ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ОРЕНБУРГСКИЙ ГОСУДАРСТВЕННЫЙ АГРАРНЫЙ УНИВЕРСИТЕТ»

Методические рекомендации для самостоятельной работы обучающихся по дисциплине

Б1.Б.16 ИНФОРМАТИКА

Направление подготовки 35.03.06 Агроинженерия

Профиль образовательной программы Электрооборудование и электротехнологии

Форма обучения очная

СОДЕРЖАНИЕ

1. ОРГАНИЗАЦИЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

1.1. Организационно-методические данные дисциплины

п.п.	Наименование тем	Общий объем часов по видам самостоятельной работы (из табл. 5.1 РПД)				
		подготовка курсового проекта (работы)	подготовка реферата/эссе	индивидуальные домашние задания (ИДЗ)	самостоятельное изучение вопросов (СИВ)	подготовка к занятиям (ПкЗ)
	2	3	4	5	6	7
	Представление информации	-	-	-	2	6
	Логические основы построения персонального компьютера	-	-	-	2	2
	Персональный компьютер	-	-	-	2	8
	Программное обеспечение	-	-	-	2	7
	Графические редакторы	-	-	-	5	3
	Базы данных	-	-	-	5	3
	Алгоритмизация и программирование	-	-	-	4	4
	Основы защиты информации	-	-	-	4	2

2. МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ ПО САМОСТОЯТЕЛЬНОМУ ИЗУЧЕНИЮ ВОПРОСОВ

2.1 Качества информации

Репрезентативность - показывает насколько представительна данная информация, то есть насколько информация, полученная в результате выборочных наблюдений соответствует характеристикам сплошных наблюдений. От репрезентативности зависит насколько адекватно информация отражает свойства объекта, явления (того, о чем информация).

Для репрезентативности информации важнейшее значение имеют:

- правильность концепции, на базе которой сформирована информация
- обоснованность отбора существенных признаков и связей отображаемого явления

Содержательность - отражает семантическую емкость информации, равную отношению количества семантической информации в сообщении к объему обрабатываемых данных, то есть:

C=Ic/Vд, где C - содержательность,

Ic - количество семантической информации,

Vд - объем данных

С увеличением содержательности информации растет семантическая пропускная способность информационной системы (т.е. одно и тоже количество семантической информации содержится в меньшем объеме данных)

Достаточность (полнота) - означает, что информация содержит минимальный, но достаточный для принятия правильного решения состав. Понятие полноты информации связано с ее смысловым содержанием (семантикой) и прагматикой. Как неполная (недостаточная), так и избыточная информация снижает эффективность решений, принимаемых ее пользователем.

Доступность восприятию - показывает, насколько легко пользователю извлечь смысл из данной информации. Она обеспечивается выполнением соответствующих процедур получения и преобразования информации к удобной для восприятия пользователя форме (в частности, путем согласования ее семантической формы с тезаурусом пользователя). Например, письмо на испанском для меня имеет нулевую доступность восприятию, поскольку я не владею этим языком, но переведенное на русский язык (согласование с моим тезаурусом) становится вполне доступным.

Актуальность - степень сохранения ценности информации в момент ее использования. Она зависит от динамики изменения характеристик, содержащихся в информации, и от интервала времени, прошедшего с момента возникновения данной информации.

Своевременность - поступление информации не позже заранее назначенного момента времени, согласованного с временем решения поставленной задачи. (Обычно, не позже, чем ее можно будет обработать и использовать для принятия или нахождения решения).

Точность - степень близости получаемой информации к реальному положению вещей (состоянию объекта, процесса, явления).

Для информации, отображаемой цифровым кодом, известны четыре классификационных понятия точности:

- формальная точность, измеряемая значением единицы младшего разряда числа;
- реальная точность, определяемая значением единицы последнего разряда числа, верность которого гарантируется;
- максимальная точность, которую можно получить в конкретных условиях функционирования системы;
- необходимая точность, определяемая функциональным назначением показателя

Достоверность определяется свойством информации отражать реально существующие объекты с необходимой точностью. Она измеряется доверительной вероятностью необходимой точности, то есть вероятностью того, что отображаемое информацией значение параметра отличается от истинного значения этого параметра в пределах необходимой точности.

Устойчивость - способность информации реагировать на изменения исходных данных без нарушения необходимой точности. Устойчивость информации, как и репрезентативность, обусловлена выбранной методикой отбора и формирования информации.

Информационные ресурсы в силу определения понятия "информация", которое связано с уменьшением степени неопределенности в системе "передатчик – приемник" непосредственно у потребителя, могут быть охарактеризованы рядом свойств или признаков, выявляемых и оцениваемых самими потребителями, а поэтому эти оценки весьма субъективными.

Каждый потребитель, исходя из имеющегося него объема знаний и опыта работы с информацией в определенной предметной области, может выделить те свойства информации, которые, по его мнению, определяют смысловую сущность этого ресурса. Последний при этом может иметь различные цели использования, различную степень новизны и другие свои характеристики, которые по-разному могут влиять на процесс принятия решения по выполнению каких-либо действий со стороны потребителя.

При работе с информацией возможны случаи, при которых для разных потребителей один и тот же ресурс может иметь различные выявленные признаки и разную степень оценки каждого из них. Так, к примеру, для одного потребителя предлагаемый информационный ресурс может оказаться совершенно новым, чрезвычайно актуальным и полезным материалом с точки зрения получения новых знаний и их использования на практике. Другой же потребитель с учетом степени его квалификации или в силу сложившихся обстоятельств может не оценить смысл содержащегося в том же материале и не придать должного значения предлагаемой информации.

Кроме того, качественная сторона информации может рассматриваться как с точки зрения потребителя конкретной информации, так и с точки зрения информационной службы, информационной системы или эксперта-специалиста, передающих данный ресурс для дальнейшего его применения. Учитывая это, оценку следует проводить на основе качественных и количественных признаков, характеризующих различные факторы и обстоятельства производства определенного информационного ресурса, запроса потребителя на последний, процессов его передачи, получения и непосредственного использования в научно-теоретической, производственно-практической или социальной деятельности человека.

Следует отметить, что качество одной и той же информации при реализации различных целей или видов деятельности (в технике, метрологии, экономике, социологии и др.) различно. Отличаются и наборы параметров (показателей), и методики определения качества информации в разных предметных областях знаний.

Характеристики качества информации определяют существенные свойства данного объекта, который может находиться в разных стадиях информационных технологий: сбора, хранения, переработки, передачи, получения и использования. Именно с таких позиций и рассмотрим основные характеристики качества.

2.2Понятие переключательной и коммутационной схемы. Примеры схем.

