**ВОПРОСЫ ДЛЯ ВСТУПИТЕЛЬНОГО ЭКЗАМЕНА**

по направлению подготовки

**«Информатика и вычислительная техника»**

**(направления подготовки: 02.04.03; 09.04.01; 09.04.02; 09.04.03; 09.04.04)**

для поступающих на обучение по образовательным программам

высшего образования – программам магистратуры

**Вопрос №1**

1. Переведите число 321 из десятичной системы счисления в двоичную.

32110 = 1010000012

1. Переведите число 753 из десятичной системы счисления в двоичную.

75310 = 10111100012

1. Переведите число 472 из десятичной системы счисления в двоичную.

47210 = 1110110002

1. Переведите число 455 из десятичной системы счисления в двоичную.

45510 = 1110001112

1. Переведите число 11101001 из двоичной системы счисления в десятичную.

111010012 = 23310

1. Переведите число 01001011 из двоичной системы счисления в десятичную.

010010112 = 7510

1. Переведите число 10101101 из двоичной системы счисления в десятичную.

101011012 = 17310

1. Переведите число 11001000 из двоичной системы счисления в десятичную.

110010002 = 20010

1. Переведите число 01001011 из двоичной системы счисления в шестнадцатеричную.

0100 10112 = 4Bh

1. Переведите число 10101101 из двоичной системы счисления в шестнадцатеричную.

1010 11012 = ADh

1. Переведите число 11001000 из двоичной системы счисления в шестнадцатеричную.

1100 10002 = C8h

1. Переведите число 11101001 из двоичной системы счисления в шестнадцатеричную.

1110 10012 = E9h

1. Переведите число 4AE9 из шестнадцатеричной системы счисления в двоичную.

4AE9h = 100 1010 1110 10012

1. Переведите число BC21 из шестнадцатеричной системы счисления в двоичную.

BC21h = 1011 1100 0010 00012

1. Переведите число 87FF из шестнадцатеричной системы счисления в двоичную.

87FFh = 1000 0111 1111 11112

1. Переведите число EE52 из шестнадцатеричной системы счисления в двоичную.

EE52h = 1110 1110 0101 00102

1. Переведите число 321 из восьмеричной системы счисления в шестнадцатеричную.

3218 = D1h

1. Переведите число 753 из восьмеричной системы счисления в шестнадцатеричную.

7538 = 1EBh

1. Переведите число 672 из восьмеричной системы счисления в шестнадцатеричную.

6728 = 1BAh

1. Переведите число 455 из восьмеричной системы счисления в шестнадцатеричную.

4558 = 12Dh

1. Какое минимальное количество бит данных потребуется для хранения числа 12910?

12710 = 11111112 (7 бит)

25510 = 111111112 (8 бит)

127 <= 129 <= 255

(27 – 1) <= 129 <= (28 – 1)

Ответ: 8 бит

Проверка: 12910 = 100000012

1. Какое минимальное количество бит данных потребуется для хранения числа 102310?

511 <= 1023 <= 1023

(29 – 1) <= 1023 <= (210 – 1)

10 бит

1. Какое минимальное количество бит данных потребуется для хранения числа 6210?

31 <= 62 <= 63

(25 – 1) <= 62 <= (26 – 1)

6 бит

1. Какое минимальное количество бит данных потребуется для хранения числа 78910?

511 <= 789 <= 1023

(29 – 1) <= 789 <= (210 – 1)

10 бит

1. Какое минимальное количество бит данных потребуется для хранения числа 400910?

2047 <= 4009 <= 4095

(211 – 1) <= 4009 <= (212 – 1)

12 бит

**Вопрос №2**

1. Что такое алгоритм?

Алгоритм – это полное и точное описание последовательности конечного числа действий, которые необходимо выполнить исполнителю для получения искомого результата.

1. Перечислите базовые структуры алгоритма.

Базовые структуры алгоритма - это структуры, с помощью которых создается алгоритм для решения определенной задачи.

Линейная структура процесса вычислений предполагает, что для получения результата необходимо выполнить некоторые операции в определенной последовательности.

Разветвленная структура процесса вычислений предполагает, что конкретная последовательность операций зависит от значений одной или нескольких переменных.

Циклическая структура процесса вычислений предполагает, что для получения результата некоторые действия необходимо выполнить несколько раз.

1. Назовите основные свойства алгоритма.

1. Дискретность – разбиение на шаги. Только выполнив требования одного предписания можно приступить к исполнению следующего.

2. Понятность. Исполнитель может выполнить алгоритм, если он ему понятен,

т.е. записан на понятном ему языке и содержит предписания,

которые исполнитель может выполнить.

3. Детерминированность (определенность). Алгоритмы, предназначенные для исполнения техническим устройством, не должны содержать предписаний, приводящих к неоднозначным действиям. Если алгоритм применять повторно к одним и тем же исходным данным, то всегда должен получаться один и тот же результат.

4. Массовость - применимость к некоторому классу объектов, возможность

получения результата при различных исходных данных на некоторой области допустимых значений.

5. Результативность (конечность). Цель выполнения алгоритма — получение конечного результата посредством выполнения указанных преобразований над исходными данными. Причем при точном исполнении всех предписаний алгоритмический процесс должен заканчиваться за конечное число шагов.

6. Эффективность. Если алгоритм должен быть выполнен не просто за конечное

время, а за разумное конечное время, то речь идет об эффективности

алгоритма. Эффективность алгоритма нередко определяет возможность его практической реализации.

1. Что такое идентификатор применительно к языкам программирования?

Идентификатор – имя любого объекта программы, состоящее из последовательности букв, цифр, знаков подчеркивания, начиная с буквы. Бывают:

Служебные: имена операторов (if then else…). Их нельзя менять.

Стандартные: имена предопределенных констант (PI), функций (sin cos), файлов. Нежелательно менять.

Пользовательские: применяют для обозначения имён программ, переменных, констант и других объектов, определенных в программе. Имя может состоять из любого количества букв или цифр (максимальная длина – 127 символов), но должно начинаться с буквы или символа подчёркивание

1. Что такое переменная применительно к языкам программирования?

Переменная - это именованная область памяти для хранения данных, которые могут изменяться в процессе исполнения программы.

Переменная характеризуется:

1) Именем («обозначением ячейки памяти»)

2) Значением (данными, содержащимися в переменной в конкретный момент времени)

3) Типом определяющим:

а) какие значения может принимать переменная;

б) какие операции можно производить с этими значениями;

в) как данные представлены в памяти компьютера.

1. Что такое константа применительно к языкам программирования?

Константы – данные, не меняющие своего значения. В программе представляются либо значением, либо именем.

В pascal стандартные константы: константа PI и 2 логических false/true.

В разделe Const (до Var) также описываются константы.

1. Что такое массив применительно к языкам программирования?

Массив — структура данных в виде набора однотипных элементов, расположенных в памяти непосредственно друг за другом.

1. Что такое подпрограмма?

Подпрограмма — поименованная часть компьютерной программы, содержащая описание определённого набора действий и которая может быть многократно вызвана из разных частей программы.

В pascal различают функции и процедуры.

1. Как обозначаются в языке Паскаль операции «И», «ИЛИ», «НