

Министерство образования Красноярского края
краевое государственное бюджетное
профессиональное образовательное учреждение
«Красноярский аграрный техникум»

КОНСПЕКТ ЛЕКЦИЙ ПО ИНФОРМАТИКЕ

Для студентов 1-2 го курса

Красноярск

2020

В конспекте представлены разделы информатики, охватывающие основные вопросы теории информации, функционирования аппаратного обеспечения, алгоритмизации, принципов работы различных программных продуктов, устройства вычислительных сетей и основы теории баз данных. Изложены основные приемы программирования, используемые студентами на практических занятиях, а также при индивидуальной и самостоятельной работе студентов.

Оглавление

ВВЕДЕНИЕ	5
ЛЕКЦИЯ 1.	7
Информация и информатика	7
Представление данных. Системы счисления	10
ЛЕКЦИЯ 2.	13
История развития вычислительной техники.	13
Классификация и состав ЭВМ.	15
ЛЕКЦИЯ 3.	19
Типы запоминающих устройств. Хранение и обработка информации.	19
Принцип работы компьютера	22
ЛЕКЦИЯ 4.	25
Программное обеспечение.	25
Операционные системы	25
Системы программирования	28
ЛЕКЦИЯ 5.	30
Технология разработки программного обеспечения	30
Тестирование и отладка программ	34
ЛЕКЦИЯ 6.	37
Вычислительные комплексы и сети	37
Сеть Интернет	40
ЛЕКЦИЯ 7.	44
Базы данных	44
Объекты предметной области и связи между ними	44
Отношения	45
СУБД	49

ЛЕКЦИИ 8, 9.	51
Некоторые приёмы программирования	51
ВОПРОСЫ ДЛЯ САМОПРОВЕРКИ	57
ЗАКЛЮЧЕНИЕ	59
СПИСОК ЛИТЕРАТУРЫ	60

Введение

В конспекте представлен материал лекций, читаемых студентам машиностроительных специальностей МГТУ им. Н.Э. Баумана (факультеты МТ, СМ, Э), изучающим дисциплину “Информатика”.

Информатика является базовой учебной дисциплиной, охватывающей сведения о технических, программных и алгоритмических средствах организации современных информационных систем и формирующей у обучаемого определенный кругозор, объем знаний, уровень алгоритмического мышления, а также практические навыки работы с конкретными программными системами.

Цель преподавания дисциплины состоит в освоении студентами современных информационных технологий, формирование представления о задачах, реализуемых с их помощью, методах их решения, формирование алгоритмического мышления. Дисциплина реализует базовую подготовку по программированию, рассчитанную на студентов младших курсов.

Студент, приступающий к изучению дисциплины должен обладать следующими компетенциями:

- владением культурой мышления, способностью к обобщению, анализу, восприятию информации, целей и выбору путей их достижения;
- способностью владеть основными методами и средствами получения, хранения, переработки информации, иметь навыки работы с компьютером как средством управления информацией, в том числе в глобальных компьютерных сетях;
- владением английским языком, способностью воспринимать научно-техническую информацию из зарубежных первоисточников;
- готовностью учитывать современные тенденции развития вычислительной техники, информационных технологий;
- способностью применять современные программные средства выполнения и редактирования изображений;
- способностью строить математические модели технологических процессов и оборудования, а также использовать стандартные программные средства их компьютерного моделирования.

Задачами преподавания дисциплины является изучение:

- современных технических и программных средств взаимодействия с компьютером;
- современных технологий сбора, представления, хранения, обработки и передачи информации с использованием компьютеров;
- методов разработки алгоритмов и приложений;
- особенностей технологий структурного и объектно-ориентированного программирования;
- языка программирования высокого уровня;
- методов тестирования и отладки разрабатываемых приложений.

Изучение дисциплины предполагает предварительное освоение следующих дисциплин (в рамках школьного курса):

- основы информатики;
- математика;
- иностранный язык (английский).

В предлагаемом конспекте лекций рассмотрены разделы информатики, определяющие базовый уровень подготовки специалистов: основы информационной культуры, современные технические средства и программный инструментарий новых информационных технологий (системное и прикладное программное обеспечение, инструментарий создания программных продуктов), принципы функционирования вычислительных сетей и основы теории баз данных. Изложены также базовые основы алгоритмизации, необходимые студентам для освоения программирования. Приведены примеры типовых алгоритмов.

Лекция 1.

Информация и информатика

Информация – это сведения о лицах, предметах, фактах, событиях, явлениях, процессах независимо от формы их представления.

Свойства информации:

- 1) атрибутивные (без них информация не существует):
 - a) *непрерывность* (возможность «сливаться» с ранее накопленной информацией);
 - b) *дискретность* (информация характеризует отдельные данные и свойства объектов);
- 2) прагматические (характеризуют степень полезности):
 - a) *новизна*;
 - b) *ценность*;
 - c) *полнота*;
 - d) *актуальность*;
 - e) *доступность*;
 - f) *достоверность*
- 3) динамические (характеризуют изменение информации с течением времени):
 - a) *накопление информации*;
 - b) *старение информации*.

Объём используемой человеком информации в мире постоянно растёт. В таблице 1 показана динамика роста человеческих знаний.

Таблица 1 – Увеличение человеческих знаний

Общая сумма человеческих знаний удваивалась:	
Каждые 50 лет	до 1800 года
Каждые 10 лет	до 1950 года
Каждые 5 лет	до 1970 года
Ежегодно	до 1990 года

Этому способствовали *информационные революции* (таблица 2), в ходе которых существенно менялись средства и способы хранения, распространения информации, её доступность.

Таблица 2 – Информационные революции

Информационная революция	Причина	Когда произошла
Первая	Появление языка и членораздельной речи	10 тыс. лет до Н.Э.
Вторая	Появление письменности	3 тыс. лет до Н.Э.
Третья	Книгопечатание	VII век Н.Э.
Четвёртая	Телефон, телеграф, радио, фотография, кинематограф, телевидение	Конец XIX – начало XX века
Пятая	Появление ЭВМ	Середина XX века

Современное общество называется *информационным*, поскольку большинство работающих людей занято обработкой информации.

При накоплении большого объёма информации и неспособности человека её обработать возникает *информационный кризис*. Преодоление информационного кризиса обеспечивается *информатизацией* общества, которая представляет собой процесс создания оптимальных условий для удовлетворения информационных потребностей человека. В этом процессе базовой технической составляющей является *вычислительная техника*, которая позволяет автоматизировать (то есть ускорить и упростить) обработку информации.

Формы представления информации (рисунок 1):

- 1) Непрерывная (аналоговая) – характеризует процесс, который не имеет перерывов и может изменяться в любой момент времени на любую величину (например - музыка);

2) Прерывистая (дискретная, цифровая) – характеризует процесс, который может изменяться лишь в определённые моменты времени и принимать лишь заранее обусловленные значения.

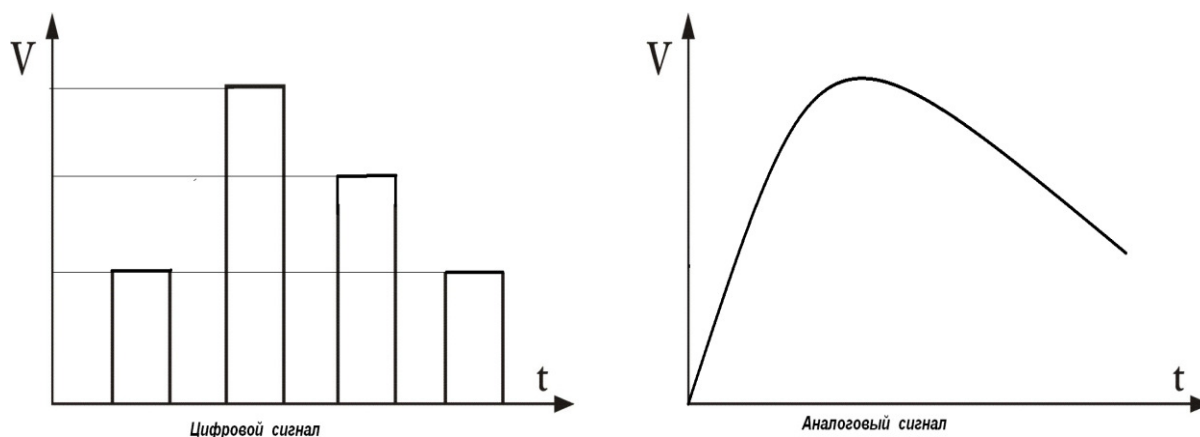


Рисунок 1 – Представление информации различными типами сигналов

Большинство современных компьютеров обрабатывают информацию в виде последовательности электрических сигналов только двух определенных уровней (например – высокого и низкого) – двоичных сигналов, то есть являются цифровыми.

Аналогом такого сигнала в информатике является *бит* (*binary digit* – двоичный разряд), который может принимать только одно из двух возможных значений (например - 0 и 1, + и – и т.д.). *Бит* – минимальная единица информации. Более крупная единица – *байт* (последовательная комбинация из 8 бит). Байт позволяет получать уже не две, а 256 возможных комбинаций.

Другие более крупные единицы:

1 Килобайт = 1024 байта;

1 Мегабайт = 1024 килобайт;

1 Гигабайт = 1024 мегабайт;

1 Терабайт = 1024 гигабайт и т.д.

Информатика – техническая наука, занимающаяся способами создания, хранения, воспроизведения, обработки и передачи информации средствами вычислительной техники, принципами функционирования этих средств и методами

управления ими. Термин информатика произошел от слияния двух французских слов *Informacion* (информация) и *Automatique* (автоматика) и дословно определял новую науку об «автоматической обработке информации». В англоязычных странах информатика называется *Computer Science* (наука о компьютерной технике).

Информационная технология – процесс, использующий совокупность средств и методов сбора, обработки и передачи первичной информации для получения информации нового качества о состоянии объекта, процесса или явления (информационного продукта).

Данные – это зарегистрированная (зафиксированная) определенным образом информация, представленная в некоторой форме (формализованном виде), что обеспечивает ее хранение, обработку и передачу. Регистрация информации возможна различными способами – изменением магнитных, оптических, химических свойств материалов.

Основные операции с данными:

- 1) сбор данных;
- 2) фильтрация;
- 3) преобразование;
- 4) транспортировка;
- 5) архивация и т.д.

Представление данных. Системы счисления

Наиболее распространенные - числовые данные могут быть представлены в различном виде. Вид этот определяется используемой системой счисления.

Система счисления (СС) – совокупность приемов и правил представления чисел в виде конечного числа символов. СС имеет свой алфавит (упорядоченный набор цифр и букв) и совокупность операций образования чисел из этих символов.

Системы счисления разделяют на не позиционные и позиционные.

Не позиционная система счисления – это система, в которой цифры не меняют своего количественного эквивалента в зависимости от местоположения (позиции) в записи числа. К не позиционным системам счисления относится, например, система *римских цифр*, основанная на употреблении латинских букв:

I – 1;	L – 50;	M – 1000.
V – 5;	C – 100;	
X – 10;	D – 500;	

Значение числа в этой системе определяется как сумма или разность цифр в числе (если меньшая цифра стоит перед большей, то она вычитается, а если после - прибавляется). Например, число 1998 записывается как MCMXCVIII.

Не позиционные системы счисления обладают следующими недостатками:

- сложность представления больших чисел (больше 10000);
- сложность выполнения арифметических операций над числами, записанными с помощью этих систем счисления.