В вычислительных и других автоматических устройствах широко применяются электрические схемы, содержащие множество переключательных элементов: реле, выключателей и т. п. При разработке таких схем с успехом может быть использован аппарат алгебры логики.

Переключательная схема– схематическое изображение некоторого устройства, состоящего из переключателей и соединяющих их проводников, а также входов и выходов, на которые подается и с которых снимается электрический сигнал.

Каждый переключатель имеет только два состояния: замкнутое и разомкнутое. Переключателю X поставим в соответствие логическую переменную x, которая принимает значение 1 только в том случае, когда переключатель X замкнут и схема проводит ток; если же переключатель разомкнут, то переменная x равна нолю. При этом два переключателя X и X связаны таким образом, что когда X замкнут, то X разомкнут, и наоборот. Следовательно, если переключателю X поставлена в соответствие логическая переменная x, то переключателю X должна соответствовать переменная x.

Всей переключательной схеме также можно поставить в соответствие логическую переменную, равную единице, если схема проводит ток, и равную нолю — если не проводит. Эта переменная является функцией от переменных, соответствующих всем переключателям схемы, и называется функцией проводимости.

Рассмотрим функции проводимости F некоторых переключательных схем:

Схема не содержит переключателей и проводит ток всегда, следовательно, F – 1;

Схема содержит один постоянно разомкнутый контакт, следовательно, F – 0;

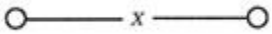


Схема проводит ток, когда переключатель x замкнут, и не проводит, когда x разомкнут, следовательно, $F(x) = x$;

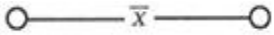


Схема проводит ток, когда переключатель x разомкнут, и не проводит, когда x замкнут, следовательно, $F(x) = \overline{x}$;

Схема проводит ток, когда оба переключателя замкнуты, следовательно, $F(x) = x \cdot y$;

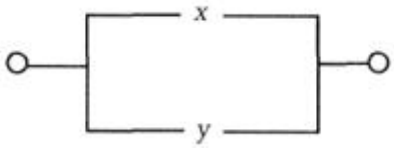


Схема проводит ток, когда хотя бы один из переключателей замкнут, следовательно, $F(x) = x \vee y$.

При рассмотрении переключательных схем решают, как правило, одну из основных задач: синтез или анализ схемы.

Синтез переключательной схемы по заданным условиям ее работы сводится к следующим трем этапам:

· составление функции проводимости по заданным условиям;

· упрощение этой функции;

· построение соответствующей схемы.

Анализ схемы характеризуется следующими этапами:

определение значений функции проводимости при всех возможных наборах входящих в эту функцию переменных;

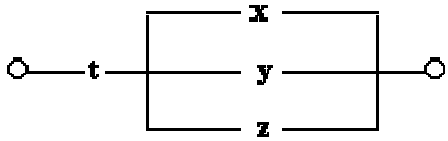
получение упрощенной формулы.

Рассмотрим примеры решения задач синтеза и анализа несложных переключательных схем.

Примеры.

1. Построим схему, содержащую 4 переключателя x, y, z и t, такую, чтобы она проводила ток тогда и только тогда, когда замкнут контакт переключателя t и какой-нибудь из остальных трёх контактов.

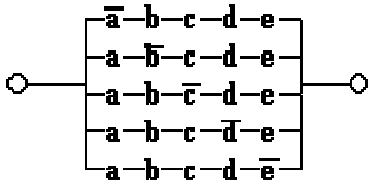
Решение. В этом случае можно обойтись без построения таблицы истинности. Очевидно, что функция проводимости имеет вид **F(x, y, z, t) = t · (x v y v z)**, а схема выглядит так:



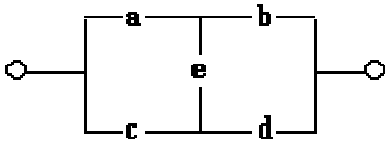
2. Построим схему с пятью переключателями, которая проводит ток в том и только в том случае, когда замкнуты ровно четыре из этих переключателей.

Решение : $F(a,b,c,d,e) = \bar{a}\cdot b\cdot c\cdot d\cdot e \vee a\cdot\bar{b}\cdot c\cdot d\cdot e \vee a\cdot b\cdot\bar{c}\cdot d\cdot e \vee a\cdot b\cdot c\cdot\bar{d}\cdot e \vee a\cdot b\cdot c\cdot d\cdot\bar{e}$

Схема имеет вид:



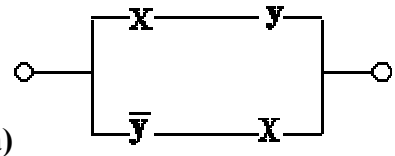
3. Найдем функцию проводимости схемы:



Решение. Имеется четыре возможных пути прохождения тока при замкнутых переключателях a, b, c, d, e : через переключатели a, b; через переключатели a, e, d; через переключатели c, d и через переключатели c, e, b. Функция проводимости

$$F(a, b, c, d, e) = a\cdot b \vee a\cdot e\cdot d \vee c\cdot d \vee c\cdot e\cdot b.$$

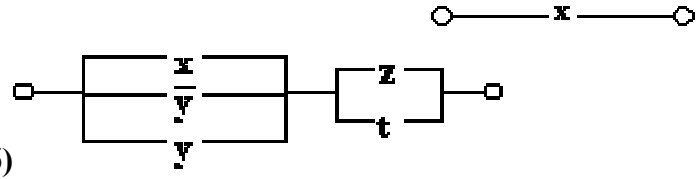
4. Упростить переключательные схемы:



а)

Решение: $F(x;y) = x\cdot y \vee \bar{y}\cdot x = x\cdot(y \vee \bar{y}) = x\cdot 1 = x$.

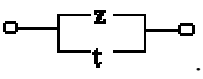
Упрощенная схема:



б)

Решение: $F(x;y;z;t) = (x \vee \bar{y} \vee y)\cdot(z \vee t) = 1\cdot(z \vee t) = z \vee t$.

Упрощенная схема:



.

2.3Дисковая память. Флэш-память

Носителями информации являются поверхности гибких и жестких дисков, в качестве немагнитных основ которых используются соответственно майлар (как и в магнитных лентах) и алюминиевые (в ряде случаев стеклянные) круги (диски). Стеклянные диски являются менее критичными к температурным изменениям и позволяют увеличить плотность записи информации. В настоящее время наиболее широкое распространение получили диски с напыленным магнитным слоем, а точнее, с металлической пленкой (например, кобальт).

