые опорные точки, удаляем части опорных точек контура.

4) Контур может быть *открытым* или *замкнутым*: если последняя опорная точка контура одновременно является и его первой точкой, то контур считается замкнутым. В противном случае контур – открыт.

5) Контур является элементарным графическим *объектом*. Из контуров можно создавать новые объекты или их группы. С несколькими контурами можно выполнять следующие операции:

1.Группирование. Каждый контур группы сохраняет свои опорные точки и свойства.

2.Комбинирование. Опорные точки остаются неизменными, но свойства составного контура становятся новыми.

3.Объединение. Образуются новые опорные точки и изменяются свойства исходных объектов. Из элементарных объектов создаются сложные объекты: новые контуры, составные контуры и группы объектов.

Свойства объектов в векторной графике.

Каждый объект в векторной графике обладает свойствами. Основными свойствами векторных объектов являются *обводка* и *заливка*. Параметры, описывающие эти свойства, определяют толщину, цвет и форму линий, образующих контур, а также цвет и текстуру внутренней области контура.

1) ***Параметры обводки***: цвет, толщина, тип линии, тип концов линии. Изменяя свойства обводки, мы управляем тем, как будет отображаться контур.

2) *Свойства заливки*. При создании замкнутого контура, его внутренняя область автоматически заливается, соответствуя текущим настройкам цвета заливки.

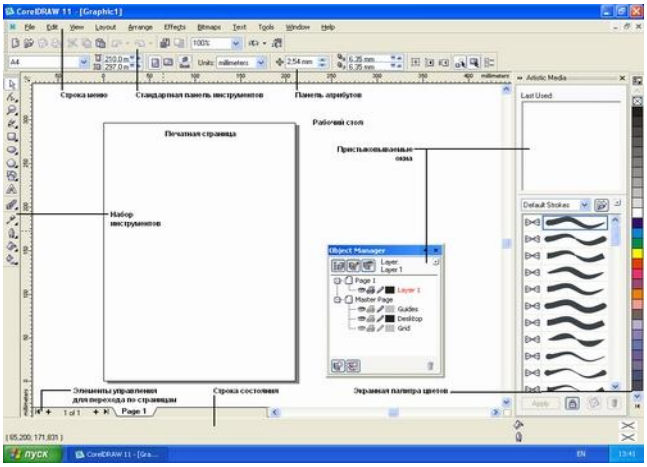
3) *Параметры заливки*. Основным параметром заливки является информация о том, чем заливается контур.

Существует несколько типов заливки: заливка основным цветом (заливается внутренняя часть контура, выбранным цветом); градиентная заливка (выбирается два цвета, и имеет место плавный переход от одного к другому); текстурная заливка (внутренняя область контура покрывается одним узором, с регулярной структурой); заливка изображением карты (Параметр – адрес файла растрового изображения, которое является заполнителем). Такое растровое изображение называют картой. Этот метод заливки есть во всех редакторах.

Программа COREL DRAW

Программа CorelDRAW 11, составляющая основу современного набора программных средств канадскойфирмы Corel, была выпущена в августе 2002 г. Она представляет собой результат двенадцатилетней эволюции, обладает удивительной универсальностью и мощностью, будучи в равной степени полезной и в промышленном дизайне, и в разработке рекламной продукции, и в подготовке публикаций, и в создании изображений для web-страниц.

Запуск-завершение программы происходит стандартным образом. После запуска программы появляется окно редактора *COREL DRAW*. Общий вид окна:



1) Строка заголовков.

2) Строка меню (11 пунктов).

3) Стандартная панель инструментов.

4) Панель свойства.

5) Панель инструментов.

6) Рабочее пространство.

7) Стандарт.

8) Строка состояния.

В соответствии со стандартами Windows под строкой заголовка окна располагается строка меню. В CorelDRAW меню очень сложное, с большим числом подменю и команд. Как и в любой другой программе, меню обеспечивает доступ к большинству функций CorelDRAW, но очень многие действия могут выполняться и без него. Еще более запутывает пользователя возможность неограниченной настройки меню — при желании любые команды и инструменты CorelDRAW можно переместить в любое меню. В левой части рабочего пространства расположен специфический для продуктов фирмы Corel элемент интерфейса — так называемый набор инструментов (toolbox). Формально являясь просто одной из множества инструментальных панелей программы, фактически он предназначен для выбора рабочего режима и поэтому используется чаще других. Выбор режима осуществляется щелчком мышью на одной из кнопок набора инструментов - это называется выбором инструмента. С выбора инструментов начинаются практически все действия пользователя над объектами изображения.

Некоторые кнопки инструментов снабжены треугольником в нижнем правом углу. Это — указание, что на самом деле с кнопкой связан не один, а несколько инструментов. Чтобы увидеть их всех, вместо быстрого щелчка кнопкой мыши ее следует нажать (отпустив только после паузы в одну-две секунды) - на экране раскроется панель конкретного инструмента. На рис. 6.37 представлена панель, раскрывающаяся кнопкой инструмента Fill (Заливка).

Рис.6.37. Панель инструмента Fill (Заливка) в раскрытом состоянии.

Чтобы выбрать инструмент, достаточно щелкнуть на его кнопке. Как правило, каждому из инструментов соответствует своя форма указателя мыши. В расположенной под строкой меню стандартной панели инструментов (toolbar) расположены элементы управления, соответствующие наиболее часто выполняемым командам: открытию, сохранению и закрытию файлов иллюстраций, операциям с системным буфером обмена, режимам и масштабу просмотра иллюстраций.

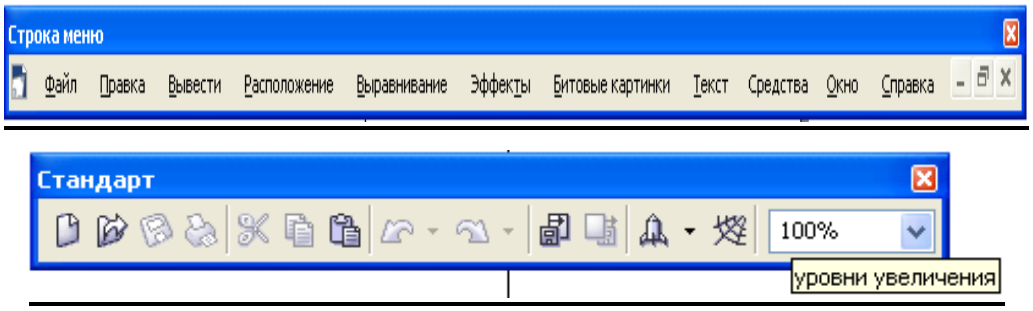
Ниже стандартной панели инструментов по умолчанию располагается панель атрибутов (property bar). Она представляет собой совокупность элементов управления, соответствующих управляющим параметрам выделенного объекта и стандартным операциям, которые можно выполнить над ним с помощью выбранного инструмента. Содержимое панели атрибутов постоянно меняется, и в последующих уроках мы будем уделять ей очень много внимания, так как она является основным рабочим инструментом пользователя.

Вдоль правой границы окна расположена экранная палитра цветов (color palette). Она применяется для задания цвета заливки и обводки объектов иллюстрации

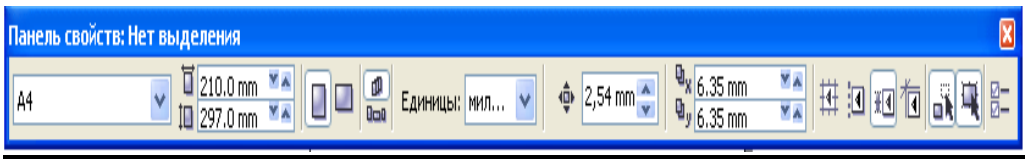
У нижнего края окна CorelDRAW находится строка состояния (status bar). В ней в процессе работы выводятся сведения о выделенном объекте и много вспомогательной информации о режиме работы программы. Основная часть рабочего пространства CorelDRAW отведена для размещения окон документов (drawing windows) CorelDRAW. После создания документа CorelDRAW в таком окне видно только изображение печатной страницы, на которой будет размещаться иллюстрация. Границы страницы показаны в виде рамки с тенью, однако они не являются элементом изображения.