Позиционная система счисления – это система, в которой количественный эквивалент цифры зависит от ее положения в числе (чем «левее» цифра в записи числа, тем её значение больше). *Основание позиционной системы счисления* – это количество разных символов в ее алфавите. Например, в двоичной системе счисления используется две цифры (0 и 1), в восьмеричной – восемь (0,1,...,6,7), а в десятичной системе счисления используется десять цифр (0,1,...,8,9). Сравнение записи чисел в разных системах счисления представлено в таблице 3.

Таблица 3 – Сравнение записи чисел в трёх системах счисления

Десятичная	Восьмеричная	Двоичная
0	0	0
1	1	1
2	2	10
3	3	11
4	4	100
5	5	101
6	6	110
7	7	111
8	10	1000
9	11	1001
10	12	1010

Наиболее используемой системой счисления является десятичная система счисления, а для представления чисел в большинстве современных ЭВМ используется двоичная система счисления

Правило перевода числа из десятичной системы в двоичную систему счисления: перевод целой части – делением на основание системы, в которую переводим (на 2), а дробной части – умножением на это основание. Операции выполняются в десятичной системе. Остатки от деления собираются в обратном порядке.

Пример: перевести число 100 в двоичную систему счисления (рисунок 2).

Решение: представим перевод числа в виде столбца, каждая строка которого содержит частное и остаток от деления данного числа на основание двоичной системы счисления $n = 2$.

$$\begin{array}{r}
 100 \quad | \quad 2 \\
 \hline
 100 \quad 50 \quad | \quad 2 \\
 \hline
 \quad 0 \quad 50 \quad 25 \quad | \quad 2 \\
 \hline
 \quad \quad 0 \quad 24 \quad 12 \quad | \quad 2 \\
 \hline
 \quad \quad \quad 1 \quad 12 \quad 6 \quad | \quad 2 \\
 \hline
 \quad \quad \quad \quad 0 \quad 6 \quad 3 \quad | \quad 2 \\
 \hline
 \quad \quad \quad \quad \quad 0 \quad 2 \quad 1 \\
 \hline
 \quad \quad \quad \quad \quad \quad 1
 \end{array}$$

Рисунок 2 – Перевод числа из десятичной системы в двоичную

В результате получим число 1100100_2 – результат перевода числа 100_{10} в двоичную систему счисления (индекс – основание системы счисления).

Как было уже сказано, в вычислительной технике используется двоичная система счисления (данные представляются в виде закодированной последовательности двоичных сигналов). Это обеспечивает высокую надёжность и помехоустойчивость вычислительной системы, так как в ней реализованы устройства лишь с двумя устойчивыми состояниями (чем проще устройство, тем оно надежнее).

При этом для описания логики функционирования аппаратных и программных средств используется алгебра логики (Булева алгебра). Она оперирует с логическими переменными, которые могут принимать тоже только два возможных значения (true — истина и false - ложь). Это очень удобно, так как обеспечивается универсальность (однотипность) процесса обработки информации на компьютере.

Лекция 2.

История развития вычислительной техники.

Этапы развития вычислительной техники представлены в таблице 4, а разделение её по поколениям (и по элементной базе) – в таблице 5.

Таблица 4 - Этапы развития вычислительной техники

<i>Этап</i>	<i>Время</i>
Ручной (абак, счеты)	3 тыс. лет до Н.Э.
Механический (арифмометр)	Конец XVII века
Электромеханический	Конец XIX века
Электронный (ЭВМ)	С середины XX века по наше время

Таблица 5 - Поколения ЭВМ

<i>Поколение</i>	<i>Годы</i>	<i>Элементная база</i>
Первое	1950-1955	Электронные лампы
Второе	1955-1965	Транзисторы
Третье	1965-1980	Интегральные микросхемы
Четвертое - пятое	С 1980 до настоящего времени	Микропроцессоры

Разделение ЭВМ на поколения условно, так как они сменялись постепенно, и временные границы между поколениями размыты. Поколения разделяют в зависимости от основных элементов, используемых при изготовлении ЭВМ.

Первое поколение ЭВМ строилось на электронных лампах. Эти ЭВМ, содержащие десятки тысяч ламп, были громоздкими, ненадёжными, требовали большой мощности (для нагрева катода).

Второе поколение ЭВМ строилось на транзисторах – полупроводниковых устройствах. По сравнению с лампами транзисторы имели малые размеры и потребляемую мощность. ЭВМ были более надёжными и занимали гораздо меньше места.

Третье поколение ЭВМ строилось на полупроводниковых интегральных схемах (ИС). ИС представляет собой электрическую цепь, которая выполнена в виде единого полупроводникового кристалла и содержит большое количество элементов (диодов и транзисторов), что позволило уменьшить размеры, потребляемую мощность, стоимость и увеличить надёжность системы.

Четвёртое поколение ЭВМ строится на больших интегральных схемах (БИС). БИС содержат миллионы транзисторов в одном кристалле и представляют собой целые функциональные узлы компьютера. Примером БИС является микропроцессор. БИС способствовали появлению персональных компьютеров.

ЭВМ пятого поколения пока существуют лишь в теории. Они основываются на логическом программировании (компьютер должен сам в зависимости от поставленной задачи составить план действий и выполнить его). Их элементная база: сверхбольшие интегральные схемы – СБИС с применением оптоэлектроники (использование эффектов взаимодействия оптического излучения с электронами в твердых телах для генерации, отображения, хранения, обработки и передачи информации) и криогенной электроники (применение явлений, происходящих в твердых телах при температурах менее 120К в присутствии электромагнитных полей, для создания электронных устройств).

Некоторая любопытная информация:

- 1946 г. - первая ЭВМ (США, «ENIAC», содержала 18 000 ламп, весила 30 тонн, размер – 4м*30м*6м, ОП – 600 бит, 5 000 операций в секунду, работала 10 лет)
- 1950 г. - первая советская ЭВМ («МЭСМ», ОП – 1800 бит)
- 1976 г. - первый персональный компьютер (ПК) компании «APPLE» (ОП – 48 кбайт, 1 МГц)
- 1983 г. – первый персональный компьютер компании IBM («IBM PC/XT», ОП – 640 Кбайт, ЖД – 10 Мбайт, 10 МГц)

Классификация и состав ЭВМ.

Возможны различные виды классификации компьютеров:

1. По элементной базе (см. выше).
2. По производительности:
 - a) *Супер-ЭВМ*. Самые мощные компьютеры, представляющие собой многопроцессорные вычислительные системы. Предназначены для решения уникальных задач (прогнозирование метеобстановки, управление космическими и оборонными комплексами и др.). Очень дорогие (стоят сотни миллионов долларов).
 - b) *ЭВМ общего назначения*. Предназначены для решения широкого класса научно-технических и статистических задач. Они обрабатывают около 60% всей информации в мире.
 - c) *Мини-ЭВМ*. Предназначены чаще всего для управления технологическими процессами предприятий. Они гораздо компактнее и дешевле ЭВМ общего назначения.
 - d) *Микро-ЭВМ и персональные компьютеры*. Появились после изобретения микропроцессора. Имеют очень широкую область применения. Ещё более компактны. К ним относятся:
 - *Учебные (используются в тренажерах)*.
 - *Бытовые (в бытовой технике)*.
 - *Профессиональные (персональные компьютеры)*.
3. По типу обрабатываемых сигналов (см. выше):
 - ЭЦВМ (цифровые).
 - АВМ (аналоговые).
 - Гибридные (смешанные).

Определение ЭВМ:

Электронная вычислительная машина (ЭВМ, компьютер) — комплекс программных и технических средств, объединённых под общим управлением и предназначенный для автоматизированной обработки информации по заданному алгоритму.

Современная ЭВМ является единым комплексом из нескольких устройств. Каждое устройство представляет собой автономный, конструктивно законченный модуль с типовым сопряжением. ЭВМ может иметь переменный состав оборудования. В основе её работы лежит принцип открытой архитектуры.

Архитектура – это наиболее общие принципы построения ЭВМ, реализующие программное управление работой и взаимодействием основных ее функциональных узлов. В основе архитектуры современных ЭВМ лежат принципы, предложенные американским ученым и теоретиком вычислительной техники Джоном фон Нейманом. В соответствии с ними выделяются пять базовых элементов компьютера:

- арифметико-логическое устройство;
- устройство управления;
- запоминающее устройство;
- система ввода информации;
- система вывода информации.

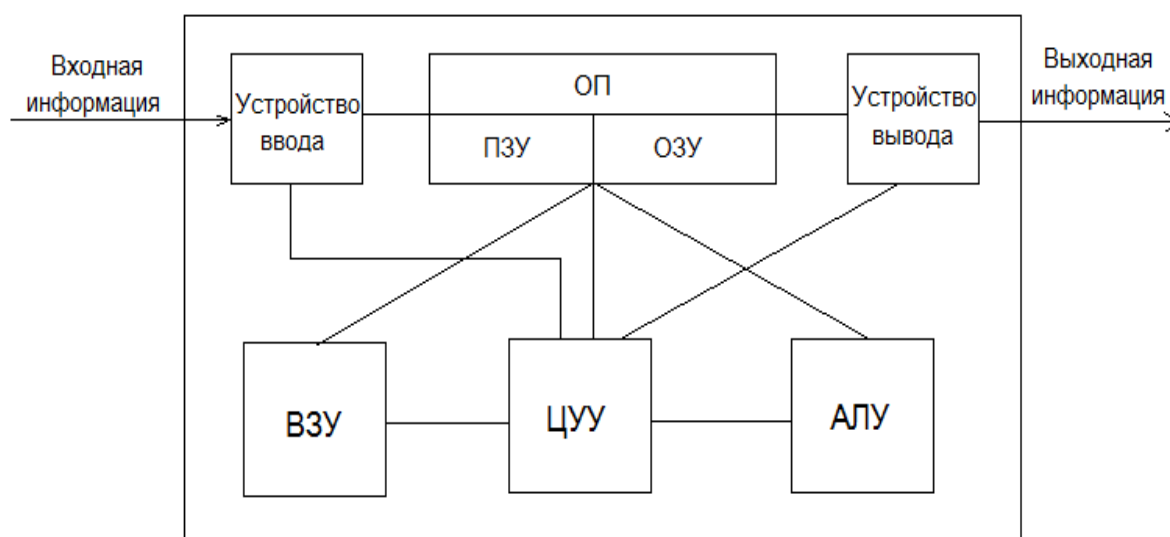


Рисунок 3 - Обобщённая структурная схема ЭВМ

В представленной на рисунке 3 обобщенной схеме можно выделить следующие элементы:

Устройство ввода служит для преобразования информации в закодированную последовательность сигналов и записи её в основную память (ОП).

Примеры устройств ввода:

- Клавиатура (ввод информации в виде последовательности символов, которые образуют команды);
- Манипуляторы: мышь, джойстик, touchpad, touchscreen (информация вводится путём выбора из предлагаемого набора какой-либо информации);
- Сканер;
- Камера;
- Микрофон.

Устройство вывода служит для преобразования результатов обработки сигналов в информацию, в удобном для пользователя виде.

Примеры устройств вывода:

- Монитор (электронно-лучевая трубка, жидкокристаллический, плазменный);
- Принтер (матричный, струйный, лазерный);
- Плоттер (графопостроитель);
- Динамик;
- Экраны, проекторы.