Перед осуществлением записи на магнитный диск он должен быть специальным образом инициализирован – отформатирован. В результате форматирования на поверхности образуются концентрические окружности (синхронизирующие метки диска), называемые **дорожками** (track). Количество дорожек зависит от типа диска. Дорожки разбиваются на участки фиксированной длины, называемые **секторами.** Количество секторов на дорожке определяется типом и **форматом** диска, и они в основном одинаковы для всех дорожек. IBM PC-совместимые ПК могут работать с несколькими размерами секторов от 128 до 1024 байт. Стандартным сектором считается сектор из 512 байт. Данные любого размера (разрядности) размещаются в секторах с фиксированным размером, а дисковые операции записи и считывания производятся с целыми секторами.

Дорожки и сектора нумеруются с нуля, начиная с внешнего края диска, при этом сектор с нулевым номером на каждой дорожке резервируется для системных целей. Диски имеют две стороны. Так как накопители на жестких дисках могут состоять из нескольких дисков (стопка), то совокупность всех дорожек, по одной на каждой стороне с одинаковыми номерами, образует **цилиндр** с номером соответствующей дорожки.

Флеш-память (англ. *flashmemory*) - разновидность полупроводниковой технологии электрически перепрограммируемой памяти (EEPROM). Это же слово используется в электронной схемотехнике для обозначения технологически законченных решений постоянных запоминающих устройств в виде микросхем на базе этой полупроводниковой технологии. В быту это словосочетание закрепилось за широким классом твердотельных устройств хранения информации.

Благодаря компактности, дешевизне, механической прочности, большому объёму, скорости работы и низкому энергопотреблению, флеш-память широко используется в цифровых портативных устройствах и носителях информации. Серьёзным недостатком данной технологии является ограниченный срок эксплуатации носителей, а также чувствительность к электростатическому разряду.

Принцип работы полупроводниковой технологии флеш-памяти основан на изменении и регистрации электрического заряда в изолированной области («кармане») полупроводниковой структуры.

Изменение заряда («запись» и «стирание») производится приложением между затвором и истоком большого потенциала, чтобы напряженность электрического поля в тонком диэлектрике между каналом транзистора и карманом оказалась достаточна для возникновения туннельного эффекта. Для усиления эффекта туннелирования электронов в карман при записи применяется небольшое ускорение электронов путём пропускания тока через канал полевого транзистора (явление инжекции горячих носителей).

Чтение выполняется полевым транзистором, для которого карман выполняет функцию затвора. Потенциал плавающего затвора изменяет пороговые характеристики транзистора, что и регистрируется цепями чтения.

Эта конструкция снабжается элементами, которые позволяют ей работать в большом массиве таких же ячеек.

2.4 Файловые менеджеры, утилиты и архиваторы.

Файловый менеджер (англ. filemannager) - компьютерная программа, предоставляющая интерфейс пользователя для работы с файловой системой и файлами. Файловый менеджер позволяет выполнять наиболее частые операции - копирования, переноса, удаления, редактирования текстовых файлов, гибкого запуска программ для работы с этими файлами…

Помимо основных функций, многие файловые менеджеры включают ряд дополнительных возможностей, например, таких как работа с сетью (через FTP, NFS и т. п.), резервное копирование, управление принтерами и пр.

Существует два вида файловых менеджеров - навигационные и ортодоксальные. Основное отличие - в последних имеется две панели, реализована соответствующая модель работы.

Наиболееизвестныеортодоксальныефайловыеменеджеры: Norton Commander, Dos Navigator, Volkov Commander, PIE Commander, FAR Manager, Total Commander, POSIX (Linux, BSD ит. д.), Midnight Commander, Krusader, GNOME Commander.

Навигационные файловые менеджеры: проводник Windows (англ. WindowsExplorer) -- встроен в Windows, Mac OS X, Finder, PathFinder, POSIX (Linux, BSD и т.д.), Konqueror -- поставляется с KDE, Nautilus (файловый менеджер) -- поставляется с GNOME

Современный файловый менеджер должен: обеспечивать удобную возможность работы с файлами, копировать, удалять, перемещать, создавать редактировать текстовые файлы, запускать внешние программы для работы с разными типами файлов, позволять с легкостью и удобством работать как клавиатурой, так и с помощью мышки, поддерживать технологию плагинов и настройку цветовых схем.

В состав базового дистрибутива файлового менеджера должны входить ряд модулей, плагинов, которые непосредственно связаны с работой с файлами:

1. Просмотр и редактирование текстовых файлов, подцветка синтаксиса, поддержка разных кодировок (включая Unicode)

2. Поиск и замена по множеству файлов, множественное переименование файлов, просмотр картинок, работа с архивами.

Утилиты

Термин «утилита» происходит от английского слова utility – полезный.

Утилиты можно рассматривать как «развитые» внешние команды операционной системы, имеющие хорошо организованный графический интерфейс, ориентированный на работу с мышью. Они служат для расширения возможностей ОС (предоставление различного сервиса), а их функции носят специализированный характер.

Системные утилиты – это обслуживающие программы вспомогательного назначения.

Утилиты дополняют возможности ОС, обеспечивая выполнение различных вспомогательных действий. Обычно некоторое количество утилит поставляется в составе соответствующей ОС, но немало утилит создано независимыми разработчиками и поставляется отдельно от ОС.

Утилиты часто используют низкоуровневые механизмы функционирования ОС, поэтому они могут работать только в тех ОС, на которые рассчитаны. Т.к. применение утилит в «чужой» для них ОС может привести, например, к уничтожению данных (это относится, прежде всего, к программам обслуживания дисков).

Примером может служить комплект стандартных утилит, встроенных в MSWindows(группа «Служебные»). Туда включен стандартный набор приложений, обеспечивающих выполнение следующих функций:

- проверка и восстановление сбойных дисков;
- оптимизация расположения файлов на диске (дефрагментация диска);
- получение информации о компьютере;
- восстановление файлов на диске;

- очистка диска и др.

К утилитам относят и два блока приложений: архиваторы и антивирусные пакеты.

Архиваторы

Архиватор (упаковщик) – программа, позволяющая за счет применения специальных методов сжатия информации создавать копии файлов меньшего размера, а также объединять копии нескольких файлов в один архивный файл.

Все существующие на сегодняшний день архиваторы можно разделить на три группы, которые можно условно назвать файловые, программные и дисковые.

1. Файловые архиваторы – позволяют упаковывать один или несколько файлов в единый архив. Размер архива, как правило, меньше чем суммарный размер исходных файлов. Воспользоваться архивными данными и программами пока они находятся в архиве нельзя. Для распаковки архива требуется разархиватор, который совмещен с архиватором в одной программе.