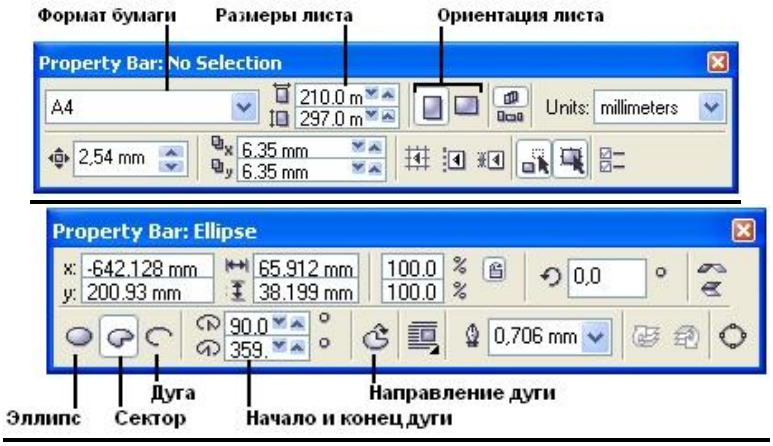
Объекты, из которых будет далее строиться иллюстрация, должны располагаться в пределах этих границ. Остальное пространство окна иллюстрации имеет свое название — рабочий стол — и используется обычно как временное хранилище объектов. Размер рабочего стола CorelDRAW значительно больше, чем его видимая на экране часть. Для просмотра невидимой части окна служат полосы прокрутки, расположенные по правому и нижнему краям окна документа. Слева от горизонтальной полосы прокрутки располагаются элементы управления, позволяющие переходить между отдельными страницами многостраничных документов — кнопки и ярлычки с названиями страниц, вместе образующие так называемый навигатор. На левом и верхнем краях окна документа расположены координатные линейки (rulers), служащие для измерения координат объектов и размещения направляющих

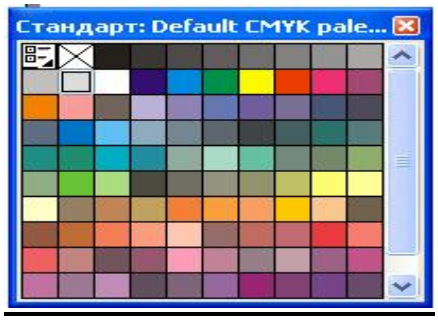
Огромную роль в интерфейсе CorelDRAW играют пристыковываемые окна (dockers), в свернутом виде представляющие собой ярлычки с названиями, расположенные слева от экранной палитры цветов. По своим функциям они напоминают диалоговые окна, но в отличие от большинства диалоговых окон могут постоянно присутствовать в рабочем пространстве. Мы будем знакомиться с пристыковываемыми окнами по мере освоения приемов работы с объектами CorelDRAW. Пока отметим, что пристыковываемые окна могут располагаться, как в середине рабочего стола, так у одного из его краев («пристыковываться» к нему). В свернутом виде от пристыковываемого окна виден только заголовок или, если оно пристыковано — только ярлычок с названием. Это позволяет компактно располагать в рабочем пространстве большое число элементов управления.

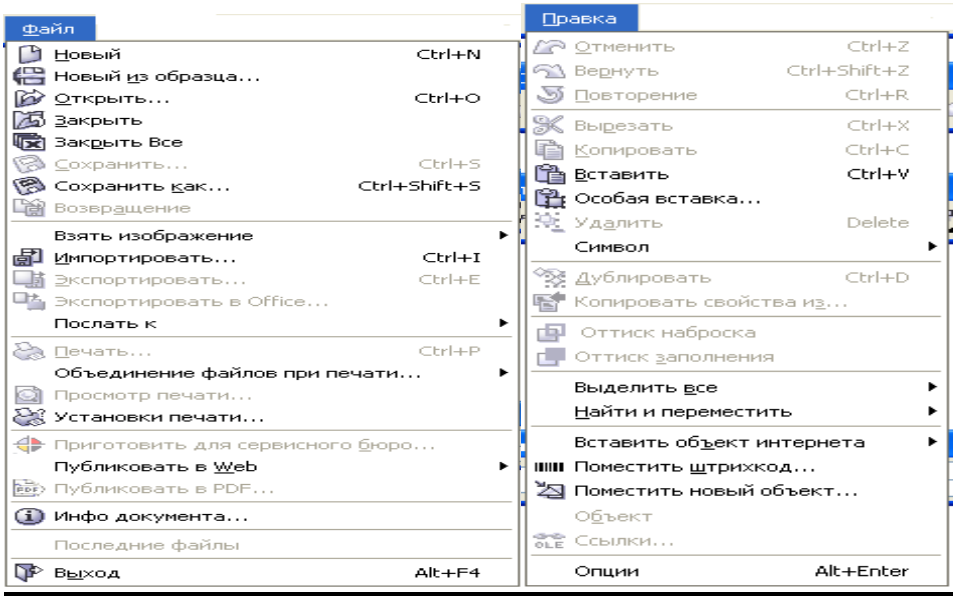



Панель инструментов


Панель свойств: Нет выделения
A4
210.0 mm
297.0 mm
Единицы: мил...
2,54 mm
6.35 mm
6.35 mm


Формат бумаги
Размеры листа
Ориентация листа
Property Bar: No Selection
A4
210.0 m
297.0 m
Units: millimeters
2,54 mm
6.35 mm
6.35 mm
Property Bar: Ellipse
x: -642.128 mm
y: 200.93 mm
65.912 mm
38.199 mm
100.0
100.0
0,0
90.0
359.
0,706 mm
Эллипс
Сектор
Дуга
Начало и конец дуги
Направление дуги


Стандарт: Default CMYK pale...


Файл
Новый Ctrl+N
Новый из образца...
Открыть... Ctrl+O
Закрыть
Закрыть Все
Сохранить... Ctrl+S
Сохранить как... Ctrl+Shift+S
Возвращение
Взять изображение
Импортировать... Ctrl+I
Экспортировать... Ctrl+E
Экспортировать в Office...
Послать к
Печать... Ctrl+P
Объединение файлов при печати...
Просмотр печати...
Установки печати...
Приготовить для сервисного бюро...
Публиковать в Web
Публиковать в PDF...
Инфо документа...
Последние файлы
Выход Alt+F4
Правка
Отменить Ctrl+Z
Вернуть Ctrl+Shift+Z
Повторение Ctrl+R
Вырезать Ctrl+X
Копировать Ctrl+C
Вставить Ctrl+V
Особая вставка...
Удалить Delete
Символ
Дублировать Ctrl+D
Копировать свойства из...
Оттиск наброска
Оттиск заполнения
Выделить все
Найти и переместить
Вставить объект интернета
Поместить штрихкод...
Поместить новый объект...
Объект
Ссылки...
Опции Alt+Enter

Вывести

- Простой каркас
- Каркас
- Набросок
- Нормально
- Расширенный
- Показать на весь экран — F9
- Показать только выбранные
- Вид сортировщика страниц
- Линейки
- Сетка
- Направляющие
- Вывести
- Включить поворот
- Прикрепить к сетке — Ctrl+Y
- Прикрепить к направляющим
- Прикрепить к объектам — Alt+Z
- Динамические направляющие — Alt+Shift+D
- Установка сетки и линейки...
- Установка направляющих...
- Установки прикрепления к объектам...
- Установки динамических направляющих...

Расположение

- Поместить страницу...
- Удалить страницу...
- Переименовать страницу...
- Перейти к странице...
- Переключить положение страницы
- Установки страницы...
- Задний фон страницы...