Основная память (ОП) – устройство, предназначенное для хранения данных и программ. Это *электронное* устройство, основанное на микросхемах. Для него характерна большая скорость доступа к данным. Состоит из ПЗУ и ОЗУ.

ПЗУ – *постоянное запоминающее устройство*. Хранит служебные программы (записанные туда при изготовлении микросхемы устройства), выполняемые во время загрузки ЭВМ (диагностика и начальная отладка, оптимизация связей, запуск загрузчика

операционной системы). Является энергонезависимой памятью (при выключении компьютера информация, записанная в ПЗУ, не пропадает).

ОЗУ – оперативное запоминающее устройство. Хранит программы, исходные данные и результаты обработки во время их использования. Является энергозависимой памятью.

ВЗУ – внешнее запоминающее устройство. Служит для длительного хранения программ и больших объёмов данных. По мере необходимости они переписываются в ОП и там используются. В настоящее время это, как правило, электромеханические устройства. В связи с этим, скорость доступа к данным у этих устройств гораздо ниже, чем у электронных.

ЦУУ – центральное устройство управления. Осуществляет управление аппаратными и программными ресурсами ЭВМ. Производит чтение команд из основной памяти, определяет адреса операндов команд, тип операции, передаёт сигнал в ОП и АЛУ.

АЛУ – арифметико-логическое устройство. Выполняет арифметические и логические операции над данными и вырабатывает различные условия, влияющие на ход вычислительного процесса.

ЦУУ и АЛУ вместе составляют ПРОЦЕССОР.

Процессор и основная память вместе составляют *центральные устройства (ядро) ЭВМ.* Остальные устройства являются *внешними устройствами ЭВМ.*

Лекция 3.

Типы запоминающих устройств. Хранение и обработка информации.

Существуют различные типы запоминающих устройств:

1) Магнитные:

- a. На магнитной ленте (стримеры) – устройства с последовательным доступом к информации (до 600 Мбайт);
- b. На магнитном диске – устройства с прямым доступом к информации:
 1. гибкие (дискеты) – 1,44 Мегабайт;
 2. жёсткие (винчестеры) – Терабайты;

Данные на дисках записываются на дорожки концентрического типа (на гибком диске 80 дорожек, а на жестком – более 1224), которые разделены на отрезки-сектора по 512 байт. Количество секторов – 18 (гибкая дискета) или 35 (жесткий диск). При этом не все секторы предназначены для файлов пользователя. Существуют различные типы специальных («служебных») секторов. Например, секторы:

1. с программой-загрузчиком;
2. с таблицей размещения файлов;
3. со справочной информацией о файлах и др.

Недостатком этих ВЗУ является возможность их размагничивания.

2) Оптические (таблица 6).

Таблица 6 – Оптические диски

Тип диска	Возможность перезаписи	Ёмкость
CD-ROM	только для чтения	700 Мб
CD-R	можно записать <u>один</u> раз	700 Мб
CD-RW	есть возможность перезаписи	700 Мб
DVD-R	можно записать <u>один</u> раз	4,7 Gb
DVD-RW	есть возможность перезаписи	4,7 Gb
Двухслойный DVD	есть возможность перезаписи	16 Gb
Blu-Ray Disc	есть возможность перезаписи	25..128 Gb

3) **Электронные** – флэш-накопители. Ёмкость их на данный момент доходит до 512 гигабайт. Отличается высокой скоростью доступа к данным и надёжностью, благодаря отсутствию механических узлов.

В большинстве случаев информация на всех устройствах указанных типов хранится, в файлах. **Файл** – именованная область носителя информации (ленты, диска, флэш-карты и пр.), предназначенная для хранения информации в различном виде. Файл имеет имя, которое состоит из собственно имени (задаётся пользователем) и расширения (как правило, создается программой, которая обрабатывает этот файл). Пользовательское имя и расширение разделяются точкой (например – spisok.txt; document.doc; kartina.bmp; program.pas; program.exe и т.д.). Расширения файлов используются операционной системой, чтобы определить программу, которую необходимо запустить для обработки файла с данным расширением. Расширение определяет тип файла (текстовый, графический, звуковой, двоичный и т.д.). Теоретически расширение может и отсутствовать (это сделает работу с файлом неудобной). Некоторые расширения файлов приведены в таблице 7.

Таблица 7 – Типы расширений

Расширение	Тип
EXE, COM	Исполняемые файлы – программы
DOC, RTF, TXT	Документы
SYS	Системные файлы
BMP, JPG, GIF, PNG	Файлы изображений
MID, MP3, WAV, WMA	Звуковые файлы
ASF, AVI, MOV, MP4, MPG	Видеофайлы

При манипулировании группами файлов (их копировании, перемещении, удалении и др.) удобно для ускорения работы использовать специальные символы в именах файлов:

? – означает один любой символ;

* – означает любое количество любых символов.

Например:

*program1.** - речь идет о всех файлах с именем program1 и любым расширением;

program?.for – все файлы с расширением for и именем program1 или program2 или program3;

. - все файлы и т.д.

Файлы по желанию пользователя могут объединяться в именованные группы – *папки* (*директории, каталоги*), которые именуются (без расширения). Папки могут объединяться в другие папки, образуя древообразную (иерархическую) структуру, записанную на носитель информации, который тоже имеет имя, состоящее из одной буквы латинского алфавита:

А, В – гибкие диски;

С, D, E... – жёсткие диски.

Пользователь может разделить жёсткий диск на несколько частей (логических дисков), которым тоже присваивается имя, состоящее из одной латинской буквы.

Папка, в которой работает пользователь, называется *текущей* (или *рабочей*). Самая первая папка (имеет имя диска) называется *корневой*. Таким образом, *полное имя файла* состоит из собственно имени с расширением и пути к этому файлу от корневой папки (путь состоит из имён папок, разделённых знаком «обратный слэш» - \).

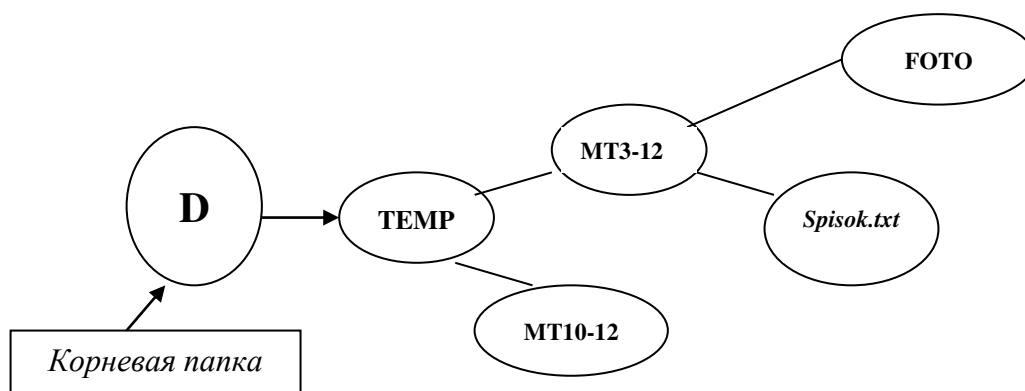


Рисунок 4 – Система файлов и папок на диске

Например, полное имя файла *Spisok.txt*, показанного на рисунке 4 в схеме файловой системы, выглядит следующим образом:

D:\TEMP\MT3-12\Spisok.txt,

где *D:\TEMP\MT3-12* – путь;

Spisok – имя;

.txt – расширение.

Принцип работы компьютера

В соответствии с принципом фон Неймана, компьютер работает под управлением программы, загруженной в основную память. Программа – совокупность команд, которые выполняются в определённой последовательности.

Примеры типовых команд: арифметическое действие, запись, считывание и пересылка данных.

Рассмотрим на схеме выполнение одной из команд (операторов) программы – команды сложения двух чисел – операндов команды (В и С) и получения результата выполнения команды – А:

$$A = B + C.$$

Компьютер работает с этой командой, как с последовательностью двоичных сигналов (используем 1 – для сигнала высокого уровня, 0 – для сигнала низкого уровня).

Тогда условно можно представить команду в таком виде:

010 1000 1001 0110,

где 010 – код операции (сложение);

1000 – адрес операнда В;

1001 – адрес операнда С;

0110 – адрес результата А.

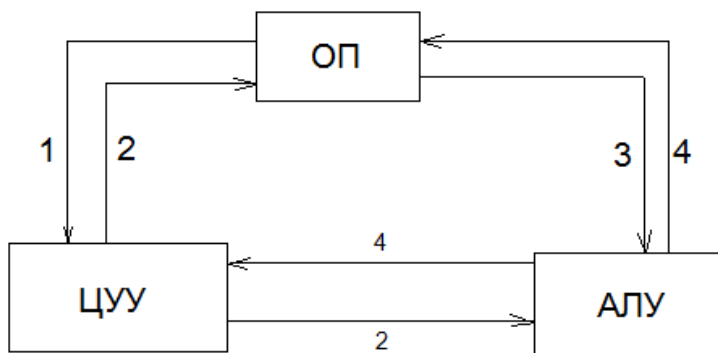


Рисунок 5 – Взаимодействие центральных устройств

Тогда последовательность действий будет выглядеть следующим образом (рисунок 5):

- 1) ЦУУ считывает команду из ОП (в которой записаны исходные данные и программа);
- 2) ЦУУ передаёт сигнал
 - в ОП об адресах операндов (В и С) и результата (А)
 - в АЛУ о коде операции (сложение);
- 3) Из ОП в АЛУ передаются значения операндов В и С;

4) АЛУ

- вычисляет сумму
- передаёт её значение в ОП
- передаёт сигнал в ЦУУ о выполнении команды, на основании которого происходит считывание следующей команды

Процесс взаимодействия центральных и внешних устройств ЭВМ происходит посредством *интерфейса* (сопряжения), под которым понимается *совокупность линии связи между устройствами, а также вид и порядок сигналов, проходящих по этим линиям.*

Типы взаимодействия:

- *множественный интерфейс* - каждое устройство компьютера соединено отдельными линиями связи с другими устройствами;
- *единый интерфейс (общая шина)* – в этом случае на одну линию связи (шину) параллельно подключены все устройства компьютера. Их взаимодействие происходит в режиме разделённого по времени интерфейса (по очереди).

Шина – не только линии связи, но и устройства синхронизации и усиления сигналов. Важная характеристика шины – пропускная способность (количество информации в единицу времени). Зависит она от разрядности шины и от тактовой частоты компьютера. Разрядность (количество проводов шины) определяет количество бит информации, обрабатываемой одновременно. Тактовая частота задает скорость выполнения операций.

Существуют шины трёх типов:

- Шины данных;
- Шины адресов;
- Шины команд.

Отдельно необходимо отметить особенности наиболее распространенных компьютеров – персональных. Первые персональные компьютеры (ПК) по сравнению с существовавшими ЭВМ имели следующие основные особенности (рисунок 6):

1. Основа элементной базы – *микропроцессор (МП)* – *программно-управляющее средство*, построенное на больших интегральных схемах (БИС);

2. Взаимодействие устройств ЭВМ происходит посредством *единого интерфейса (общей шины)*.



Рисунок 6 – Схема персонального компьютера

Центральные устройства:

МП – микропроцессор.

ОП – основная память.

Внешние устройства:

ВЗУ – внешнее запоминающее устройство.

УВВ – устройство ввода.

Увыв – устройство вывода.

ТКУ – телекоммуникационное устройство.