Кроме этого практически в любой программе архиваторе имеется возможность создания самораспаковывающихся файлов, который имеет расширение exe. Он содержит кроме упакованных данных разархивирующий модуль. (Rar, Zip, Ice, Ain)

2. Программные архиваторы – позволяют упаковать за один прием один единственный файл – выполняемую программу exe типа, которая при запуске самораспаковывается в оперативной памяти и тут же начинает работу. Программа становиться в два раза меньше и при этом сохраняет работоспособность. (LZEXE – UNLZEXE, EXEPACK — UPACKEXE)

3. Дисковый архиватор – представляет собой резидентный драйвер, который незаметно для пользователя архивирует любую записываемую на диск информацию и распаковывает ее обратно при чтении. При этом на диске создается огромный архив, который отображается как еще один логический раздел винчестера.

Принцип работы архиватора состоит в том, что программа ищет повторяющиеся фрагменты в файлах, после чего все найденные повторения заменяются ссылками на первые фрагменты. При этом, записывая информацию подобным образом, архиватор обязательно должен запомнить, что и откуда он «отрезал», что и куда он «приклеил», и что за чем стоит в этой очереди.

Естественно, для различных типов файлов степень архивации будет разная. Файлы, содержащие текстовые данные, сжимаются максимально. Файлы-программы имеют очень маленькую степень сжатия из-за малого количества повторяющихся значений. Графические файлы, содержащие простые графические объекты черно-белого цвета, можно сжать в пять-шесть раз. Графические файлы типа TIFF Compressed не сжимаются вообще, потому что сжимаются при создании.

Хотя изначально сложно определить процент сжатия, но все же использование архиватора дает положительный результат, если вы пытаетесь экономно расходовать место на диске.

Самыми популярными программами архивирования, пожалуй, можно считать ZIP, RAR, ARJ.Основным недостатком архивов является невозможность прямого доступа к данным. Их сначала необходимо извлечь из архива или распаковать. Операция распаковки, впрочем, как и упаковки, требует некоторых системных ресурсов. Это не мгновенная операция. Поэтому архивы в основном применяют со сравнительно редко используемыми данными. Например, для хранения резервных копий или установочных файлов.

2.5 Векторные и растровые форматы

Формат TIFF

TIFF (англ. TaggedImageFileFormat) — формат хранения растровых графических изображений. TIFF стал популярным форматом для хранения изображений с большой

глубиной цвета. Он используется при сканировании, отправке факсов, распознавании текста, в полиграфии, широко поддерживается графическими приложениями.

Структура формата гибкая и позволяет сохранять изображения в режиме цветов с палитрой, а также в различных цветовых пространствах.

TIFF является теговым форматом и в нём используются основные, расширенные и специальныетеги.Основные теги составляют ядро формата и должны поддерживаться всеми продуктами, реализующими формат TIFF в соответствии со спецификацией. Поддержка расширенных тегов, в отличие от основных необязательна.

Формат JPEG

JPEG (англ.JointPhotographicExpertsGroup, по названию организации-разработчика) — один из популярных графических форматов, применяемый для хранения фотоизображений. Файлы, содержащие данные JPEG, обычно имеют расширения .jpeg, .jfif, .jpg, .JPG, или .JPE. Алгоритм JPEG позволяет сжимать изображение как с потерями, так и без потерь.

Алгоритм JPEG в наибольшей степени ***пригоден*** для сжатия фотографий и картин, содержащих реалистичные сцены с плавными переходами яркости и цвета. Наибольшее распространение JPEG получил в цифровой фотографии и для хранения и передачи изображений с использованием сети Интернет.

Формат PDF

PDF (англ. PortableDocumentFormat) — кроссплатформенный формат электронных документов, созданный фирмой AdobeSystems с использованием ряда возможностей языка PostScript. Чаще всего PDF-файл является комбинацией текста с растровой и векторной графикой, реже — текста с формами, JavaScript'ом, 3D-графикой и другими типами элементов. В первую очередь***предназначен*** для представления в электронном виде полиграфической продукции, — значительное количество современного профессионального печатного оборудования может обрабатывать PDF непосредственно. Для просмотра можно использовать официальную бесплатную программу AdobeReader, а также программы сторонних разработчиков.

Формат CALS

Растровый **формат CALS** (англ. ComputerAidedAcquisitionandLogisticsSupport) стардарт, разработанный подразделением министерства обороны США для стандартизации обмена графическими данными в электронном виде, особенно в областях технической графики, CAD/CAM и приложений обработки изображений.

CALS - хорошо документированный, хотя и громоздкий.

Формат BMP

BMP (от англ. BitmapPicture) — формат хранения растровых изображений, разработанный компанией Microsoft. С форматом BMP работает огромное количество программ, так как его поддержка интегрирована в операционные системы Windows и OS/2. Файлы формата BMP могут иметь расширения .bmp, .dibи .rle.

Формат PCX

PCX (PCExchange) — стандарт представления графической информации, не столь популярный аналог BMP, хотя поддерживается специфическими графическими редакторами, такими как AdobePhotoshop, CorelDraw, GIMP и др. В настоящее время практически вытеснен форматами, которые поддерживают лучшее сжатие: GIF, JPEG и PNG.**Тип формата** — растровый.

Формат PNG

PNG (англ. portablenetworkgraphics) — растровый формат хранения графической информации, использующий сжатие без потерь.

Область применения

Формат PNG спроектирован для замены устаревшего и более простого формата GIF, а также, в некоторой степени, для замены значительно более сложного формата TIFF.

Формат PNG**позиционируется** прежде всего **для использования в Интернете** и редактирования графики.

Формат SunRaster

Формат изрбражений SunRaster это родной растровый формат платформ SunMicrosystems использующих операционную систему SunOS. Этот формат поддерживает черно-белые, полутоновые и цветные растровые данные произвольной глубины цвета. Поддерживается также использование цветовых карт и простой компрессии данных Run-Length. Обычно большинство изображений в операционной системе SunOS представлены в формате SunRaster. Также этот формат поддерживается большинством программ работы с изображениями под UNIX.

2.6 Реляционная модель базы данных

Почти все современные системы основаны на **реляционной** (relational) модели управления базами данных. Название **реляционная** связано с тем, что каждая запись в такой базе данных содержит информацию, относящуюся только к одному конкретному объекту.

В **реляционной** СУБД все обрабатываемые данные представляются в виде плоских таблиц. Информация об объектах определенного вида представляется в табличном виде: в столбцах таблицы сосредоточены различные атрибуты объектов, а строки предназначены для сведения описаний всех атрибутов к отдельным экземплярам объектов.

Модель, созданная на этапе инфологического моделирования, в наибольшей степени удовлетворяет принципам реляционности. Однако для приведения этой модели к реляционной необходимо выполнить процедуру, называемую **нормализацией**.