Выравнивание

- Конвертирование
- Очистить конвертирования
- Выровнять и выпустить
- Упорядочивание
- Группа — Ctrl+G
- Разгрупировать — Ctrl+U
- Разгруппировать Все
- Комбинировать — Ctrl+L
- Разбить на части — Ctrl+K
- Блокировать объект
- Разблокировать объект
- Разблокировать все объекты
- Формат
- Преобразовать в кривую — Ctrl+Q
- Преобразовать Набросок в объект — Ctrl+Shift+Q
- Закрыть путь

Эффекты

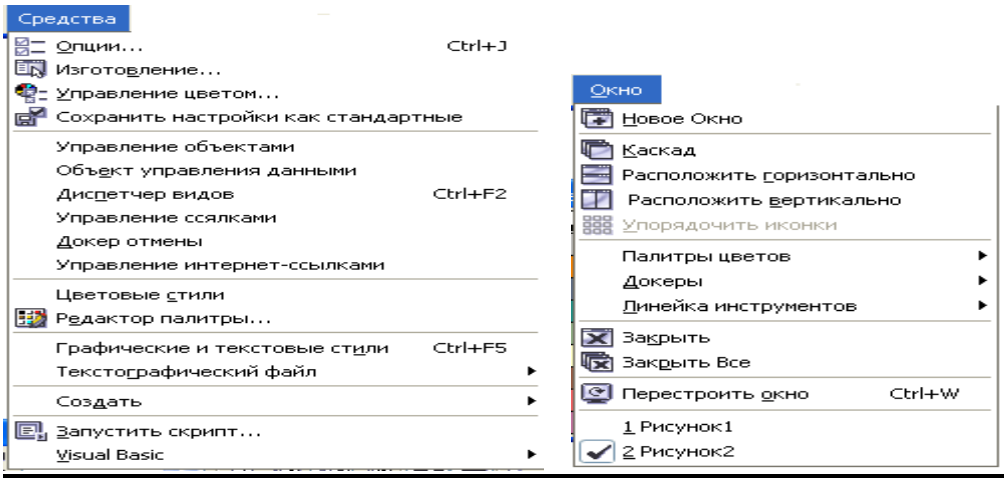
- Коррекция
- Трансформация
- Исправление
- Художественный способ
- Смешанное
- Контур — Ctrl+F9
- Оболочка — Ctrl+F7
- Выпуклость
- Линза — Alt+F3
- Добавить перспективу
- PowerClip
- Поворот
- Убрать эффект
- Копировать эффект
- Клонировать эффект

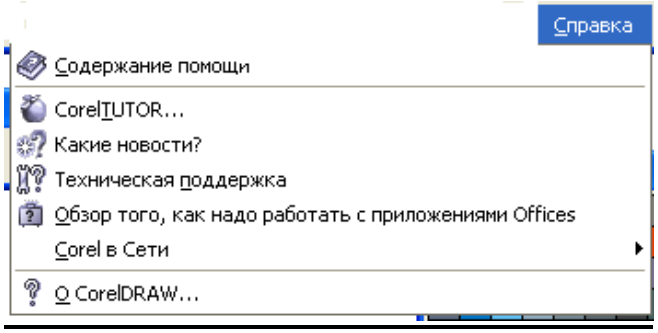
Битовые картинки

- Конвертировать в битовое изображение...
- Править битовую картинку...
- Обрезать битовое изображение
- Трассировать изображение...
- Делать повторную выборку...
- Режим
- Увеличить битовое изображение
- Цветовая маска битмапа
- Разорвать связь
- Обновить по ссылке
- 3D эффекты
- Живописный штрих
- Размывка
- Камера
- Преобразование цвета
- Контур
- Творческие
- Искажение
- Шум
- Резкость
- Добавления

Текст

- Формат текста... — Ctrl+T
- Изменить текст... — Ctrl+Shift+T
- Вставить символ — Ctrl+F11
- Подогнать текст по пути
- Подгонка текста по рамке
- Выравнивание по базе — Alt+F12
- Выровнять текст
- Инструменты письма
- Кодировка...
- Сменить регистр...
- Сделать текст веб-совместимым
- Преобразовать — Ctrl+F8
- Статистика текста...
- Вывести непечатаемые символы
- Связать
- Развязать





<u>Создание нового документа.</u>

1. На рабочей области представлена рабочая страница, хотя изображение совсем не обязательно располагать только на ней, а потом перемещать её в нужное место.

2. Страницу создают команды: Файл – Новый. При этом создаётся страница, принятого по умолчанию формата.

3. Параметр. Файл – Параметры документа. Параметры документа: можно просмотреть и изменить параметры страницы.

4. Если нужно изменить нестандартный формат, то в раскрывающемся листе “формат”, выбирают пункт заказной, и только потом задают его размеры.

Панель инструментов является основным средством для создания рисунков. Основные инструменты в панели составляют 5 групп знаков:

1. В первую группу входят значки инструментов - выделение объектов. Инструментом выделения выделяют объекты целиком. Инструментом частное выделение выделяют часть контура.

2. Вторая группа инструментов предназначена для рисования: эллипс, прямоугольник. Инструмент “карандаш” служит для создания контуров произвольной формы. Перо - основной инструмент. Оно служит для создания прямоугольных, прямолинейных и криволинейных сегментов.

Текст - применяют для создания текстовых объектов.

Ножницы - выделение контуров.

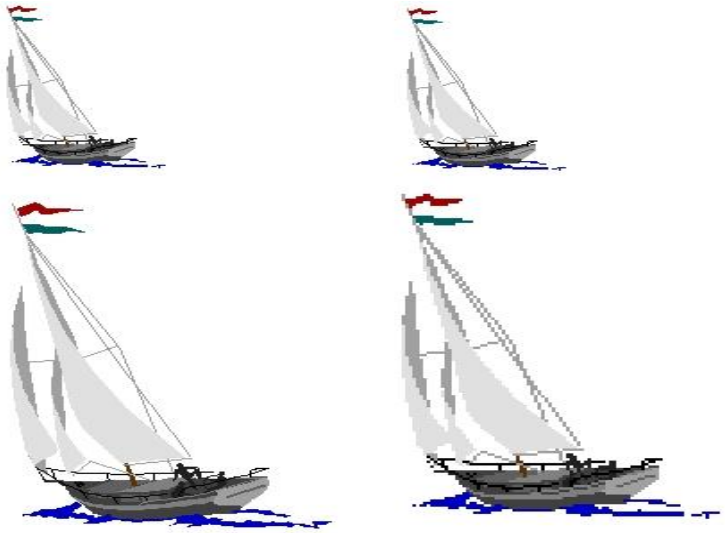
3.Инструменты этой группы предназначены для манипуляции с выделенными объектами.

4. Четвертая группа включает инструменты превращения диаграмм. Превращение представляет собой группу альтернативных инструментов для выполнения трансформации объекта. Диаграмма представляет собой обширную группу альтернативных инструментов деловой графики для построения диаграмм.

5.Последнюю группу составляют дополнительные инструменты управления параметром (масштаб, рука, линейка), инструмент для выбора цвета по образцу (пипетка). А также средства для заливки контуров (заливка, градиент).

6.6. Создание и редактирование готовых изображений в редакторе COREL DRAW

Представление векторного изображения в памяти компьютера сложнее, чем точечного (хотя, как правило, при этом оно намного компактнее). Векторное изображение существенно более гибко в работе. Чтобы увеличить или уменьшить его, требуется всего лишь изменить один управляющий параметр изображения — масштаб. При этом размер файла с векторным изображением не увеличится ни на один байт. Внесенные изменения будут учтены при рендеринге, и четкость изображения не пострадает. На рисунке представлены результаты увеличения точечного и векторного изображения.



Векторное изображение (слева) можно, в отличие отточенного (справа), масштабировать без потери четкости и детали.

Как уже отмечалось в предыдущем уроке, основой работы с изображением в CorelDRAW являются объекты. Несколько упрощая, можно констатировать следующее: все графические объекты, с которыми приходится иметь дело пользователю этой программы, можно разбить на две категории — линии и примитивы Линии, их форма произвольна и не связана никакими ограничениями, кроме творческого замысла художника. Сказать «построим линию» означает не сказать почти ничего, и никаких четких зрительных образов за понятием «линия» не стоит.

Однако если сказать, что четыре попарно равных отрезка прямых линий, соединяясь в конечных точках, образуют при этом четыре прямых угла, то в сознании всплывает не только четкий зрительный образ, но и термин для обозначения подобных объектов — «прямоугольник». Конечно, нельзя построить просто прямоугольник, без дополнительного уточнения не обойтись, но отличия прямоугольников от других графических объектов достаточно очевидны и позволяют выделить их в отдельный класс объектов. Чтобы выделить конкретный объект этого класса, достаточно задать значения его атрибутов, например высоту и ширину. В CorelDRAW имеется несколько классов таких четко определяемых графических объектов под обобщающим названием примитивы.