Современные ПК активно подключаются к компьютерным сетям. Поэтому в их архитектуре появляются ТКУ - модем, сетевая карта и др.

Модем - устройство для преобразования цифрового сигнала в аналоговый и наоборот. Он используется для подключения к сети Internet через телефонную, которая является аналоговой, линию.

Лекция 4.

Программное обеспечение.

Программное обеспечение (ПО) – организованная совокупность обрабатывающих программ и обрабатываемых данных, реализованная на ЭВМ.

ПО делится на две группы (рисунок 7):

1. *Общее ПО* – предназначено для обеспечения функционирования компьютера и эффективной работы на нём. Этим ПО пользуется каждый пользователь. В состав ПО входит: операционная система (ОС) и специальный комплекс программ технического обслуживания (КПТО).
2. *Специальное (или прикладное) ПО* – предназначено для решения специальных прикладных задач. С ним работают пользователи-специалисты какой либо прикладной области (математики, экономисты, художники, программисты и др.). В его состав входят пакеты прикладных программ (ППП). Среди них отдельно выделим системы программирования (СП).

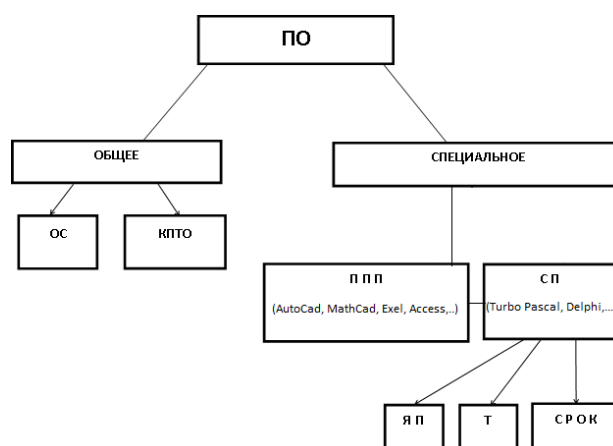


Рисунок 7 – Типы программного обеспечения

Операционные системы

Операционная система – комплекс программ, обеспечивающий организацию вычислительного процесса на компьютере.

Основные функции ОС:

1. Управление аппаратными и программными ресурсами ЭВМ.
2. Организация интерфейса (взаимодействия) пользователя с ЭВМ.
3. Запуск на выполнение прикладных программ.

ОС представляет собой пакет программ в виде файлов, расположенных, как правило, на жестком диске. После включения компьютера она автоматически загружается в основную память с помощью специальной программы-загрузчика, записанной в ПЗУ. В состав ОС в качестве дополнительных к основному пакету программ, как правило, добавляются *драйверы* – специальные программы, обеспечивающие нормальную, полноценную работу дополнительных внешних устройств (принтеров, сканеров и т.п.).

На практике используются различные ОС: MS DOS, WINDOWS, UNIX, LINUX, OS/2, OS X и т.д. Наиболее распространенной является операционная система фирмы Microsoft – WINDOWS, обладающая интуитивно понятным, дружественным графическим интерфейсом, и позволяющая одновременно работать с несколькими приложениями.

Графический интерфейс основан на системе окон и значков.

Окно WINDOWS – прямоугольная область экрана, внутри которой может выполняться какая-либо программа (окно приложения), могут выводиться сообщения или вводиться данные (окно диалога), могут располагаться какие-либо объекты – файлы, папки, диски и т.д. (групповое окно).

3 состояния окна:

- 1) обычное;
- 2) развёрнутое (во весь экран);
- 3) свёрнутое.

Значок – графическое представление объекта ОС (диска, папки, файла).

Ярлык – особый вид значка, ссылка на объект WINDOWS. Ярлыки используются для удобства запуска программ из разных мест и обеспечения сохранности объектов.

Организация интерфейса пользователя с компьютером осуществляется с помощью диалога, который может быть различным.

Типы диалога пользователя с компьютером:

- 1) меню (пользователь выбирает один вариант действий из нескольких предложенных);
- 2) вопросы, требующие ответа типа да/нет (частный случай меню);
- 3) шаблоны (ОС воспринимает информацию пользователя в строго определённой заданной форме);

4) команды.

Диалог также может быть *синхронным* и *асинхронным*. При синхронном диалоге его участники (пользователь и компьютер) поочередно находятся в активном состоянии, которое характеризуется обработкой сообщений, их анализом и выработкой решений. При асинхронном диалоге его участники одновременно находятся в активном состоянии (пользователь может в любой момент вмешиваться в работу компьютера и вносить в нее какие-либо изменения).

Программы операционной системы, постоянно находящиеся в оперативной памяти называются *ядром операционной системы*. Программы, загружающиеся в оперативную память по мере необходимости – *транзиты ОС*.

Динамическая память – часть оперативной памяти, свободная от ядра и транзитов. Она используется прикладными программами – программами, решающими какие-либо специальные (прикладные) задачи.

Решение прикладной задачи на компьютере под управлением ОС можно представить следующим образом (рисунок 8):



Рисунок 8 – Алгоритм решения задачи

Комплекс программ технического обслуживания – пакет сервисных программ для:

- проверки (определения наличия неисправностей);
- диагностики (локализации и классификации ошибок);
- отладки (исправления ошибок).

Эти программы записаны в ПЗУ, могут входить в состав ОС или устанавливаться дополнительно. К ним можно отнести программы для очистки, исправления, дефрагментации дисков, программы-антивирусы и т.п.

Системы программирования

Системы программирования предназначены для автоматизации процесса написания программ. В их состав входит язык программирования (ЯП), транслятор (Т) и специальные средства редактирования, отладки и компоновки (СРОК).

Язык программирования – совокупность правил, определяющих систему записей, составляющих программу, а так же определяющих синтаксис и семантику (смысл) используемых грамматических конструкций.

Типы языков программирования:

- Машинно-зависимые языки (зависят от типа компьютера):
 - Язык машинных команд (двоичный код).
 - Язык ассемблера (язык символьного кодирования). Ассемблер – специальная программа, которая переводит написанный код в машинные команды.
- Машинно-независимые языки - языки высокого уровня (Паскаль, Бейсик, С++ и др.).

Транслятор – системная программа, осуществляющая перевод программы с языка программирования высокого уровня на язык машинных команд.

Типы трансляторов:

- *Интерпретатор* – программа, которая преобразует каждый оператор программы в машинную команду и сразу передаёт её на выполнение. После выполнения преобразуется следующий оператор и т.д. (Плюс интерпретатора – удобство отладки программы. Минус – эта программа постоянно находится в оперативной памяти).

- *Компилятор* – преобразует в машинный код всю программу целиком и только потом отдаёт ее на выполнение (Плюс – не заполняется оперативная память).

Средства редактирования, отладки и компоновки включают в себя следующие программы: редактор (позволяет набирать и редактировать текст программы), отладчик (для нахождения ошибок), компоновщик (подключает к разрабатываемой программе библиотечные подпрограммы, отлаживает связи между ними и создает исполняемый файл).

Лекция 5.

Технология разработки программного обеспечения

Жизненный цикл программы состоит из этапа разработки программы и этапа её эксплуатации и сопровождения (контроль работоспособности и при необходимости внесение изменений и дополнений).

Технология разработки ПО – совокупность приёмов, позволяющих создать безошибочную программу в течение заданного времени. Состоит из четырёх этапов:

- 1) формулировка задачи на естественном языке и создание математической модели;
- 2) разработка нового или выбор существующего метода численного решения математической задачи (алгоритма);
- 3) написание программы на языке программирования;
- 4) тестирование и отладка программ.

На первом этапе необходимо наиболее глубоко исследовать предметную область (процесс, объект, явление), а также разработать наиболее полную математическую модель, учитывающую основные особенности предметной области.

На втором этапе при разработке алгоритма необходимо использовать приёмы *структурного программирования* (см. ниже), позволяющие создавать надёжно работающие программы. Алгоритм принято представлять в виде графической схемы, которая составляется из нескольких геометрических фигур – блоков. Основные блоки схемы алгоритма выглядят следующим образом (рисунок 9):

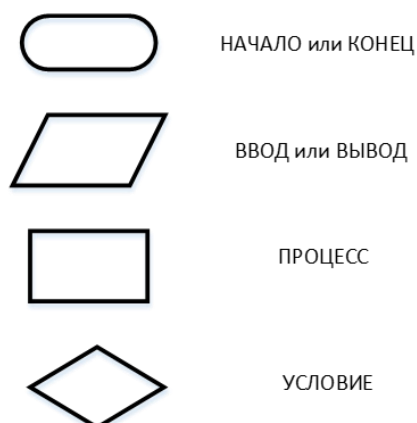


Рисунок 9 – Блоки схемы алгоритма

Например, схема алгоритма простейшей программы линейной структуры (ввод, сложение двух чисел A , B и вывод результата C) выглядит следующим образом (рисунок 10):

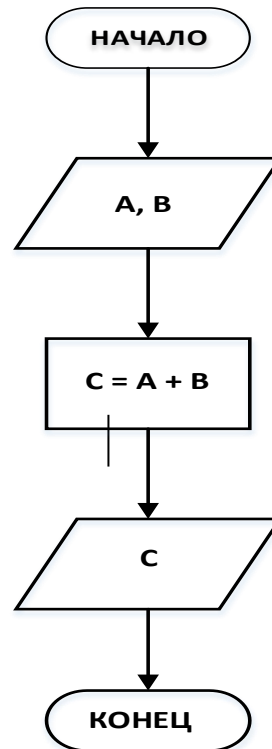


Рисунок 10 – Схема алгоритма линейной структуры

А схема оператора условной передачи управления (if A then B else C , где A – условие, B – действие, выполняющееся при истинности A , а C – действие, выполняющееся в противном случае) выглядит так (рисунок 11):

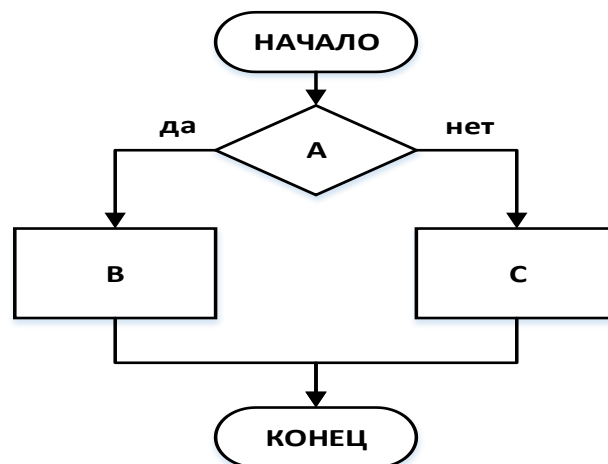


Рисунок 11 – Схема условного оператора

Для оператора цикла с известным числом повторений (for $I:=N$ to M do S) схема

выглядит следующим образом (рисунок 12):