Теория нормализации оперирует с пятью **нормальными формами**. Эти формы предназначены для уменьшения избыточности информации, поэтому каждая последующая нормальная форма должна удовлетворять требованиям предыдущей и некоторым дополнительным условиям. При практическом проектировании баз данных четвертая и пятая формы, как правило, не используются. Мы ограничились рассмотрением первых четырех нормальных форм.

Введем понятия, необходимые для понимания процесса приведения модели к реляционной схеме.

Отношение - абстракция описываемого объекта как совокупность его свойств. Проводя инфологический этап проектирования, мы говорили об абстракции объектов и приписывали им некоторые свойства. Теперь же, проводя концептуальное проектирование, мы переходим к следующему уровню абстракции. На данном этапе объектов, как таковых, уже не существует. Мы оперируем совокупностью свойств, которые и определяют объект.

Экземпляр отношения - совокупность значений свойств конкретного объекта.

Первичный ключ - идентифицирующая совокупность атрибутов, т.е. значение этих атрибутов уникально в данном отношении. Не существует двух экземпляров отношения содержащих одинаковые значения в первичном ключе.

Простой атрибут - атрибут, значения которого неделимы.

Сложный атрибут - атрибут, значением которого является совокупность значений нескольких различных свойств объекта или несколько значений одного свойства.

Требования к реляционным моделям

Рациональные варианты концептуальной схемы базы данных должны удовлетворять третьей нормальной форме, а также следующим требованиям:

- Выбранный перечень отношений должен быть минимален. Отношение используется, если только его необходимость обусловлена задачами.
- Выбранный перечень атрибутов должен быть минимален. Атрибут включается в отношение только в том случае, если он будет использоваться.

- Первичный ключ отношения должен быть минимальным. То есть невозможно исключить ни один атрибут из идентифицирующей совокупности атрибутов, не нарушив при этом однозначной идентификации.

При выполнении операций над данными не должно возникать трудностей

2.7 Программы циклической структуры

При решении подавляющего большинства задач (в том числе и весьма несложных) в программе практически невозможно задать в явном виде все операции, которые необходимо выполнить. В самом деле, пусть необходимо вычислить сумму первых n членов гармонического ряда: Y= 1+ 1/2+ 1/3 + …+ 1/ n

Если значение n не фиксируется, а является исходным данным, вводимым в процессе выполнения программы (и даже константой, описанной в программе), то аналогичный оператор присваивания записать невозможно. Ибо запись вида Y:= 1+1/2+1/3+…+1/ n в языках программирования недопустима.

Для устранения возникающих трудностей служат операторы цикла. Они позволяют повторять выполнение отдельных частей программы. Можно выделить четыре оператора цикла:

- простой арифметический оператор цикла (цикл с параметром с шагом 1),
- сложный арифметический оператор цикла (цикл с параметром произвольного шага),
- итерационный оператор цикла с предусловием,
- итерационный оператор цикла с постусловием.

Простой арифметический оператор цикла Паскаля (цикл с параметром)

Вернемся к рассмотренной выше задаче вычисления суммы первых n членов гармонического ряда, правила которой невозможно задать в виде арифметического выражения, если значение n заранее не фиксировано.

На самом деле вычисление этой суммы можно осуществить по очень простому и компактному алгоритму: предварительно положим y=0 (с помощью оператора присваивания y:=0), а затем выполним оператор присваивания y:= y+1/ i для последовательных значений i= 1,2,…, n. При каждом очередном выполнении этого оператора к текущему значению y будет прибавляться очередное слагаемое. Как видно, в этом случае процесс вычислений будет носить циклический характер: оператор y:= y+1/i должен выполняться многократно, т.е. циклически, при различных значениях i.

Этот пример циклического вычислительного процесса является весьма типичным; его характерные особенности состоят в том, что

- число повторений цикла известно к началу его выполнения (в данном случае оно равно значению n, которое предполагается заданным к этому времени);
- управление циклом осуществляется с помощью переменной порядкового типа, которая в этом циклическом процессе принимает последовательные значения от заданного начального до заданного конечного значений (в нашем случае – это целочисленная переменная i, принимающая последовательные значения от 1 до n).

Для компактного задания подобного рода вычислительных процессов и служит оператор цикла с параметром. Чаще всего используется следующий вид этого оператора В Паскале:

ForV:=E1 toE2 doS,

где for (для), to (увеличиваясь к) и do (выполнять, делать) – служебные слова, V – переменная порядкового типа, называемая параметром цикла, E1 и E2 – выражения того же типа, что и параметр цикла, S – оператор, который и выполняется многократно в цикле, называемый телом цикла.

Заметим, что в Паскале после do должен стоять один оператор, если необходимо выполнить несколько действий, то они должны быть объединены в один составной оператор путем заключения в операторные скобки.

Этот оператор цикла Паскаля предусматривает присваивание параметру цикла V последовательных значений от начального значения, равного значению выражения E1, до конечного значения, равного значению выражения E2, т.е. при каждом повторении выполняется оператор присваивания V:=succ(V), и выполнение оператора S при каждом значении параметра цикла V. При этом значения выражений E1 и E2 вычисляются один раз, при входе в оператор цикла, а значение параметра цикла V не должно изменяться в результате выполнения оператора S. Если заданное конечное значение меньше начального значения (что допустимо), то оператор S не выполняется ни разу.

В Паскале считается, что при нормальном завершении выполнения оператора цикла значение параметра цикла не определено.

С использованием оператора цикла с параметром алгоритм вычисления суммы первых n членов гармонического ряда может быть задан следующим образом:

Пример кода программы для суммирования первых n членов гармонического ряда

Readln(n); Y:= 0;

For i:= 1 to n do y:= y+1/i;

В некоторых случаях бывает удобно, чтобы параметр цикла Паскаля принимал последовательные, но не возрастающие, а убывающие значения. Для таких случаев в Паскале предусмотрен оператор цикла с параметром следующего вида:

For V:= E1 downto E2 do S,

где downto (уменьшаясь к) – служебное слово, а все остальные слова и выражения имеют прежний смысл. Изменение параметра цикла от большего значения к меньшему происходит при выполнении присваивания V:=pred(V). Заметим, что начальное значение может быть меньше конечного значения. В этом случае оператор S не выполнится ни разу. Значение параметра цикла по завершении выполнения такого цикла так же считается неопределенным.

Следует запомнить и то, что для обоих вариантов записи цикла с параметром справедливо: если начальное и конечное значения равны, то тело цикла (оператор S) выполнится один раз.

Арифметический оператор цикла Паскаля с произвольным шагом

Естественным усложнением простого арифметического цикла Паскаля, является цикл, в котором параметр цикла изменяется не на 1, а на произвольную величину – шаг приращения. При этом в процессе выполнения цикла шаг изменяется по заданному закону. Стандартные операторы для реализации такого цикла есть в Форте, в других языках их приходится организовывать из простейшего арифметического цикла.