В этом уроке мы познакомимся с набором примитивов CorelDRAW 11, их атрибутами и способами построения на рисунке, а также с другими объектами, которые, не являясь в строгом смысле слова примитивами, во многом похожи на них.

Упражнение 1. Построение прямоугольников.

Выполняя это упражнение, мы познакомимся с основным способом построения прямоугольников, а заодно и с некоторыми вспомогательными элементами рабочей среды CorelDRAW 11.

1. Создайте новый документ. Для этого упражнения можно выбрать лист бумаги с произвольной ориентацией любого размера.

2. Щелкните на кнопке инструмента Rectanchaptere (Прямоугольник) в наборе инструментов. После этого указатель мыши на экране примет форму перекрестья с прямоугольником (рис.6.38) — это визуальное подтверждение того, что в настоящий момент активен инструмент построения прямоугольников.

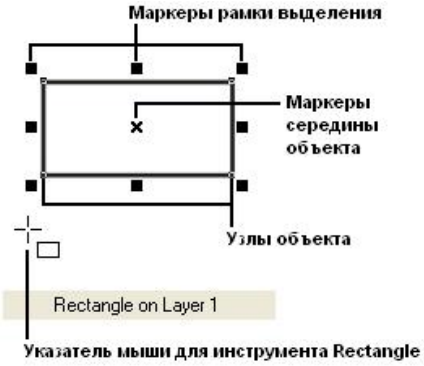


Рис.6.38. Выделенный прямоугольник, элементы рамки выделения и сообщение в строке состояния.

3.Чтобы построить прямоугольник, перетащите указатель инструмента Rectanchaptere (Прямоугольник) по диагонали создаваемого объекта. Обратите внимание, что в процессе перетаскивания указателя мыши в строке состояния выводятся текущие значения высоты и ширины прямоугольника, а на экране отображается его постоянно меняющийся абрис.

В момент отпускания кнопки мыши при окончании перетаскивания на экране появляется прямоугольник в окружении маркеров рамк)/выделения и с маркером центра (см. рис. 6.38), а в строке состояния - «сообщение о том, что выделен объект, относящийся к классу прямоугольников.

Рамкой выделения называется группа из восьми маркеров (небольших квадратов с черной заливкой), обозначающих на экране габариты выделенного объекта или нескольких объектов. В центре рамки выделения находится маркер центра в виде косого крестика. Элементы рамки выделения используются при преобразованиях объектов, которые обсуждаются в следующих уроках. В настоящий момент для нас важно, что на панели атрибутов и в строке состояния отображаются сведения о выделенном объекте.

Упражнение 2. Применение клавиш-модификаторов

Выполняя это упражнение, мы научимся пользоваться клавишами-модификаторами, позволяющими упростить построение прямоугольников при наличии дополнительных ограничений.

1. Повторите прием построения прямоугольника, освоенный в предыдущем упражнении, но в процессе перетаскивания указателя мыши но диагонали будущего объекта удерживайте нажатой клавишу Ctrl. Обратите внимание на то, что при этом абрис строящегося объекта независимо от направления перемещения мыши остается строго квадратным. Это — самый простой способ построения квадратов в CorelDRAW.

2. Постройте еще один прямоугольник, но теперь при перетаскивании указателя мыши удерживайте нажатой клавишу Shift. Обратите внимание, что если все ранее построенные прямоугольники располагались так, что в точке начала перетаскивания указателя мыши оказывался угловой маркер, то теперь там оказался маркер середины. Этот прием очень удобен, когда заранее известно, где должен располагаться центр прямоугольника.

Примечание.

Оба модификатора можно использовать совместно, то есть если при перетаскивании указателя инструмента Rectanchaptere (Прямоугольник) одновременно удерживать нажатыми клавиши Ctrl и Shift, то будет построен квадрат «от середины».

Теперь познакомимся с тем, как выглядит панель атрибутов для прямоугольников (рис.6.39) и какие элементы управления на ней расположены.

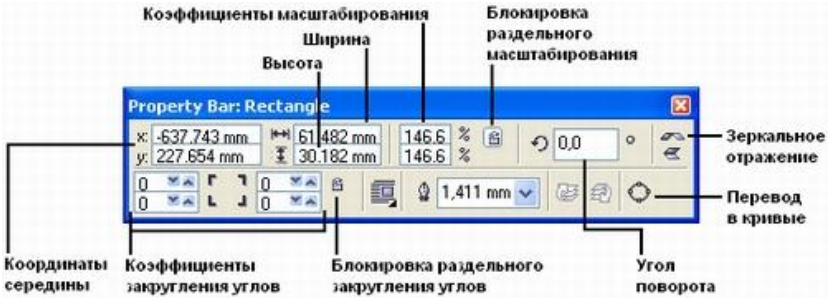


Рис.6.39. Панель атрибутов при работе с прямоугольниками (панель перемещена в центр рабочего пространства и представлена в виде окна).

На панели атрибутов представлены элементы управления, определяющие параметры модели объекта (в данном случае — прямоугольника), и кнопки, позволяющие выполнять стандартные действия над объектами этого класса.

Object(s) Position (Координаты середины). Два поля, содержащие точные значения координаты середины прямоугольника в текущей системе координат (обычно связанной с левым нижним углом страницы). Введя в эти поля новые значения, можно переместить прямоугольник.

Object(s) Size (Высота и ширина). Значения в этих полях управляют геометрическими размерами прямоугольника. Меняя их, можно сделать прямоугольник больше или меньше

Упражнение 3. Закругление углов прямоугольника.

В этом упражнении мы ознакомимся с приемами закругления углов прямоугольника — всех вместе и по отдельности.

1. Постройте прямоугольник произвольных размеров.

2. Выберите в наборе инструментов инструмент Shape (Форма), переместите его указатель на любой из расположенных в углах прямоугольника узлов и перетащите его вдоль любой из сторон прямоугольника. Обратите внимание, что по мере удаления указателя мыши от угла прямоугольника все четыре угла начинают закругляться, причем, чем дальше перетаскивается указатель, тем больше становится радиус закругления (рис.6.40).

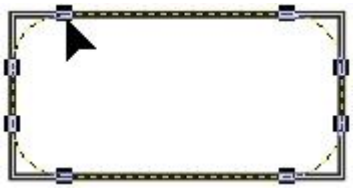


Рис.6.40. Закругление углов прямоугольника инструментом Shape (Форма).

3. Постройте еще один прямоугольник рядом с первым. Теперь попробуем закруглить только один из его углов. Для этого наведите указатель инструмента Rectanchaptere (Прямоугольник) на узел, расположенный в правом верхнем углу прямоугольника, и перед началом перетаскивания узла щелкните мышью.

4. После щелчка сбрасывается выделение всех узлов, кроме того, на котором был выполнен щелчок. Теперь перетаскивание узла приводит к закруглению только выделенного угла прямоугольника.

5. Перетаскивайте узел вдоль короткой стороны прямоугольника «до упора». Обратите внимание, что один из пары узлов, образовавшейся из углового узла прямоугольника, перемещается мышью, а второй движется синхронно с ним вдоль смежной стороны. Перемещение прекращается, когда один из узлов (неважно который) достигнет середины стороны прямоугольника.

Внимание!

Максимальный радиус закругления угла прямоугольника (100%) равен половине длины его короткой стороны.

6. Щелчком мыши отожмите на панели атрибутов кнопку блокировки раздельного закругления углов. Введите в левое нижнее поле из группы полей для задания коэффициентов закругления углов значение 50 и щелкните в любом другом поле той же панели. Обратите внимание, как закруглился левый нижний угол.

Эллипсы.

При работе с CorelDRAW эллипсы можно считать просто растянутыми вдоль одного из диаметров окружностями. Поскольку теперь мы умеем строить прямоугольники, научиться строить эллипсы будет значительно проще — большинство базовых приемов уже освоены.

Так же как класс объектов «прямоугольник» намного шире геометрического понятия «прямоугольник», класс объектов «эллипс» включает в себя объекты, с геометрической точки зрения эллипсами не являющиеся, а именно секторы и дуги эллипсов, которые получаются из эллипса приемами, аналогичными закруглению углов прямоугольника.