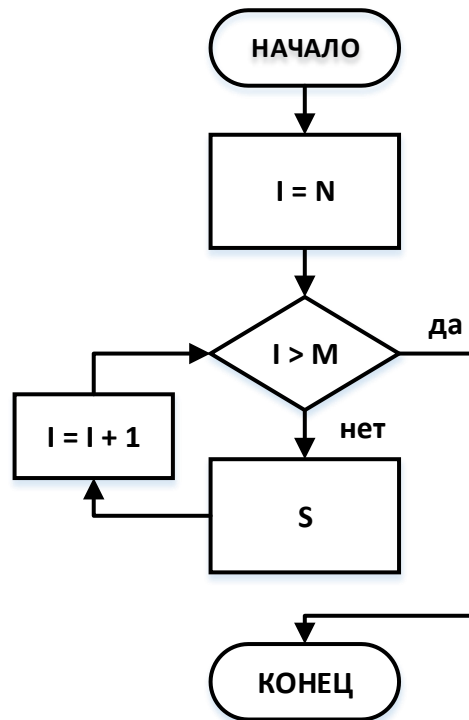


Рисунок 12 – Схема оператора цикла с известным числом повторений

Для операторов цикла с неизвестным числом повторений с предусловием (while A do S) и постусловием (repeat S until B) схемы выглядят следующим образом (рисунок 13):

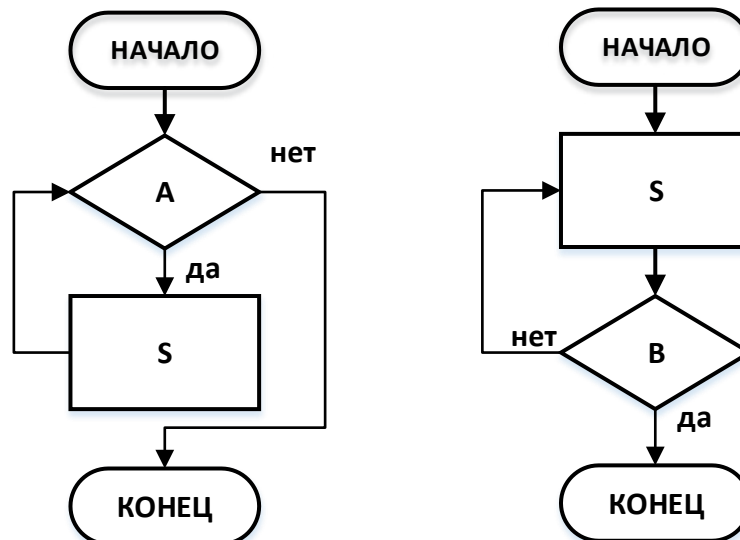


Рисунок 13 – Схемы операторов цикла с неизвестным числом повторений

На третьем этапе при выборе языка программирования необходимо учитывать тип решаемой задачи. Например, для вычислительных задач удобнее использовать язык С,

Fortran и подобные им. При разработке интернет-приложений – язык Java. Языки Pascal, Basic считаются универсальными и часто используются для обучения программированию.

Для создания безошибочной программы за приемлемое время используются основные приемы *структурного программирования*.

Суть его заключается в следующем:

Исходная сложная задача условно разбивается на более простые подзадачи, которые являются относительно независимыми друг от друга. Каждая из этих задач программируется в отдельной программе-модуле. Эти прикладные модули объединяются в единое целое специальным управляющим модулем, который может входить в группу подобных модулей (в случае решения сложных задач), объединённых основным управляющим модулем. В результате получается структурированная иерархическая система – программа, представляющая собой композицию из последовательных или вложенных друг в друга модулей.

Принципы разбиения на подзадачи-модули:

- доступность восприятия;
- незначительный размер (желательно не более 100 строк программы);
- учет возможностей изменения модуля в дальнейшем;
- учет наличия уже готовых модулей.

Модульный подход имеет положительные стороны:

- упрощение создания и дальнейшей модификации программ;
- создание библиотеки модулей;
- возможность параллельной работы с несколькими модулями одновременно;
- уменьшение объема, занимаемой ОП компьютера.

На уровне прикладных модулей при программировании используются три основные *базовые (управляющие) конструкции*, которые могут изменять ход вычислительного процесса:

- *Конструкция следования* (например, оператор GOTO);
- *Конструкции ветвления*:
 - конструкция условного ветвления (IF);
 - конструкция выбора (CASE).

Обе эти конструкции могут быть полные и неполные (без ELSE).

- *Конструкции повторения:*
 - с известным числом повторений (FOR);
 - с неизвестным числом повторений:
 - ✓ с предусловие (WHILE);
 - ✓ с постусловие (REPEAT).

Существует два метода создания многомодульных пакетов программ:

1. Метод восходящего проектирования.

Суть его заключается в том, что каждая прикладная подзадача программируется в отдельном модуле, который отдельно компилируется, тестируется и отлаживается независимо от других модулей. После этого прикладные модули объединяются управляющими модулями, и затем происходит компиляция и отладка всей многомодульной системы. Недостаток этого метода заключается в сложности *организации связей между модулями* и в *проблемах с исправлением ошибок*, допущенных на ранней стадии программирования. Однако, этот метод приемлем при разработке широкого круга относительно несложных задач.

2. Метод нисходящего проектирования.

Этот метод используется при разработке сложных многоуровневых программ. Суть его заключается в том, что программирование начинается с разработки основного управляющего модуля. Затем программируются и подключаются вспомогательные управляющие модули и отлаживаются связи между ними. В конце к разработанной программе подключаются прикладные модули-программы. На каждом из этих этапов происходит общая компиляция и отладка всего комплекса.

Тестирование и отладка программ

Тестирование и отладка написанной программы являются содержанием четвертого этапа разработки ПО.

Тестирование – выполнение программы с целью обнаружения наличия ошибок.

Тест – совокупность специально подобранных исходных данных и соответствующих им результатов расчетов (как промежуточных, так и окончательных).

Отладка – выполнение программы с целью локализации, диагностики и исправления ошибок.

Причины возникновения ошибок:

- некорректность текста (синтаксические ошибки);
- некорректность компоновки (ошибки редактирования);
- некорректность данных (семантические ошибки);
- некорректность алгоритма (семантические ошибки).

Синтаксические ошибки проявляются на этапе компиляции (система программирования выводит сообщение об ошибке и указывает место в программе, содержащее ошибку).

После компиляции следует компоновка программы, при которой могут быть *ошибки редактирования* (неправильное использование подключаемых модулей).

Семантические ошибки могут проявляться как на этапе выполнения программы (до её завершения), так и после выполнения программы. К первым относятся такие ошибки, как, например, деление на ноль, выход за границы диапазона, нехватка памяти и т.п. О них выводится сообщение компилятором, что облегчает исправление. Семантические ошибки второго типа находить и исправлять гораздо сложнее, так как компилятор их не может найти (они связаны с погрешностями самого алгоритма).

Для поиска этих ошибок используются различные специальные приёмы. Они основаны на получении дополнительной информации о ходе вычислительного процесса.

Некоторые из этих приёмов:

1) *Слежение*:

- *трассировка* – построчное выполнение программы (клавиши F7, F8 в Turbo Delphi);

- *математическое слежение* – контроль за изменением значений определенных переменных в процессе расчёта (подсказки при наведении курсора на идентификатор при трассировке).
- 2) *Печать в узлах* – вывод значений заданных переменных в узловых точках программы (разветвление или схождение алгоритма, точки входа и выхода из подпрограммы и др.).
 - 3) *Прокрутка* – вывод значений всех переменных используемых в программе после выполнения каждого оператора в программе.

Лекция 6.

Вычислительные комплексы и сети

Обработка информации при помощи ЭВМ развивается по двум направлениям:

- с использованием *вычислительных комплексов*;
- с использованием *вычислительных сетей*.

Вычислительные комплексы служат для повышения производительности и надежности обработки информации. Они объединяют несколько ЭВМ, территориально расположенных в одном месте, и делятся на два типа:

- многомашинные комплексы (несколько самостоятельных ЭВМ, в том числе и резервных, объединенных общим управлением);
- многопроцессорные комплексы (несколько процессоров, работающих с одной общей памятью с различными возможными типами доступа к ней).

Использование вычислительных комплексов позволяет разделить поставленную задачу на несколько подзадач (если это позволяет сама задача) и решать их параллельно.

Вычислительная или компьютерная сеть (КС) – это совокупность ПО и компьютеров, соединенных с помощью каналов связи и специального сетевого оборудования (см. далее) в единую систему для распределённой обработки данных.

Компьютерные сети могут классифицироваться по разным критериям. Например, по территориальному признаку, т.е. по масштабу охвата территории, сети делят на локальные (LAN – Local Area NetWork), региональные (MAN – Metropolia Area NetWork) и глобальные (WAN – Wide Area NetWork):

- локальные сети, как правило, размещаются в одном здании или на территории одного предприятия (примером локальной сети является локальная сеть в учебном классе);

- региональные сети объединяют несколько предприятий или город (примером сетей такого типа является сеть кабельного телевидения);

- глобальные сети охватывают значительную территорию, часто целую страну или континент, и представляют собой объединение сетей меньшего размера (примером глобальной сети является сеть Интернет).

Информация в сети передаётся по каналам связи, которые могут быть:

- кабельными каналами (телефонный кабель, витая пара, коаксиальный кабель, оптоволоконный кабель);
- радиоканалами.

Для подключения к сети используется специальное оборудование - устройства сопряжения, предназначенные для согласования сигналов внутреннего интерфейса ЭВМ с сигналами сети:

- модемы (при подключении через телефонную сеть);
- сетевые адаптеры (при подключении к одному каналу);
- мультиплексоры (при подключении к нескольким каналам),

Компьютерные сети используются в следующих целях:

- совместного использования ресурсов (данных, оборудования, программ);
- обеспечения надёжного хранения данных (в разных местах);
- для передачи данных между удалёнными друг от друга пользователями.

Взаимодействие в КС происходит по определенным правилам – *протоколам*, которые обеспечивают подключение к сети разнотипных ЭВМ с различными ОС.

Основные характеристики компьютерных сетей:

- скорость передачи (Мбит/с);
- достоверность передачи информации (ошибок/знак);
- надёжность (среднее время безотказной работы в сетях).

Компьютеры сети могут быть *серверами* и *клиентами* (рабочими станциями).

Сервер – компьютер, обеспечивающий пользователей сети ресурсами (оборудованием, данными и программами, выполняющими задания пользователей). Серверы могут быть *файловыми* (предназначены для хранения и обработки большого объема данных для всех пользователей), *выделенными* (на них устанавливается общая сетевая ОС и общие внешние устройства – принтеры, модемы, винчестеры т.п.), а также – одновременно файловыми и выделенными.

Клиент – компьютер, через который пользователь получает доступ к сети.

Компьютеры, объединенные в локальную сеть, физически могут располагаться различным образом. Однако порядок их подсоединения к сети определяется топологией – усредненной геометрической схемой соединений узлов сети.

Наиболее распространенными топологиями локальных сетей, в которых передающей средой является кабель, являются кольцо, шина, звезда (рисунки 14, 15 и 16).

Топология кольцо предусматривает соединение узлов сети замкнутым контуром и используется для построения сетей, занимающих чаще всего сравнительно небольшое пространство. Выход одного узла сети соединяется с входом другого. Информация по кольцу передается от узла к узлу в одном направлении. Каждый промежуточный узел

ретранслирует посланное сообщение. Принимающий узел распознает и получает только адресованное ему послание.

Последовательная организация обслуживания узлов сети снижает ее быстродействие, а выход из строя одного из узлов может привести к нарушению функционирования кольца (при отсутствии дополнительного контура).