Итерационные операторы цикла Паскаля

Итерационные циклы отличаются от циклов с параметром тем, что в них заранее неизвестно число повторений.

Рассмотрим теперь математическую задачу. Пусть нам необходимо вычислить сумму первых членов гармонического ряда, удовлетворяющих условию 1/i>= e, где 0< e<1, а i=1,2,3…...Эту задачу можно решить по следующему алгоритму: положить предварительно y=0 и i=0, а затем в цикле увеличивать i на 1, к значению y добавлять очередное слагаемое 1/ i до тех пор, пока текущее значение 1/ i впервые окажется больше заданного значения 0< e<1.

Очевидно, что число повторений этого цикла заранее не известно. В подобного рода случаях мы можем лишь сформулировать условие, при выполнении которого процесс добавления к сумме очередного слагаемого должен завершиться.

Для задания таких вычислительных процессов и служит оператор цикла Паскаля с постусловием. Этот оператор имеет вид:

RepeatS1; S2;…; SiuntilB,

где repeat (повторять) и until (до) – служебные слова, через Si обозначен любой оператор Паскаля, а через B – логическое выражение.

При выполнении этого оператора цикла последовательность операторов, находящихся между словами repeat и until, выполнится один или более раз. Этот процесс завершается, когда после очередного выполнения заданной последовательности операторов логическое выражение B примет (впервые) значение true. Таким образом, с помощью логического выражения B задается условие завершения выполнения оператора цикла. Поскольку в данном случае проверка условия производится после выполнения последовательности операторов (тела цикла), этот оператор цикла и называется оператором цикла с постусловием.

С использованием этого вида оператора цикла Паскаля задача о суммировании первых членов гармонического ряда, удовлетворяющих заданному условию, может быть реализована следующим образом:

Заметим, что оператор цикла с постусловием является более общим, чем оператор цикла с параметром — любой циклический процесс, задаваемый с помощью цикла с параметром можно представить в виде цикла с постусловием. Обратное утверждение неверно. Например, задача о суммировании первых n членов гармонического ряда, рассмотренная ранее, с оператором цикла с постусловием будет выглядеть так:

Оператор цикла Паскаля с предусловием

В случае оператора цикла Паскаля с постусловием входящая в него последовательность операторов заведомо будет выполняться хотя бы один раз. Между тем довольно часто встречаются такие циклические процессы, когда число повторений цикла тоже неизвестно заранее, но при некоторых значениях исходных данных предусмотренные в цикле действия вообще не должны выполняться, и даже однократное выполнение этих действий может привести к неверным или неопределенным результатам.

Пусть, например, дано вещественное число M. Требуется найти наименьшее целое неотрицательное число k, при котором 3 k> M. Эту задачу можно решить по следующему алгоритму: предварительно положить y=1 и k=0; затем в цикле домножать значение y на 3 и увеличивать значение k на 1 до тех пор, пока текущее значение y впервые окажется больше значения M. На первый взгляд, здесь можно воспользоваться оператором цикла с постусловием:

Для задания подобного рода вычислительных процессов, когда число повторений цикла заранее неизвестно и действия, предусмотренные в цикле, могут вообще не выполняться, и служит оператор цикла с предусловием. Этот оператор цикла имеет в Паскале следующий вид:

While B do S,

где while (пока), do (делать, выполнять) – служебные слова, B – логическое выражение, S – оператор. Здесь оператор S выполняется ноль или более раз, но перед каждым очередным его выполнением вычисляется значение выражения B, и оператор S выполняется только в том случае, когда значение выражения B true. Выполнение оператора цикла завершается, когда выражение B впервые принимает значение false. Если это значение выражение B принимает при первом же его вычислении, то оператор S не выполнится ни разу.

Оператор цикла Паскаля с предусловием можно считать наиболее универсальным – с использованием таких операторов можно задать и циклические процессы, определяемые операторами цикла с параметром и постусловием.

Отметим отличия и особенности хорошего стиля работы с рассмотренными циклическими операторами.

Цикл с предусловием While (пока условие истинно)	Цикл с постусловием Repeat (до истинности условия)
1. До начала цикла должны быть сделаны начальные установки переменных, управляющих условием цикла, для корректного входа в цикл	
2. В теле цикла должны присутствовать операторы, изменяющие	

переменные условия так, чтобы цикл через некоторое число итераций завершился	
3. Цикл работает пока условие истинно (пока True)	3. Цикл работает пока условие ложно (пока False)
4. Цикл завершается, когда условие становится ложным (до False)	4. Цикл завершается, когда условие становится истинным (до True)
5. Цикл может не выполниться ни разу, если исходное значение условия при входе в цикл False	5. Цикл обязательно выполнится как минимум один раз
6. Если в теле цикла требуется выполнить более одного оператора, то необходимо использовать составной оператор	6. Независимо от количества операторов в теле цикла, использование составного оператора не требуется
Цикл со счетчиком (с параметром) For	
• Начальная установка переменной счетчика цикла до заголовка не требуется	
• Изменение в теле цикла значений переменных, стоящих в заголовке не допускается	
• Количество итераций цикла неизменно и точно определяется значениями нижней и верхней границ и шага приращения	
• Нормальный ход работы цикла может быть нарушен оператором goto или процедурами Break и Continue	
• Цикл может не выполниться ни разу, если шаг цикла будет изменять значение счетчика от нижней границы в направлении, противоположном верхней границе	

Оператор, который выполняется в цикле, сам может быть циклом. Это относится ко всем видам циклов. В результате мы получаем вложенные циклы. Механизм работы вложенных циклов удобнее всего рассмотреть на примере вложенных циклов с параметром. Пусть нам нужно описать работу электронных часов, начиная с момента времени 0 часов, 0 минут, 0 секунд. Значение минут станет равным 1 только после того, как секунды «пробегут» все последовательные значения от 0 до 59. Часы изменят свое значение на 1 только после того, как минуты «пробегут» все последовательные значения от 0 до 59. Таким образом, вывод всех значений времени от начала суток до конца суток может быть представлен следующим фрагментом программы:

```
For h:=0 to 23 do
For m:=0 to 59 do
For s:=0 to 59 do
Writeln(h,':',m,':',s);
```

Для удобства реализации циклических структур на Паскале в последних версиях языка введены операторы break и continue, применяемые внутри циклов. Они расширяют возможности использования циклов и улучшают структуру программы.

В процессе выполнения тела цикла до полного завершения цикла могут возникнуть дополнительные условия, требующие завершения цикла. В этом случае цикл может быть прекращен оператором break.