В геометрии размеры эллипса определяются размерами его полуосей, в CorelDRAW — размерами габаритного прямоугольника (совпадающего с рамкой выделения). Эллипс касается рамки выделения в тех местах, где у нее располагаются четыре средних маркера сторон (рис.6.41). У только что построенного эллипса имеется всего один узел.

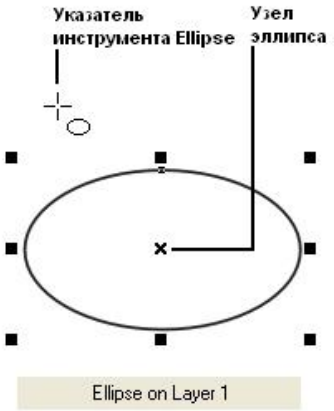


Рис.6.41. Эллипс, рамка выделения, указатель инструмента Ellipse (Эллипс) и сообщение в строке состояния.

Познакомимся с приемами построения и модификации эллипсов.

Упражнение 4. Построение и модификация эллипсов, дуг и секторов.

1. Чтобы не перегружать графикой страницу, на которой мы работали с прямоугольниками, начнем, со вставки в документ CorelDRAW еще одной страницы. Для этого воспользуйтесь командой Layout > Insert Page (Макет > Добавить страницу) н щелкните на кнопке ОК в раскрывшемся диалоговом окне.

2. Выберите в наборе инструментов инструмент Ellipse (Эллипс) и перетащите указатель инструмента по диагонали габаритной рамки будущего эллипса. Обратите внимание на изменение сообщений в строке состояния н значений в панели атрибутов в процессе перетаскивания, после отпускания кнопки мыши на рисунке появляется эллипс в рамке выделения.

Подсказка.

Клавиши-модификаторы работают с инструментом Ellipse (Эллипс) точно так же, как с инструментом Rectanchaptere (Прямоугольник). Удерживая нажатой клавишу Ctrl, можно построить не эллипс, а правильный круг, а клавиша Shift позволяет строить эллипс, растягивая его не от угла, а от середины габаритного прямоугольника. При удержании одновременно обеих клавиш-модификаторов будет строиться круг от центра. Освобождать клавиши-модификаторы следует только после отпускания кнопки мыши.

3. Найдите узел вновь построенного эллипса и наведите на него указатель мыши. Указатель инструмента Ellipse (Эллипс) должен смениться указателем инструмента Shape (Форма). Нажмите кнопку мыши и сместите узел по направлению к центру габаритного прямоугольника, а затем, не выходя за границу эллипса, — по часовой стрелке (рис.6.42). После отпускания кнопки мыши эллипс будет преобразован в сектор. При этом в строке состояния и в панели атрибутов будут отображаться центральный угол сектора и направления ограничивающих его радиусов.

Рис.6.42. Преобразование эллипса в сектор перетаскиванием узла внутри эллипса.

4. Постройте еще один эллипс и повторите описанную на предыдущем шаге последовательность действий, только на этот раз перемещайте узел эллипса не внутри него, а снаружи. В результате будет построена дуга эллипса, а не сектор.

Подсказка.

Если в процессе перетаскивания узла эллипса удерживать нажатой клавишу Ctrl, то центральный угол дуги или сектора будет меняться не плавно, а скачками по 15°. Это бывает удобно при построении секторов и дуг заранее заданной величины. Теперь познакомимся с элементами панели атрибутов для объекта класса «эллипс» (рис.6.43).

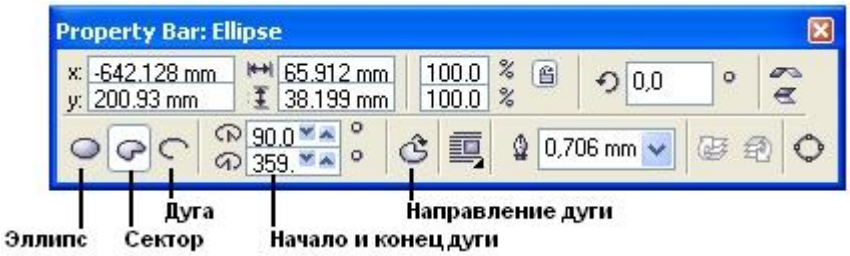


Рис.6.43. Панель атрибутов для эллипсов.

Упражнение 5. Построение и модификация многоугольников.

Выполняя это упражнение, мы освоим приемы построения многоугольников и их модификации с помощью инструмента Polygon (Многоугольник).

1. Вставьте в открытый документ CorelDRAW новую страницу. На этот раз проделаем это с помощью контекстного меню. Щелкните на ярлычке последней страницы документа правой кнопкой мыши и выберите в контекстном меню команду Insert Page After (Вставить страницу после).

На этот раз диалогового окна с запросом параметров страницы не появится — новая страница будет создана с теми же значениями атрибутов, что и предыдущая. Вызовите контекстное меню новой страницы щелчком правой кнопки мыши на ее ярлычке и воспользуйтесь командой Rename Page (Переименовать страницу), чтобы назначить ей имя Многоугольники.

2. Выберите инструмент Polygon (Многоугольник), щелкнув в наборе инструментов на соответствующей кнопке (см. выше). На панели атрибутов установите число узлов базового многоугольника равным 10. Теперь по умолчанию будут строиться десятиугольники.

3.Постройте многоугольник, перетащив по диагонали его габаритного прямоугольника указатель инструмента Polygon (Многоугольник). Рядом постройте еще один многоугольник, но в ходе перетаскивания указателя инструмента удерживайте нажатой клавишу Ctrl. Второй многоугольник должен получиться равносторонним.

4.Обратите внимание на узлы построенного многоугольника. Наведите указатель инструмента Polygon (Многоугольник) на любой из узлов, расположенных в серединах сторон многоугольника, — при этом форма указателя должна измениться, что говорит о временной активизации инструмента Shape (Форма). Удерживая нажатой клавишу Ctrl, перетащите этот узел по радиусу примерно на ноловипу.расстояния до центра. Вместе с «захваченным» узлом будут перемещаться и все остальные дополнительные узлы, размещенные в серединах сторон многоугольника. В результате получится фигура, похожая на метательное оружие ниндзя — сюрикен (рис. 6.44 в середине).

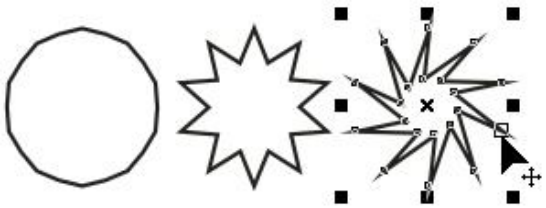


Рис.6.44. Исходный многоугольник и его модификации, полученные перетаскиванием узлов инструментом Polygon (Многоугольник).

Примечание.

Нажатие клавиши Ctrl при перетаскивании узлов многоугольника ограничивает свободу их перемещения движением по радиусам базового эллипса.

5. Теперь наведите указатель инструмента на основной узел, расположенный в одной из вершин многоугольника, и перетащите его, но уже не по радиусу, а по часовой стрелке вокруг центра. На рис.6.44 (справа) видны указатель инструмента Shape (Форма) и узлы многоугольника в процессе перетаскивания. В результате лучи сюрикена заострятся еще больше, и фигура утратит осевую симметрию, сохранив симметрию центральную.

6. Постройте еще один многоугольник, па этот раз, стараясь, чтобы он был вписан не в круг, а в горизонтально вытянутый эллипс (рис.6.45, слева). Наведите указатель инструмента Polygon (Многоугольник) на маркер середины рамки выделения (после чего он должен превратиться в четырехглавую стрелку) и перетащите многоугольник вправо. Перед тем как отпустить левую кнопку мыши, щелкните ее правой кнопкой (рядом с четырехглавой стрелкой должен появиться значок «плюс»). В результате на странице появится смещенная копия ранее построенного многоугольника. Повторите эту операцию еще два раза, чтобы получился ряд из четырех одинаковых «сплюснутых» десятиугольников.

Подсказка.

Такая процедура, позволяющая совмещать создание копии выделенного объекта с последующим перемещением, очень удобна и ее стоит запомнить.

Рис.6.45. Исходный многоугольник и результаты его преобразования в звезду.