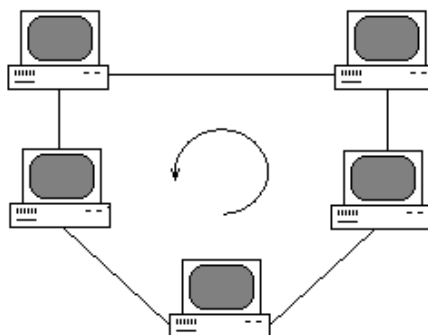


Рисунок 14 - Топология кольцо

При топологии *шина* узлы подключены к одной передающей линии - шине. Они передают свои сообщения по очереди в режиме распределенного по времени интерфейса. Сообщение от каждого узла распространяется по шине в обе стороны. Оно поступает на все узлы, но принимает его только тот узел, которому оно адресовано. Узлы не ретранслируют сообщение, поэтому выход из строя одного узла не приводит к нарушению функционирования сети. Производительность сети зависит от количества узлов в сети (при увеличении количества узлов она уменьшается), так как в каждый момент времени передачу может вести только один узел.

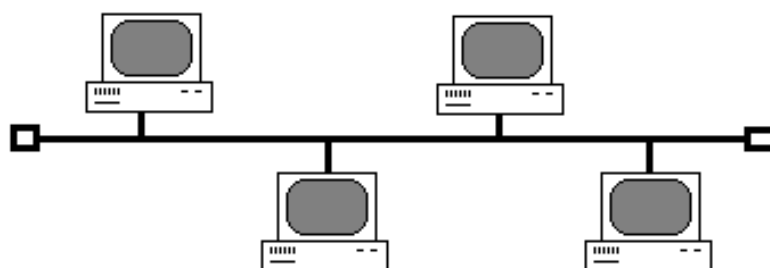


Рисунок 15 - Топология шина

При топологии *звезда* все устройства сети связаны с центральным узлом, который ретранслирует, коммутирует и маршрутизирует (находит путь от источника к приёмнику) все передаваемые данные. В качестве центрального узла может выступать либо *концентратор* (hub), который передаёт сообщение широкоэвещательно (на все узлы), а воспринимает его только узел - приёмник, либо - коммутатор (switch), который передаёт сообщение только приёмнику (за счет чего повышается пропускная способность).

Данная топология значительно упрощает взаимодействие узлов сети друг с другом, но в то же время работоспособность локальной вычислительной сети зависит от надежности работы центрального узла.

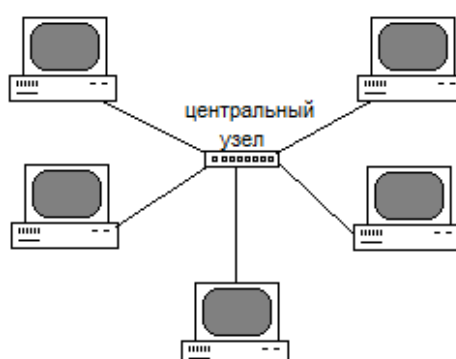


Рисунок 16 - Топология звезда

При построении реальных вычислительных сетей используются эти топологии, а так же их сочетания.

Сеть Интернет

Сеть Интернет – глобальная компьютерная сеть, точнее - сообщество сетей. В состав его на добровольной основе входят различные региональные и локальные сети. У этого сообщества нет единого центра управления.

Протоколы Интернета можно разделить на два типа:

- 1) *базовые* (обеспечивают физическую передачу сообщений между узлами в сети – протоколы нижнего уровня):
 - протокол TCP - используется для управления передачей данных (регулировка, синхронизация, организация их в виде пакетов);
 - протокол IP - используется для определения адресов получателей сообщений;
- 2) *прикладные* (обеспечивают функционирование служб сети Интернет – протоколы высокого уровня):
 - протокол HTTP – служит для передачи гипертекстовых документов;
 - протокол FTP – используется для передачи файлов;
 - протокол SMTP – используется для передачи электронной почты.

Каждому компьютеру, подключенному к сети Интернет (даже временно), присваивается *числовой адрес*, называемый *IP-адресом*. IP-адрес содержит информацию, необходимую для идентификации узла в сети. Он состоит из четырех чисел, разделенных точками.

IP-адрес трудно запоминаем пользователем, поэтому некоторые узлы в сети Интернет имеют символьные DNS-адреса (Domain Name System – система доменных имен), например, `www.site.net`. В сети Интернет существуют специальные DNS-серверы, которые по DNS-адресу выдают его IP-адрес. DNS-адрес может иметь произвольную длину, образуется как символьный адрес в локальной сети и включает в себя несколько уровней доменов. Уровни доменов разделяются точками. Самый правый домен – домен верхнего уровня. Чем левее домен, тем ниже его уровень.

Для доступа к ресурсам расположенных в сети компьютеров используется унифицированный указатель ресурса – URL (Uniform Resource Locator). Адрес URL является сетевым расширением понятия полного имени ресурса, например, файла или приложения и пути к нему в ОС. В адресе URL, кроме имени файла и директории, где он находится, указывается сетевое имя компьютера, на котором этот ресурс расположен, и протокол доступа к ресурсу, который можно использовать для обращения к нему. Ресурсы предоставляются только для чтения и копирования.

Информация в сети передается небольшими порциями – пакетами (группами байт фиксированной длины). Любой Клиент и любой Сервер умеют преобразовывать поток передаваемой информации в набор отдельных пакетов и "склеивать" полученные пакеты обратно в поток информации. Обычно размер пакетов в сети небольшой - от нескольких байт до нескольких килобайт.

Каждый пакет состоит из заголовка и информационной части. Заголовок - это аналог почтового конверта. В заголовке указывается, кому и от кого этот пакет передан - адрес отправителя пакета и адрес получателя, а также иная служебная информация, необходимая для успешной "склейки" пакетов получателем. В информационной части - собственно сама передаваемая информация. Адреса отправителя/получателя в заголовке пакета используется сетевым оборудованием для определения - куда какой пакет отправлять.

Применение пакетной передачи данных позволяет повысить надежность передачи информации и строить сеть таким образом, что маршруты доставки от одной точки сети до другой пакетов информации могут проходить по разным физическим каналам связи и, меняться в зависимости от их работоспособности или загрузки. Это значительно увеличивает "живучесть" сети в целом - даже если часть каналов связи будут неработоспособными, информация все равно может быть доставлена по другим работающим каналам.

Основные популярные сервисы сети Интернет:

- почтовая служба (e-mail);
- информационный сервис (www);
- служба передачи файлов (ftp).

E-mail предназначена для обмена электронными письмами между пользователями. Она построена по принципу клиент-серверной архитектуры (пользователь работает с клиентской программой, которая взаимодействует с сервером почтового сервиса – mail.ru, gmail.com, yandex.ru, rambler.ru и т.п.). Зарегистрировавшись на сервере, пользователь получает адрес, который имеет следующий формат - <логин пользователя>@<имя почтового домена>. Используется в почтовой службе SMTP-протокол (Simple Mail Transfer Protocol – протокол пересылки почты).

WWW-сервис является основной информационной службой Интернета, которая охватывает всю глобальную сеть («world-wide-web» - «всемирная паутина»). Информация в сети представляется в виде гипертекстовых документов (созданных с помощью языка HTML) - *web-страниц*. Располагаются эти документы на специальных *web-серверах*.

Совокупность web-страниц, объединённых общей тематикой и связанных гиперссылками, - *web-сайт*.

Web-сайт широкой тематики, содержащий сотни тысяч страниц и различные дополнительные сервисы (новости, почта, обсуждение, голосование и др.) – *портал*.

Сайт, содержащий самостоятельно обновляемую пользователем информацию личного характера – *блог*.

Сайт, на котором можно общаться (и не только в реальном времени) по определённой тематике – *форум*.

Средство общения в реальном времени – *чат*.

Для работы в сети пользователю необходима специальная программа-обозреватель – *браузер*. Браузер (Internet Explorer, Mozilla, Firefox, Opera, Google Chrome и т.п.) по требованию пользователя обеспечивает запрос информационного ресурса по его URL у web-сервера, на котором он хранится и отображает его содержимое пользователю. При этом используется HTTP-протокол (HyperText Transfer Protocol – протокол передачи гипертекста) или HTTPS-протокол (HyperText Transfer Protocol Secure) - расширение протокола HTTP, поддерживающее шифрование.

FTP-сервис используется для удобной передачи файлов большого размера (программ, изображений, видеофайлов). Хранятся такие файлы на специальных ftp-

серверах, для доступа к которым используются специальные программы, пересылающие файлы по ftp-протоколу (file transfer protocol – протокол передачи файлов).

Лекция 7.

Базы данных

Данные – информация, зафиксированная в определённой форме, пригодной для обработки, хранения, передачи.

База данных (БД) – совокупность определенным образом связанных данных, описывающая некоторую предметную область (часть реального мира, представляющую интерес для исследования и использования). База данных – современная форма хранения и доступа к информации. Базы данных предназначены для централизованного накопления и коллективного многоцелевого использования информации. Их использование позволяет ускорить процесс поиска и обработки информации, существенно уменьшить документооборот.

Основные требования, предъявляемые к базам данных:

- 1) полнота;
- 2) непротиворечивость;
- 3) отсутствие дублирования;
- 4) актуальность информации;
- 5) защищённость от разрушения;
- 6) возможность быстрого и полного восстановления.

Данные хранятся в БД в соответствии с моделью данных. Существуют различные типы моделей данных, например: сетевая, иерархическая, реляционная. Наибольшее распространение получила реляционная модель, в которой данные хранятся в виде двумерных таблиц.

Объекты предметной области и связи между ними

При разработке базы данных сначала исследуется предметная область (например, «Университет»). В ней выделяются основные объекты. Они могут быть реальными («Студент») или абстрактными («Дисциплина»). Каждый объект характеризуется набором свойств – *атрибутов объекта (поля данных)*. Для каждого объекта атрибуты заполняются определенными значениями. Атрибуты могут быть простыми и ключевыми.

Ключевой атрибут (ключ) – это отдельные элементы данных, по которым можно определить все остальные элементы данных («Номер зачетной книжки»). Ключ может быть простым или составным («Фамилия», «Имя», «Отчество»).

После определения основных объектов предметной области с помощью их ключевых атрибутов устанавливаются связи между этими объектами:

- а) 1:1 («один к одному») – каждому экземпляру объекта А соответствует только один экземпляр объекта В и наоборот (рисунок 17).

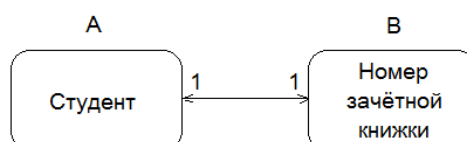


Рисунок 17 – Связь «один к одному»

- б) 1:М («один ко многим») – каждому экземпляру объекта А может соответствовать 0, 1 или несколько экземпляров объекта В, однако каждому экземпляру объекта В соответствует только 1 экземпляр объекта А (рисунок 18).

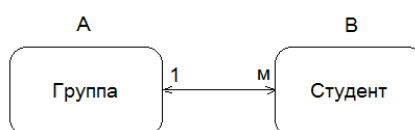


Рисунок 18 – Связь «один ко многим»

- с) М:М («многие ко многим») – каждому экземпляру объекта А соответствует 0, 1 или несколько экземпляров объекта В и наоборот (рисунок 19).

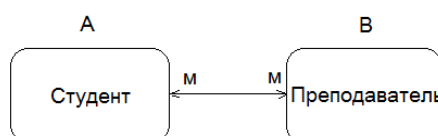


Рисунок 19 - Связь «многие ко многим»

Выделенные основные объекты предметной области с установленными связями между ними представляют собой *инфологическую модель*.