В ходе выполнения цикла может возникнуть условие, при котором необходимо пропустить все или некоторые действия, предусмотренные в цикле, не прекращая работу цикла совсем. Для этого используется оператор continue, который передает управление в ту точку программы, где проверяется условие продолжения или прекращения цикла.

2.8 Криптографическая защита информации

Проблема защиты информации путем ее преобразования, исключающего ее прочтение посторонним лицом, волновала человеческий ум с давних времен. История криптографии - ровесница истории человеческого языка. Более того, первоначально письменность сама по себе была криптографической системой, так как в древних обществах ею владели только избранные. Священные книги Древнего Египта, Древней Индии тому примеры.

Криптографические методы защиты информации – это специальные методы шифрования, кодирования или иного преобразования информации, в результате которого ее содержание становится недоступным без предъявления ключа криптограммы и обратного преобразования. Криптографический метод защиты, безусловно, самый надежный метод защиты, так как охраняется непосредственно сама информация, а не доступ к ней (например, зашифрованный файл нельзя прочесть даже в случае кражи носителя). Данный метод защиты реализуется в виде программ или пакетов программ.

Современная криптография включает в себя четыре крупных раздела:

1. *Симметричные криптосистемы*. В симметричных криптосистемах и для шифрования, и для дешифрования используется один и тот же ключ. (Шифрование - преобразовательный процесс: исходный текст, который носит также название открытого текста, заменяется шифрованным текстом, дешифрование - обратный шифрованию процесс. На основе ключа шифрованный текст преобразуется в исходный);
2. *Криптосистемы с открытым ключом*. В системах с открытым ключом используются два ключа - открытый и закрытый, которые математически связаны друг с другом. Информация шифруется с помощью открытого ключа, который доступен всем желающим, а расшифровывается с помощью закрытого ключа, известного только получателю сообщения. (Ключ - информация, необходимая для беспрепятственного шифрования и дешифрования текстов);
3. *Электронная подпись*. Системой электронной подписи. называется присоединяемое к тексту его криптографическое преобразование, которое позволяет при получении текста другим пользователем проверить авторство и подлинность сообщения.
4. *Управление ключами*. Это процесс системы обработки информации, содержанием которых является составление и распределение ключей между пользователями.

Основные направления использования криптографических методов - передача конфиденциальной информации по каналам связи (например, электронная почта), установление подлинности передаваемых сообщений, хранение информации (документов, баз данных) на носителях в зашифрованном виде.

Хэш-функции

В самых различных отраслях информационных технологий находят свое применение хэш-функции. Они предназначены для того, чтобы, с одной стороны, значительно упростить обмен данными между пользователями и обработку файлов, используемых в тех или иных целях, с другой - оптимизировать алгоритмы обеспечения контроля доступа к соответствующим ресурсам. Хэш-функция - один из ключевых инструментов обеспечения парольной защиты данных, а также организации обмена документов, подписанных с помощью ЭЦП. Существует большое количество стандартов, посредством которых может осуществляться кэширование файлов. Многие из них разработаны российскими специалистами. В каких разновидностях могут быть представлены хэш-функции? Каковы основные механизмы их практического применения?

Для начала исследуем понятие хэш-функции. Под данным термином принято понимать алгоритм преобразования некоторого объема информации в более короткую последовательность символов посредством математических методов. Практическую значимость хэш-функции можно проследить в самых разных областях. Так, их можно задействовать при проверке файлов и программ на предмет целостности. Также

криптографические хеш-функции задействуются в алгоритмах шифрования. Характеристики Рассмотрим ключевые характеристики исследуемых алгоритмов. В числе таковых: наличие внутренних алгоритмов преобразования данных исходной длины в более короткую последовательность символов; открытость для криптографической проверки; наличие алгоритмов, позволяющих надежно шифровать изначальные данные; адаптированность к расшифровке при задействовании небольших вычислительных мощностей. В числе иных важнейших свойств хэш-функции:

Требования к хэш-функциям

Существует ряд требований к хэш-функциям, предназначенным для практического задействования в той или иной области. Во-первых, соответствующий алгоритм должен характеризоваться чувствительностью к изменениям во внутренней структуре хешируемых документов. То есть в хэш-функции должны распознаваться, если речь идет о текстовом файле, перестановки абзацев, переносы. С одной стороны, содержимое документа не меняется, с другой — корректируется его структура, и этот процесс должен распознаваться в ходе хеширования. Во-вторых, рассматриваемый алгоритм должен преобразовывать данные так, чтобы обратная операция (превращение хэша в изначальный документ) была на практике невозможна. В-третьих, хэш-функция должна предполагать задействование таких алгоритмов, которые практически исключают вероятность формирования одинаковой последовательности символов в виде хэш, иными словами - появления так называемых коллизий. Их сущность мы рассмотрим чуть позже.

В числе главных требований к рассматриваемым алгоритмам - обеспечение однонаправленности шифрования. Человек, имеющий в распоряжении только хэш, практически не должен иметь возможности получить на его основе исходный документ. В какой структуре может быть представлена используемая в подобных целях хеш-функция? Пример ее составления может быть таким: H (hash, то есть, хэш) = f (T (текст), H1), где H1 — алгоритм обработки текста T. Данная функция хеширует T таким образом, что без знания H1 открыть его как полноценный файл будет практически невозможно. Использование хэш-функций на практике: скачивание файлов Изучим теперь подробнее варианты использования хэш-функций на практике. Задействование соответствующих алгоритмов может применяться при написании скриптов скачивания файлов с интернет-серверов.

В большинстве случаев для каждого файла определяется некая контрольная сумма - это и есть хэш. Она должна быть одинаковой для объекта, располагающегося на сервере и скачанного на компьютер пользователя. Если это не так, то файл может не открыться либо запуститься не вполне корректно. Хэш-функция и ЭЦП Использование хэш-функций распространено при организации обмена документами, содержащими электронно-цифровую подпись. Хэшируется в данном случае подписываемый файл, для того чтобы его получатель мог удостовериться в том, что он подлинный. Хотя формально хэш-функция не входит в структуру электронного ключа, она может фиксироваться во флеш-памяти аппаратных средств, с помощью которых подписываются документы, таких как, например, eToken. Электронная подпись представляет собой шифрование файла при задействовании открытого и закрытого ключей. То есть к исходному файлу прикрепляется зашифрованное с помощью закрытого ключа сообщение, а проверка ЭЦП осуществляется посредством открытого ключа. Если хэш-функция обоих документов совпадает - файл, находящийся у получателя, признается подлинным, а подпись отправителя распознается как верная.