7. Выделите первую копию, щелкнув па ней указателем инструмента Polygon (Многоугольник), и щелкните на кнопке переключения режимов многоугольника и звезды. Выпуклый многоугольник превратится в звезду, а в поле заострения углов многоугольника появится значение 1.

8. Повторите то же действие со второй копией, но после преобразования в звезду переместите ползунок поля заострения на одно деление вправо. В результате узлы базового многоугольника будут соединены через два, и лучи звезды станут острее.

9. Для третьей копии переместите ползунок заострения в крайнее правое положение. Значение заострения будет равно трем, и увеличить его не удастся, поскольку соединение узлов базового многоугольника через четыре приведет к его распаду на пять отрезков.

10. В завершение упражнения попробуйте использовать описанные выше приемы модификации путем перетаскивания узлов многоугольника инструментом Polygon (Многоугольник) при нажатой и не нажатой клавише Ctrl. Только не увлекайтесь — это по-настоящему завораживающее занятие с четко выраженным гипнотическим воздействием, и оторваться от него очень трудно, особенно если число сторон в многоугольнике велико!

Упражнение 6. Построение супер линии.

Выполняя это упражнение, мы повторим приемы построения линий и освоим приемы работы с суперлиниями на примере режима кисти.

1. Вставьте в открытый документ CorelDRAW новую страницу и, пользуясь приемами построения линии, постройте в ее верхней части стилизованное изображение змеи (рис.6.46, сверху). Вначале постройте замкнутую кривую, соответствующую абрису головы и тела змеи. В качестве глаз и ноздрей постройте небольшие эллипсы, а зигзагом на спине послужит ломаная линия, состоящая из прямолинейных сегментов. Если абрис получится кривоватым — не беда, в последующих уроках мы ознакомимся с приемами, позволяющими не только с идеальной точностью

строить подобные изображения, но и корректировать их, а для этого упражнения высокая точность не нужна.

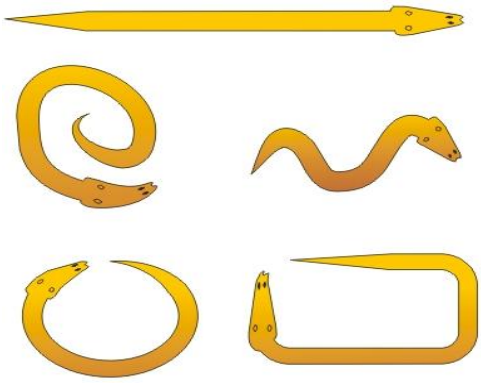


Рис.6.46. Определение нового мазка и его применение для построения супер линий.

2. Теперь необходимо выделить все составные части изображения змеи. Проще всего это делается инструментом Pick (Выбор): выберите его и щелкните на свободном месте страницы, отменив, таким образом, выделение. Затем перетащите указатель инструмента по диагонали воображаемого прямоугольника, охватывающего изображение змеи целиком. После отпускания кнопки мыши выделенными окажутся все элементы изображения, оказавшиеся внутри этого прямоугольника.

3. Выберите в наборе инструментов инструмент Artistic Media (Супер линия) и включите режим кисти, щелкнув на соответствующей кнопке панели атрибутов. Затем щелкните там же на кнопке с. изображением дискеты и задайте имя файла для сохранения мазка — например, zmejuka.cmx. После щелчка на кнопке OK новый мазок готов к использованию.

4. Отмените выделение изображения змеи, нажав клавишу Esc. Перетащите указатель инструмента Artistic Media (Суперлиния) слева направо, но волнообразной траектории. После отпускания кнопки мыши наша змея... зазмеилась!

5. Чтобы свернуть змею в клубок, постройте с помощью инструмента Spiral (Спираль) логарифмическую спираль на 2-3 витка.

Выберите инструмент Artistic Media (Суперлиния) и, раскрыв список мазков, щелкните на образце с упрощенным изображением змеи. Если клубок окажется слишком плотным, попробуйте изменить ширину суперлинии или удалить объект и повторить этот шаг, увеличив коэффициент расширения спирали.

6. Чтобы свернуть змею в кольцо, используйте в качестве управляющей линии эллипс, построенный соответствующим инструментом.

7. Чтобы «изготовить» оригинальную прямоугольную рамку, постройте прямоугольник, а затем закруглите три его угла — кроме верхнего левого. Это поможет избавиться от резких изломов супер линии, в которой построенный прямоугольник будет играть роль управляющей линии.

Режим распылителя.

Режим распылителя инструмента Artistic Media (Суперлиния) формирует не один подчиненный объект, а целую группу, размещая копии заранее определенного изображения или его отдельных частей (шаблон распылителя) вдоль управляющей липни. Два примера получающихся в результате соединенных объектов представлены на рис.6.47. В обоих случаях управляющая кривая одна и та же, а шаблоны распылителя — разные.

Впрочем, такие составные объекты можно отнести к линиям только с очень большой натяжкой. Функциональные возможности режима распылителя весьма широки. В частности, можно управлять размерами отдельных фрагментов изображения, входящих в шаблон распылителя, параметрами, определяющими их положение на управляющей линии, порядком следования фрагментов. В комплект поставки CorelDRAW входит большое число шаблонов распылителя, позволяющих добиваться впечатляющих декоративных эффектов. Предусмотрен также механизм пополнения числа шаблонов.

Чаще всего супер линиями, построенными в режиме распылителя, пользуются для формирования оригинальных рамок и фонов.

Рис. 6.46. Суперлинии, построенные в режиме распылителя.
Пристыковываемое окно **Artistic Media.**

Для построения и изменения супер линий можно пользоваться не только панелью атрибутов инструмента Artistic Media (Суперранние), по и одноименным пристыковываемым окном (рис.6.47). Чтобы отобразить его на экране, выберите команду Window > Dockers > Artistic Media (Окно > Пристыковываемые окна > Супер линия).

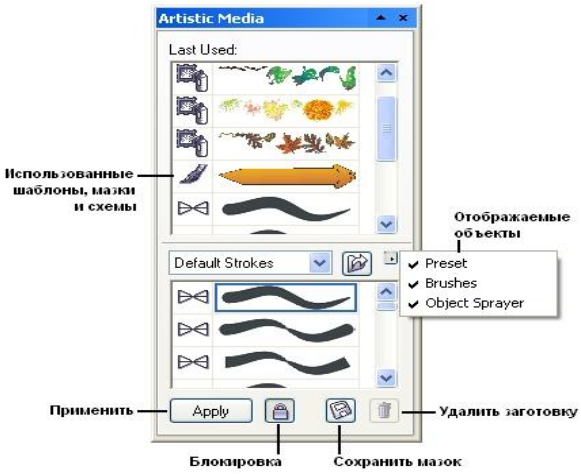


Рис.6.47. Пристыковываемое окно Artistic Media (Супер линия) и его элементы.

В пристыковываемом окне приведены два списка с изображениями заготовок супер линий, мазков и шаблонов распылителя. В нижнем списке представлен полный набор схем, соответствующий текущему содержимому

папки \CustomMediaStrokes, в верхнем — те из схем, которые были недавно использованы для построения супер линий.

Содержимым списка можно управлять при помощи меню отображаемых объектов, раскрываемого кнопкой с изображением направленного вправо треугольника. Три приведенных в меню режима включают и выключают отображение заготовок, мазков и шаблонов распылителя (сверху вниз).

Кнопка блокировки управляет режимом применения схемы к выбранной на изображении линии. Если кнопка нажата, то сразу после выбора схемы в одном из списков (щелчком мыши на соответствующей альтернативе) эта схема применяется для построения супер линии. Если кнопка блокировки отжата, для применения схемы необходимо дополнительно щелкнуть на кнопке Apply (Применить).

Использование элементов этого пристыковываемого окна позволяет существенно упростить работу с тремя режимами построения супер линий: заготовки, кисти и распылителя.

В частности, чтобы преобразовать кривую в управляющую кривую супер линию, достаточно выбрать в одном из двух списков пристыковываемого окна нужное изображение заготовки, мазка или шаблона распылителя и перетащить его мышью на эту кривую.