Отношения

Объект предметной области может быть представлен в виде таблицы-отношения – таблицы особого рода, у которой:

- каждая строка содержит информацию об одном экземпляре объекта (строка отношения - *кортеж*);
- все столбцы однородные, то есть все элементы в столбце имеют одинаковый тип и длину, имеют имя и содержат информацию об отдельном атрибуте объекта;
- каждый элемент представляет собой один элемент данных об объекте;
- все строки и столбцы уникальны (нет повторений);
- в таблицах нет пустых ячеек.

Базы данных, основанные на таблицах-отношениях, называются *реляционными* (*relation - отношение*). Набор отношений (таблиц) используется в БД для хранения информации об объектах реального мира и моделирования связей между ними. Например, для хранения объекта «студент» используют отношение **СТУДЕНТ**, в котором свойства объекта располагаются в столбцах таблицы, являющихся атрибутами объекта (таблица 8):

Таблица 8 – Отношение **СТУДЕНТ**

Фамилия	Возраст	Группа
Петров	17	МТЗ-12
Иванова	16	МТЗ-12
Сидоров	17	МТ10-12

Список имен атрибутов отношения называется *схемой отношения*. Схему отношения **СТУДЕНТ** можно записать так: **СТУДЕНТ = (Фамилия, Возраст, Группа)**.

Реляционная БД – набор взаимосвязанных отношений. Каждое отношение (таблица) в ЭВМ представляется в виде файла записей.

Над таблицами - отношениями можно выполнять восемь различных операций теории множеств и реляционной алгебры (объединение, выборка, проекция, пересечение, сложение, умножение, разность, деление). Вследствие этого из введенных (базовых) отношений можно получать много новых (вычисляемых) таблиц - отношений (отчетов, выборок, запросов и т.п.).

Благодаря тому, что информация в базах данных представлена в двух видах – хранимая информация (исходные, введенные таблицы) и вычисляемая информация (таблицы, полученные на основании исходных), можно существенно экономить память и ускорить процесс обработки этой информации.

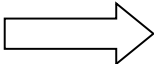
Для создания простой и надёжной базы данных необходимо нормализовать отношения. *Нормализация отношений* – пошаговый процесс разложения отношений на

более мелкие и простые. Не смотря на увеличение при этом количества отношений, операции доступа к данным существенно ускоряются благодаря улучшению корректности, устранению дублирования и обеспечению непротиворечивости данных в базе.

Существует несколько *нормальных форм*:

1-я нормальная форма. Отношение считается находящимся в первой нормальной форме, если все его атрибуты – неделимые (простые). К примеру, приведенное ниже на рисунке 20 отношение не нормализовано, поскольку содержит сложный атрибут **Спорт**. Чтобы привести это отношение к нормализованному виду, нужно избавиться от этого сложного атрибута.

Фамилия	Группа	Спорт	
		Вид	Разряд
Иванов	МТЗ-12	Шахматы	МС
		Хоккей	1 разряд
Петров	МТ10-12	Футбол	2 разряд
		Шашки	КМС

Фамилия	Вид спорта	Группа	Разряд
Иванов	Хоккей	МТЗ-12	1 разряд
Иванов	Шахматы	МТЗ-12	МС
Петров	Футбол	МТ10-12	2 разряд
Петров	Шашки	МТ10-12	КМС

Рисунок 20 – Приведение к первой нормальной форме

В полученном отношении ключ является составным, состоящим из атрибутов **Фамилия** и **Вид спорта**.

2-я нормальная форма. Отношение считается находящимся во второй нормальной форме, если все его атрибуты зависят от составного ключа в целом, а не от его частей. Следовательно, если отношение находится в первой нормальной форме и имеет простой, а не составной ключ, то оно автоматически находится и в первой, и во второй нормальной форме.

Например, в отношении **ВЕДОМОСТЬ** (рисунок 21), имеющем составной ключ «**Студент, Дисциплина**», атрибут **Лектор** зависит только от **Дисциплины**, а не от всего

ключа. Это отношение можно нормализовать, «разбив» его на два отношения **УСПЕВАЕМОСТЬ** и **ПРЕПОДАВАТЕЛЬ**:



Рисунок 21 – Приведение ко второй нормальной форме

3-я нормальная форма. Отношение считается находящимся в третьей нормальной форме, если устранены зависимости между не ключевыми атрибутами (транзитивные зависимости). Например, в отношении **ПРЕДМЕТ = (Название, Лектор, Кафедра, Телефон)** не ключевой атрибут **Телефон** зависит от не ключевого атрибута **Кафедра**.

Для устранения транзитивной зависимости необходимо «расщепить» исходное отношение на два **ДИСЦИПЛИНА = (Название, Лектор, Кафедра)** и **ДАННЫЕ КАФЕДРЫ = (Кафедра, Телефон)**.

Дальнейшее упрощение таблиц связано с дальнейшим ограничением типов зависимости между атрибутами отношений.

После нормализации отношений и установления связей между ними формируется инфологическая модель предметной области. Ниже (на рисунке 22) представлен пример инфологической модели фирмы, оформляющей сделки с заказчиками через своих сотрудников-менеджеров:

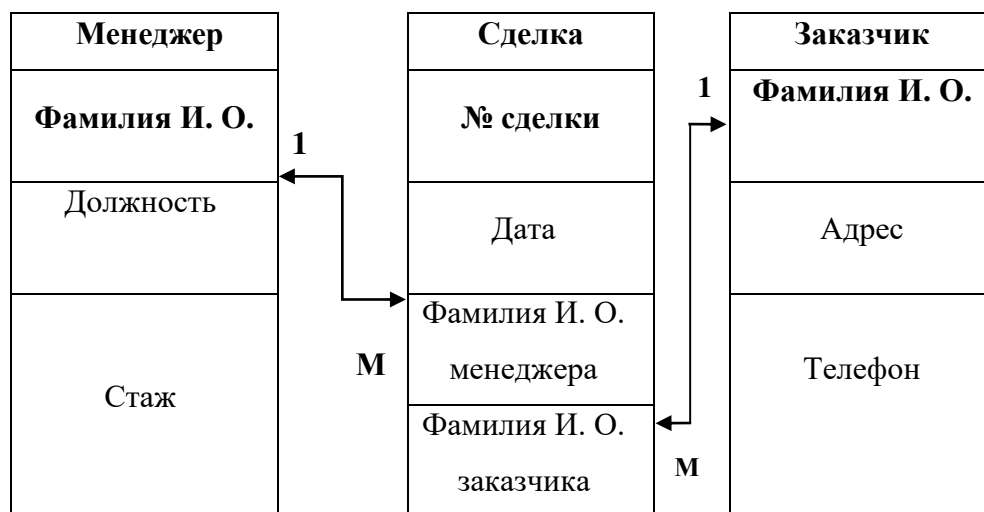


Рисунок 22 – Модель фирмы

На основании инфологической модели разрабатывается модель данных, которая дает описание логической структуры базы данных на языке описания данных (ЯОД), – *даталогическая модель (ДМ)*.

Для привязки ДМ к среде хранения используется модель данных физического уровня – *физическая модель (ФМ)*. На этом этапе физического проектирования базы данных осуществляется выбор типа носителя, разрабатывается формат хранимых записей и проектируются методы доступа к данным.

СУБД

После этого уже возможно формирование (заполнение) базы данных и непосредственно работа с ней. Работа с базами данных сводится к выполнению следующих операций:

- 1) запись (заполнение базы данных);
- 2) просмотр;
- 3) редактирование (добавление, удаление, исправление);
- 4) выборка (запросы, отчеты).

Эти операции накопления и манипулирования данными выполняет специальная программа – *система управления базами данных (СУБД)*.

По технологии решения задач, выполняемых СУБД, базы данных можно разделить на два вида:

- централизованная БД (хранится целиком на ВЗУ одной вычислительной системы и, если система входит в состав сети, то возможен доступ к этой БД других систем);

- распределенная БД (состоит из нескольких, иногда пересекающихся или дублирующих друг друга БД, хранящихся на ВЗУ разных узлов сети).

СУБД предоставляет доступ к данным БД двумя способами:

- локальный доступ (предполагает, что СУБД обрабатывает БД, которая хранится на ВЗУ того же компьютера);

- удаленный доступ (это обращение к БД, которая хранится на одном из узлов сети).

Удаленный доступ может быть выполнен по технологии файл-сервер или клиент-сервер. Технология файл-сервер предполагает выделение одной из вычислительных систем, называемой сервером, для хранения БД. Все остальные компьютеры сети (клиенты) исполняют роль рабочих станций, которые копируют требуемую часть централизованной БД в свою память, где и происходит обработка. Технология клиент-

сервер предполагает, что сервер, выделенный для хранения централизованной БД, дополнительно производит обработку запросов клиентских рабочих станций. Клиент посылает запрос серверу. Сервер пересылает клиенту данные, являющиеся результатом поиска в БД по ее запросу.

Система управления базами данных – совокупность программных и языковых средств.

Программные средства обеспечивают организацию ввода, обработки и хранения данных, а также обеспечивают взаимодействие всех частей системы при её функционировании (настройка, тестирование, восстановление).

Языковые средства обеспечивают взаимодействие пользователя с базой данных.

К ним относятся:

- языки манипулирования данными (ЯМД) – языки запросов к БД, представляющие собой систему команд для работы с данными (выборка, запрос, вставка, удаление и т.п.);
- языки определения данных (ЯОД) – языки, предназначенные для создания схемы базы данных (описания типов данных, структуры базы, взаимодействия и связей между элементами).

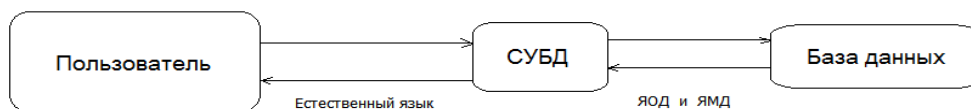


Рисунок 22 - Схема взаимодействия пользователя с базой данных

Современная СУБД – прикладная программа, которая предназначена для облегчения работы неквалифицированного пользователя с БД. Он работает с ней на естественном языке без знания языка манипулирования данными и языка определения данных (рисунок 22). Одним из примеров такой СУБД является широко известный продукт фирмы Microsoft – СУБД Access.

Лекции 8, 9.

Некоторые приёмы программирования

Наиболее часто используемые приёмы программирования, применяемые студентами на практических занятиях, рассматриваются на примерах обработки числового массива A с количеством элементов N . Ниже представлены фрагменты программ на языке **Pascal** для каждого из предложенных приемов.

Вычисление суммы и произведения элементов массива:

```

.....
S:=0;           // сумма
P:=1;           // произведение
FOR I:=1 TO N DO
  BEGIN
    S:= S+A[I];
    P:= P*A[I];
  END;
.....

```

Нахождение максимального (минимального) элемента массива (на примере нахождения максимального элемента):

```

.....
IMAX:=1;
FOR I:=2 TO N DO
  IF A[I]>A[IMAX]
    THEN
      IMAX:=I;
.....

```

Здесь $IMAX$ – номер максимального элемента;
 $A[IMAX]$ – максимальный элемент.

Удаление заданного (K-го) элемента массива:

```

.....
FOR I:=K TO N-1 DO
  A[I]:=A[I+1];
N:=N-1;
.....