Хеширование, как мы отметили выше, не является непосредственно компонентом ЭЦП, однако позволяет весьма эффективно оптимизировать алгоритмы задействования электронной подписи. Так, шифроваться может, собственно, только хэш, а не сам документ. В итоге скорость обработки файлов значительно возрастает, одновременно становится возможным обеспечивать более эффективные механизмы защиты ЭЦП, так как акцент в вычислительных операциях в этом случае будет ставиться не на обработке

исходных данных, а на обеспечении криптографической стойкости подписи. Хэш-функция к тому же делает возможным подписывать самые разные типы данных, а не только текстовые.

3. МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ ПО ПОДГОТОВКЕ К ЗАНЯТИЯМ

3.1 Лабораторные работы по теме «Информация и информационные процессы. Представление информации»

При подготовки к занятию необходимо обратить внимание на следующие моменты.

1. Информатика как наука. Информация: определение, формы представления. Передача информации.
2. Сигналы и данные. Основные структуры данных. Файлы.
3. Виды информации. Свойства информации.
4. Количество информации (вероятностный и объёмный подходы). Единицы измерения информации.
5. Кодирование информации. Кодирование различных форм представления информации (числовой, текстовой).
6. Кодирование информации. Кодирование различных форм представления информации (графической, звуковой).
7. Качество информации.

3.2 Лабораторные работы по теме «Информационно-логические основы построения персонального компьютера»

Правила техники безопасности. Введение в предмет. Способы сбора информации. Кодирование информации

При подготовки к занятию необходимо обратить внимание на следующие моменты.

1. Представление информации в компьютере.
2. Позиционные и непозиционные системы счисления.
3. Двоичный алфавит.
4. Двоичная, восьмеричная, шестнадцатеричная системы счисления.
5. Перевод чисел из двоичной, восьмеричной, шестнадцатеричной систем счисления в десятичную и обратно.

3.3 Лабораторные работы по теме «Состав и структура ЭВМ и ПЭВМ»

Измерение информации. Информационно-логические основы построения персонального компьютера. Логические основы построения персонального компьютера

При подготовки к занятию необходимо обратить внимание на следующие моменты.

1. Компьютер: определение, состав, минимальная и расширенная конфигурация. Устройства ввода и вывода информации.
2. Процессор: назначение, устройство, важнейшие характеристики.
3. Структура памяти компьютера. Внутренняя память.
4. Внешняя память. Основные носители информации и их важнейшие характеристики.
5. Форматирование диска.
6. История создания вычислительной техники.
7. Краткая история развития ЭВМ.
8. Важнейшие этапы компьютерной научно-технической революции.

3.4 Лабораторные работы по теме «Программное обеспечение ПК»

Первоначальные сведения и правила работы в операционной системы Windows. Работа с сервисными программами в операционной системе Windows. Возможности графического редактора Paint и текстового редактора WordPad

При подготовки к занятию необходимо обратить внимание на следующие моменты.

1. Функциональная организация компьютера (магистрально-модульный принцип построения компьютера). Принцип Фон-Неймана.

2. Программное обеспечение компьютера. Классификация программного обеспечения.

3. Операционные системы. Загрузка операционной системы.

4. Операционная система MS DOS. Составные части MS DOS.

5. Начальная загрузка MS DOS. Работа с файлами и каталогами в MS DOS(команды).

6. Формы представления и структуры информационных моделей.

3.5 Лабораторные работы по теме «Текстовые и графические редакторы»

Текстовый редактор. Программные средства решения задач презентационного представления документации. Табличный процессор: работа с листами и графиками

При подготовки к занятию необходимо обратить внимание на следующие моменты.

1. Текстовый процессор WORD. Его возможности Создание простых текстовых документов. Установка параметров страницы, колонтитулы, нумерация страниц.

2. Текстовый процессор WORD. Создание составных текстовых документов: работа с таблицами, диаграммы. Формулы, вычисляемые таблицы.

3. Текстовый процессор WORD. Создание маркированных, нумерованных и многоуровневых списков. Создание оглавления.

4. Текстовый процессор WORD. Гиперссылки и макросы.

5. Основы компьютерной графики.

6. Профессиональные графические редакторы.

7. Форматы графических файлов.

3.6 Лабораторные работы по теме «Электронные таблицы и базы данных»

Табличный процессор: операции с условием. Табличный процессор: работа с массивами. Специальные методы работы с программой Excel. Защита информации в компьютерах и сетях.

При подготовки к занятию необходимо обратить внимание на следующие моменты.

1. Электронные таблицы. Основные манипуляции с таблицами. Блоки, абсолютная и относительная адресация, запись чисел и формул в Excel.

2. MS Excel. Функции: математические; статистические; функции даты.

3. MS Excel Условные (логические) функции.

4. MS Excel. Сортировки данных. Фильтрация данных в списке. Промежуточные итоги.

5. MS Excel Сводные таблицы. Консолидация.

6. Моделирование. Информационные модели. Этапы построения компьютерной информационной модели.

7. Формы представления и структуры информационных моделей.

8. Основные понятия БД. Классификация.

9. СУБД MicrosoftAccess – основные возможности. Базовые объекты СУБД MSAccess.

10. Типы данных в MSAccess. Связи в БД. Целостность БД.

3.7 Лабораторные работы по теме «Основы алгоритмизации и программирования»

Проектирование базы данных в СУБД MS Access. Создание таблиц и пользовательских форм для ввода данных в СУБД MS Access. Модификация таблиц и работа с данными с использованием запросов в СУБД MS AccessПри подготовки к занятию необходимо обратить внимание на следующие моменты.

Работа в глобальной сети Internet.

Защита информации в компьютерах и сетях

1. Технологии программирования.
2. Языки программирования, их назначение, особенности.
3. Классификация языков программирования. Транслятор и компилятор.
4. Структурное программирование.
5. Основные понятия объектно-ориентированного программирования
6. Компьютерная графика. Виды компьютерной графики.

3.7 **Лабораторные работы по теме «Локальные и глобальные сети ЭВМ. Основы защиты информации»**

Работа с данными и создание отчетов в СУБД MS Access. Комплексная работа с объектами СУБД MS Access. Алгоритмизация математических задач. Элементы программирования на языке высокого уровня Локальные и глобальные сети ЭВМ

При подготовки к занятию необходимо обратить внимание на следующие моменты.

1. Запросы. Определение условий отбора.
2. Передача информации. Локальные компьютерные сети.
3. Классификация сетей. Сетевые топологии.
4. Глобальные сети. Основные возможности, предоставляемые сетью Интернет.
5. Технические и программные ресурсы Интернета.
6. Информационная система. Основные свойства информационных систем.
7. Лицензионные, условно бесплатные и бесплатные программы.
8. Защита информации. Криптографические средства защиты информации.