Чтобы запомнить изображение как мазок или шаблон распылителя, достаточно выделить его и перетащить мышью в нижний список. После отпускания кнопки мыши открывается диалоговое окно, в котором указывается, сохраняется ли изображение как мазок или как шаблон распылителя, а затем задается имя файла.

Работа с простым текстом.

В качестве упражнения для освоения приемов работы с простым текстом создадим макет условного меню еще более условного кафе. Вид макета представлен на рис.6.48. — как видите, мы пока не очень заботимся о его художественных достоинствах.

Чтобы не сковывать творческую фантазию, приведем лишь самые общие указания по этапам работы.

1. Создайте новую страницу в открытом документе CorelDRAW. Выберите инструмент Text (Текст), постройте в верхней части страницы блок простого текста произвольных размеров и перейдите в диалоговое окно Edit Text (Редактирование текста), щелкнув на соответствующей кнопке панели атрибутов.

2. Выберите в раскрывающемся списке гарнитур любую, включающую символы кириллицы (например, Arial), и введите текст меню, разбивая его на абзацы нажатием клавиши Enter. Названия разделов меню и каждого из блюд должны располагаться в отдельных абзацах. По завершении ввода закройте диалоговое окно Edit Text (Редактирование текста).

Рис. 6.48. Макет меню.

3. Задайте гарнитуры и кегли для отдельных абзацев. Дело это творческое — помните, что рисунок гарнитуры создает общее настроение макета, а от выбора кеглей для заголовков различных уровней зависит визуальная сбалансированность макета. Технически это выполняется следующим образом. Продолжая работать в диалоговом окне Edit Text (Редактирование текста), перетащите указатель мыши по части текста, подлежащего форматированию, чтобы выделить его.

Выделенный текст отображается на темном фоне. Затем в раскрывающихся списках гарнитур и кеглей выберите желаемые альтернативы.

4. Устанавливая текстовый курсор в абзацы заголовков разделов меню, задайте параметры буквиц (в данном макете заголовки невелики по длине, и целесообразно указать высоту буквиц не более двух строк). Для этого вначале щелкните на кнопке панели атрибутов с изображением буквицы, а затем откройте диалоговое окно форматирования щелчком на кнопке в диалоговом окне Edit Text (Редактирование текста). Перейдя на вкладку эффектов, установите желаемую высоту буквицы.

5. Выделяя абзацы, соответствующие названиям блюд каждого из разделов меню, задайте для них маркеры списка. В данном макете использованы маркеры из символов специальной гарнитуры Food. По завершении форматирования закройте диалоговое окно Edit Text (Редактирование текста).

6. Постройте цепочку связанных рамок простого текста. Рамок в цепочке может быть либо шесть, либо три — в последнем случае текст в двух последних рамках придется размещать в несколько колонок. Цепочку стройте, «заряжая» указатель инструмента Pick (Выбор) щелчком на нижнем индикаторе последней рамки цепочки. Выбирайте расположение рамок и их размеры в соответствии с макетом, приведенным на рисунке. Перетаскивая тем же инструментом маркеры рамки выделения, отрегулируйте размеры рамок простого текста таким образом, чтобы в первой из них разместился заголовок меню, а в последующих — по одному из его разделов.

7. Сохраните документ CorelDRAW в файле с именем texts.cdr.

ГЛОССАРИЙ

Адекватность – степень соответствия реальному объективному состоянию дела.

Актуальност*ь* – это степень соответствия информации текущему моменту времени.

Арифметико-логическое устройство, которое выполняет арифметические действия. Кроме того, выполняет сравнения, используемые в "логических" операциях сравнения. Результатом сравнения является "Истина" или "Ложь".

Базовое программное обеспечение (base software) — минимальный набор программных средств, обеспечивающих работу компьютера.

Блок генерирует команды ко всем другим устройствам (оперативной памяти, устройствам ввода - вывода, арифметико-логическому устройству).

Блок управления, функции которого - интерпретировать команды программы.

Данные – это изолированные, несистематизированные факты и сообщения, поступающие как от внешних, так и от внутренних источников в различной форме: цифр, слов, сигналов и др. В соответствии с методом регистрации данные.

Достоверность - отражает уровень полезности информации.

Доступность информации – мера возможности получить ту или иную информацию.

Задача (problet, (task) - проблема, подлежащая решению.

Индуктивный способ - предполагает выдвижение гипотез, декомпозицию сложного объекта, анализ, затем синтез.

Инструментарий технологии программирования — совокупность программ и программных комплексов, обеспечивающих технологию разработки, отладки и внедрения создаваемых программных продуктов.

Информатика – это техническая наука, систематизирующая приемы создания, хранения, воспроизведения, обработки и передачи данных средствами вычислительной техники, а также принципы функционирования этих средств и методы управления ими.

Информация – это обработанные проанализированные и систематизированные в соответствии с поставленными целями данные, ставшие

полезными и значимыми для лиц, которые используют их для принятия решений. Информация – это продукт взаимодействия данных и адекватных им методов.

Метод главного программиста заключается в том, что реализация программного проекта от начала до конца (проектирования, программирования, отладки, выпуска технической документации и инструкций пользователю) осуществляется под руководством главного программиста, который стоит во главе бригады (3— 10 человек).

Метод иерархических диаграмм - в этом методе определяется связь между входными, выходными данными и процессом обработки с помощью иерархической декомпозиции системы (без детализации). По сути, используются три элемента: вход, обработка, выход.

Методо-ориентированный пакет предназначен для решения задачи пользователя одним из нескольких методов, предусмотренных в пакете, причем метод либо назначается пользователем, либо выбирается автоматически на основе анализа входных данных.

Методы, модели и алгоритмы являются интеллектуальной составляющей компонентой информатики. К ним относятся методы преобразования информации и решения различных задач, математические модели изучаемых объектов и процессов, передачи и приема информации, доступа информации и обработки ее.

Модифицируемость. Эта характеристика отражает возможность внесения изменений в ПИ без значительных затрат времени на последующую отладку.

Модуль - это отдельная, функционально законченная программная единица, которая структурно идентифицируется (или оформляется) стандартным образом по отношению к компилятору и по отношению к объединению ее с другими аналогичными единицами и загрузке.

Модульное программирование - каждый модуль тестируется отдельно, затем после кодирования и тестирования происходит их интеграция и тестируется вся система.

Мой компьютер (My Computer) служит для организации доступа ко всем дисководам и другим устройствам данного компьютера, к сетевым устройствам.

Корзина - корзина для ненужных файлов.

Окно - Все приложения Windows работают в прямоугольных областях, которые называются окнами (отсюда и произошло название этой операционной

системы). Существует два основных типа окон - окна приложений и окна документов.

Оперативная память - это место, где компьютер временно хранит данные, требуемые для всех непосредственных (немедленных) действий, которые он выполняет.

Операционные оболочки - специальные программы, предназначенные для облегчения общения пользователя с командами операционной системы. Операционные оболочки имеют текстовый и графический варианты интерфейса конечного пользователя.

Операционные оболочки — специальные программы, предназначенные для облегчения общения пользователя с командами операционной системы.

Основными компонентами, составляющими объекты исследований информатики, являются: технические средства, программные средства и методы, модели и алгоритмы

Отладка ПС – это деятельность, направленная на обнаружение и исправление ошибок в ПС с использованием процессов выполнения его программ.

Пакет прикладных программ - это совокупность совместимых программ для решения определенного класса задач. ППП всегда ориентируется на пользователей определенной квалификации, как в программировании, так и в той области, к которой относятся задачи, решаемые с применением этого ППП.

Панель задач (Taskbar) – это элемент рабочего стола. На ней находится кнопка Пуск (Start). Кнопка Пуск открывает главное меню, содержащее команды доступа к прикладным и служебным программам, системе помощи и находящимся в работе документам.

Полнота информации определяет ее достаточность для принятия решений.

Предметом информатики является информационный ресурс, объединяющий в себе и информацию и знания.

Приложение (application) - программная реализация на компьютере решения задачи.

Проблемно-ориентированные пакеты предназначены для решения групп (последовательностей) задач, использующих общие данные.

Проводник- Проводник Windows специально предназначен для просмотра структуры папок и данных.

Программа (program, routine) - упорядоченная последовательность команд (инструкций) компьютера для решения задачи.