```

Вставка нового элемента (M) на заданное (K-е) место в массив:

```

.....
N:=N+1;
FOR I:=N DOWNTO K+1 DO
  A[I]:=A[I-1];
A[K]:=M;
.....

```

Сортировка массива**1. Метод обмена («метод пузырька»):**

Суть данного метода заключается в сравнении соседних элементов массива, начиная с первой и до последней пары. Если предыдущий элемент пары больше последующего (при сортировке по возрастанию), то соседи меняются местами. Для полной сортировки необходимо сделать $N-1$ таких «проходов» по массиву.

```

.....
FOR J:= 1 TO N-1 DO
  FOR I:= 1 TO N-1 DO
    IF A[I] > A[I+1]
      THEN
        BEGIN
          BUF:= A[I];
          A[I]:= A[I+1];
          A[I+1]:= BUF;
        END;
.....

```

Здесь *BUF* – буферная переменная для перестановки;

Метод обмена можно ускорить двумя способами:

а) уменьшать на каждом проходе количество сравниваемых пар:

```

.....
FOR J:= 1 TO N-1 DO      // или  FOR J:=N-1 DOWNT0 1 DO
FOR I:= 1 TO N-J DO      //      FOR I:=1 TO J DO
    IF A[I] > A[I+1]
    THEN
        BEGIN
            BUF:= A[I];
            A[I]:= A[I+1];
            A[I+1]:= BUF;
        END;
.....

```

б) прекращать сортировку при окончании перестановок:

```

.....
K:=N;
REPEAT
    PORYADOK:=TRUE;
    K:= K-1;
    FOR I:= 1 TO K DO
        IF A[I] > A[I+1]
        THEN
            BEGIN
                BUF:= A[I];
                A[I]:= A[I+1];
                A[I+1]:= BUF;
                PORYADOK:=FALSE;
            END;
    UNTIL PORYADOK;
.....

```

2. Метод выбора:

Суть метода выбора: при сортировке по возрастанию находится минимальный элемент в диапазоне от первого до последнего и меняется местами с первым. Далее находится минимальный элемент от второго до последнего и меняется со вторым и т.д.

```

.....
FOR I:= 1 TO N-1 DO
BEGIN
  MIN:= A[I];
  IMIN:=I;
  FOR J:= I+1 TO N DO
    IF A[J] < MIN
      THEN
        BEGIN
          MIN:= A[J];
          IMIN:= J;
        END;
  A[IMIN]:= A[I];
  A[I]:= MIN;
END;
.....

```

3. Метод вставки:

Считается, что массив разделён на две части: отсортированную (в начале массива) и неотсортированную (в остальной части массива). В самом начале отсортированная часть состоит из первого элемента. Первый элемент из неотсортированной части (его номер – I) вставляется в отсортированную часть без изменения порядка (в позицию J):

```

.....
FOR I:= 2 TO N DO
  BEGIN
    BUF:= A[I];
    J:= 1;
    WHILE BUF > A[J] DO
      J:= J+1;
    FOR K:= I DOWNTO J+1 DO
      A[K]:= A[K-1];
    A[J]:=BUF;
  END;
.....

```

Поиск в массиве

1. Метод перебора:

Просматриваются элементы массива, начиная с первого, и сравниваются с искомым значением до тех пор, пока не произойдёт совпадение или не будет просмотрен весь массив.

```

.....
I:= 0;
NAIDEN:= FALSE;
REPEAT
  I:= I+1;
  IF A[I] = ISKOMOE
    THEN
      NAIDEN:=TRUE;
UNTIL NAIDEN OR (I=N);
IF NAIDEN
  THEN
    WRITELN('Элемент найден, его номер - ',I)
  ELSE
    WRITELN('Элемент не найден');
.....

```

2. Метод бинарного поиска:

Метод бинарного поиска (метод деления пополам) используется только для упорядоченных массивов. Суть его заключается в том, что находится центральный (серединный) элемент массива и сравнивается с искомым. Если они равны, то поиск прекращается. Если они не равны, то, если искомый элемент больше центрального (при сортировке по возрастанию), из рассмотрения исключается половина массива от первого до центрального элемента включительно. Если же искомый элемент меньше центрального, то исключается часть массива, начиная от центрального до последнего элемента. В остальной части находится центральный элемент и сравнивается с искомым и т.д. до тех пор, как произойдёт совпадение или начало области поиска станет больше её конца.

```

.....
NAIDEN:= FALSE;

```

```
NA:= 1; // номер первого элемента области поиска
KO:= N; // номер последнего элемента области поиска
REPEAT
    SR:=(KO-NA) DIV 2+NA;
    IF A[SR] = ISKOMOE
        THEN
            NAIDEN:=TRUE
        ELSE
            IF ISKOMOE > A[SR]
                THEN
                    NA:= SR+1
                ELSE
                    KO:= SR-1;
UNTIL NAIDEN OR (NA>KO);
IF NAIDEN
    THEN
        WRITELN('Элемент найден, его номер - ',SR)
    ELSE
        WRITELN('Элемент не найден');
```

Вопросы для самопроверки

1. Что такое информация?
2. Каковы задачи информатики?
3. Что такое информационные технологии?
4. Сколько было информационных революций? Какова их суть?
5. Что такое информационный кризис и информатизация общества?
6. Чем отличается информация от данных?
7. Какие существуют формы представления информации?
8. Какие бывают системы счисления?
9. Как перевести числа из десятичной в двоичную систему счисления?
10. Сколько этапов развития вычислительной техники?
11. Что такое ЭВМ (компьютер)?
12. Какие существуют типы классификации ЭВМ?
13. Что входит в состав ЭВМ?
14. Какие существуют типы устройств ввода ЭВМ?
15. Какие существуют типы устройств вывода ЭВМ?
16. Какое назначение у основной памяти ЭВМ?
17. Какие существуют типы внешних запоминающих устройства ЭВМ?
18. Что входит в состав центральных устройств ЭВМ?
19. Как обрабатывается машинная команда центральными устройствами?
20. Как взаимодействуют центральные и внешние устройства ЭВМ?
21. Какие существуют типы интерфейса?
22. Что такое шина? Каковы её основные характеристики и типы?
23. Что собой представляет обобщенная структурная схема персонального компьютера?
24. Что такое программное обеспечение ЭВМ? Каковы его основные типы и состав?
25. Что такое операционная система? Каковы её основные функции и виды?
26. Какие существуют типы диалога пользователя с компьютером?
27. Что такое система программирования? Каково её назначение и состав?
28. Каковы основные этапы разработки программных комплексов?
29. В чем заключаются основы структурного программирования?
30. Какие существуют базовые управляющие конструкции?

31. В чем суть «восходящего» и «нисходящего» способов проектирования программ?
32. Что такое алгоритм и схема алгоритма?
33. В чем отличие тестирования и отладки программ?
34. Какие существуют типы ошибок в программах?
35. Какие существуют методы получения дополнительной информации о процессе выполнения программы?
36. Какие существуют типы вычислительных комплексов? Для чего они предназначены?
37. Какие известны типы компьютерных сетей? Из чего они состоят? Каковы их основные характеристики?
38. Какие известны типы топологии компьютерных сетей?
39. Какова структура сети Интернет?
40. Что такое протокол сети?
41. Какие типы адресов компьютера существуют в сети Интернет?
42. Что такое унифицированный указатель ресурса?
43. Какие существуют основные службы сети Интернет?
44. Что такое базы данных, и каково их назначение?
45. Каковы основные требования к базам данных?
46. Что такое предметная область и её объект?
47. Какие типы связей могут быть между объектами предметной области?
48. Что такое отношение и реляционная база данных?
49. В чем суть нормализации отношений?
50. Что такое инфологическая модель предметной области?
51. Какова схема взаимодействия пользователя с базой данных?
52. Что такое система управления базами данных?
53. Как можно оптимизировать сортировку массива методом обмена («пузырька»)?
54. В чём суть сортировки массива методом выбора?
55. В чём суть сортировки массива методом вставки?
56. В чём суть поиска в массиве методом перебора?
57. В чём суть и особенности метода бинарного поиска?

Заключение

Основная задача информатики заключается в определении общих закономерностей процессов обработки информации: создания, передачи, хранения и использования в различных сферах человеческой деятельности. Прикладные задачи связаны с разработкой методов, необходимых для реализации информационных процессов с использованием технических средств.

После освоения дисциплины «Информатика», основу которой наряду с лекциями составляют и практические занятия, студент должен приобрести определенные знания, умения и владения.

Студент должен знать:

- методы представления информации в ЭВМ и выполнения арифметических и логических операций над двоичными числами;
- принципы работы технических и программных средств в информационных системах;
- типовые алгоритмы решения задач;
- язык программирования;
- среду программирования.

Студент должен уметь:

- разрабатывать алгоритмы и кодировать их на языке программирования;
- проводить оценку функциональных возможностей компьютеров;
- использовать современные информационные технологии и инструментальные средства для решения различных задач.

Студент должен иметь навыки:

- самостоятельной работы с учебной и справочной литературой;
- использования программных комплексов и прикладных программ вычисления на компьютере;
- разработки алгоритмов и кодирования приложений для решения профессиональных задач;
- тестирования и отладки приложений;
- представления результатов в удобном для пользователя виде, создания диалоговых и графических приложений.

Список литературы

1. Парфилова Н. И., Пруцков А. В., Пылькин А. Н., Трусов Б. Г. Информатика и программирование. Основы информатики: Учебник для студ. учреждений высш. проф. образования / Под ред. Б.Г. Трусова – М.: Издательский центр «Академия», 2019. – 60 с.
2. Исаев А.Л., Чеповский А.М. Введение в теорию баз данных: Учебно-методическое пособие.- М.: МГТУ. – 2017.
3. Бройдо В.Л., Ильина О.П. Вычислительные системы, сети и телекоммуникации. – 4-е изд. – СПб.: Питер, 2019. – 560 с.
4. Информатика. Общий курс: учеб. / А.Н. Гуда, М.А. Бутакова, Н.М. Нечитайло, А.В. Чернов; под ред. академика РАН В.И. Колосникова. – 4-е изд. – М.: Издательско-торговая корпорация «Дашков и К»; Ростов н/Д: Наука-Спектр, 2011. – 400 с.
5. Информатика: учеб. / Б.В. Соболев, А.Б. Галин, Ю.В. Панов и др. – 3-е изд., доп. и перераб. – Ростов н/Д: Феникс, 2017. – 446 с.
6. Информатика: учеб. пособие / Н.И. Иопа; Рязан. гос. радиотехн. акад. – Рязань, 2005. – 216 с.
7. Максимов Н.В., Партыка Т.Л., Попов И.И. Архитектура ЭВМ и вычислительных систем: учеб. – 3-е изд., перераб и доп. – М.: Форум, 2010. – 512 с.
8. Мельников В.П. Информационные технологии: учеб. для студ. вузов. – М.: Издательский центр «Академия», 2008. – 432 с.
9. Могилев А.В., Пак Н.И., Хеннер Е.К. Информатика: учеб. пособие для студ. пед. вузов / под ред. Е.К. Хеннера. – 3-е изд., перераб. и доп. – М.: Издательский центр «Академия», 2004. – 848 с.
10. Новые информационные технологии. учеб. пособие / под ред. проф. В.П. Дьяконова. – М.: СОЛОН-Пресс, 2005. – 640 с.
11. Острейковский В.А. Информатика: учеб. для вузов. – 5-е изд., стер. – М.: Высш. шк., 2009. – 511 с.