Программирование сверху-вниз - это некоторая многоуровневая дисциплина написания программ. На верхнем уровне исходный алгоритм представляется в виде некоторой иерархической схемы, элементы которой описываются на естественном для данной проблемы языке.

Программное изделие (ПИ) - это программа на носителе данных, являющаяся продуктом промышленного производства.

Программное обеспечение (software) - совокупность программ обработки данных и необходимых для их эксплуатации документов.

Программные средства включают комплекс как отдельных, так и пакетов прикладных программ различной проблемной ориентации, системы и языки программирования, системы управления базами данных, операционные системы, средства обмена информацией в компьютерных сетях.

Программный продукт - комплекс взаимосвязанных программ для решения определенной проблемы (задачи) массового спроса, подготовленный к реализации как любой вид промышленной продукции.

Программотехника (software engineering) — технология разработки, отладки, верификации и внедрения программного обеспечения.

Рабочий стол - это наивысший объект в иерархии объектов. Рабочий стол содержит пиктограммы-значки окон, которые соответствуют программе, документу или папке(folder). Как минимум включает два объекта: Мой компьютер (My Computer) и Recycle Bin (корзина).

Сервисное программное обеспечение - программы и программные комплексы, которые расширяют возможности базового программного обеспечения и организуют более удобную среду работы пользователя программы архивирования данных, которые обеспечивают процесс сжатия информации в файлах с целью уменьшения объема памяти для ее хранения.

Сервисное программное обеспечение — программы и программные комплексы, которые расширяют возможности базового программного обеспечения и организуют более удобную среду работы пользователя.

Сетевые операционные системы - комплекс программ, обеспечивающих обработку, передачу и хранение данных в сети. Сетевая ОС предоставляет пользователям различные виды сетевых служб (управление файлами, электронная почта, процессы управления сетью и др.), поддерживает работу в абонентских системах. Сетевые операционные системы используют архитектуру клиент-сервер или одноранговую архитектуру. Вначале сетевые операционные системы поддерживали лишь локальные вычислительные сети (ЛВС).

Сетевые операционные системы — комплекс программ, обеспечивающих обработку, передачу и хранение данных в сети.

Составление технического задания на программирование - на этом этапе выбирают методы решения задачи; разрабатывают обобщенный алгоритм решения комплекса задач, функциональную структуру алгоритма или состав объектов, определяют требования к комплексу технических средств системы обработки информации, интерфейсу конечного пользователя.

Средства для создания приложений — совокупность языков и систем программирования, а также различные программные комплексы для отладки и поддержки создаваемых программ.

Срок окупаемости (величина, обратная коэффициенту эффективности) - показатель эффективности использования капиталовложений - представляет собой период времени, в течение которого произведенные затраты на ПИ окупаются полученным эффектом.

Строка меню - В этой строке мы можем проводить операции с данным окном, операции с содержимым этого окна, операции с внешним видом окна и его объектов, а также можем вызвать справку по данному окну.

Структурное программирование основано на фиксации для программиста допустимых структур.

Тестирование - это процесс исполнения программ с целью выявления (обнаружения) ошибок.

Технические средства включают ЭВМ с различными периферийными средствами (ввода, преобразования формы представления, регистрации, вывода информации), средства телекоммуникации

Устройства обработки. К устройствам обработки относятся электронно-вычислительные машины или компьютеры.

Утилитарные программы ("программы для себя") предназначены для удовлетворения нужд их разработчиков.

Утилиты - программы, служащие для выполнения вспомогательных с операций обработки данных или обслуживания компьютеров (диагностики, тестирования аппаратных и программных средств, оптимизации использования дискового пространства, восстановления разрушенной на магнитном диске информации и т.п.).

Утилиты — программы, служащие для выполнения вспомогательных с операций обработки данных или обслуживания компьютеров

Центральный процессор - это мозг компьютера, в котором непосредственно и происходит обработка данных и вычисление результатов и который управляет всеми его действиями.

Экономический эффект - результат внедрения какого-либо мероприятия, выраженный в стоимостной форме, в виде-экономии от его осуществления. Так, для организаций (предприятий), использующих ПИ.

Эффективность - выполнение требуемых функций при минимальных затратах ресурсов.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

I.Законы Республики Узбекистан.

1.1. Закон Республики Узбекистан «Об информатизации».// «Народное слово», 2004г., 11февраля.

1.2. Закон Республики Узбекистан «Об электронной коммерции».// «Народное слово», 2004 г., 21-май.

II.Указы Президента Республики Узбекистан.

2.1. Указ Президента Республики Узбекистан «О дальнейшем развитии компьютеризации и внедрении информационно-коммуникационных технологий».// «Народное слово», 2002 г., 1-июня.

III.Постановления Кабинета Министров Республики Узбекистан.

3.1. Постановление Кабинета Министров Республики Узбекистан «О дальнейшем развитии компьютеризации и внедрении информационно-коммуникационных технологий». «Народное слово», 2002 г., 8-июня.

IV.Газеты и журналы.

4.1. Информационные технологии.

4.2. Научно-техническая информация.

4.3. Информатика.

4.4. Экономический вестник Узбекистана.

V.Специализированная литература.

5.1. Информатика. Базовый курс 2-издание. Под редакцией С.В.Симоновича. –СПб.: Питер, 2008. -640 с.

5.2. Бройдо В.Л.Вычислительные системы, сети и телекоммуникации. Учебник для вузов. - СПб. :Питер, 2003,688с.

5.3. Бэйн С.Эффективная работа:Corel Draw 11.

5.4. Додж М.,Стинсон К. Эффективная работа: Excel 2002; перев.с.англ. СПб.:Питер,2002 ,992с.

5.5. Зихерт К.,Ботт Э. Эффективная работа:Windows XP; перев.с англ.- СПб.:Питер,2003, 1072 с.

5.6. Комер Д.Принципы функционирования Интернета; перев.с англ.- СПб.:Питер,2002, 384 с.

5.7. Крёнке Д. Теория и практика построения баз данных; перев.с англ.- СПб.:Питер,2003, 800 с.

5.8. Миллхоллон М.,Мюррей К. Эффективная работа: Word 2002; перев.с англ.- СПб.:Питер,2003, 944 с.

5.9. Немногин.С.А.Turbo-Pascal. Программирование на языке высокого уровня.Учебник для вузов. СПб.:Питер,2003 544с.

5.10. Олифер В.Г., Олифер Н.А.Компьютерные сети. Принципы, технологии, протоколы. Учебник для вузов. СПб.:Питер,2003 864с.

5.11. Павловская Т.А.Паскаль. Программирование на языке высокого уровня.Учебник для вузов. СПб.:Питер,2003 400с.

5.12. Пасько В.Б.Эффективная работа в Интернет. СПб.:Питер,2003 544с.

5.13. Петров М.Н.., Молочков В.П.Компьютерная графика: Учебник для вузов. СПб.:Питер,2003 736с.

5.14. Попов В.Б.Практикум по Интернет - технологиям. Учебный курс- СПб.:Питер,2002 480с.

5.15. Рейнбоу В. Компьютерная графика: Энциклопедия; перев.с англ.- СПб.:Питер,2003, 768 с.

5.16. Тайц А.М., Тайц А.А., Петров М.Н. Эффективная работа: Photoshop 7. СПб.:Питер,2003 768с.

5.17. Феддема Э. Эффективная работа: Access 2002; перев.с англ.- СПб.:Питер,2003, 944 с.

5.18 . Шалин П.А. Энциклопедия Windows XP. - СПб.:Питер,2003,688 с.

5.19. И.Г.Лесничная, И.В.Миссинг, Ю.Д.Романова, В.И.Шестаков. Информатика и информационные технологии. 2-е издание Учебное пособие.- Москва: Эксмо,2007.

VI. Интернет сайты.

6.1. www.search.re.uz – Система поиска информации в Узбекистане.

6.2. www.ictcouncil.gov.uz – сайт Координационного Совета Кабинета Министров РУз. По развитию компьютеризации.

6.3. www.ecsoman.edu.ru – Учебно-методические комплексы по предметам, обучаемых в высших учебных заведениях Российской Федерации.

6.4. www.unitech.uz – Служба телекоммуникаций в Узбекистане.

СОДЕРЖАНИЕ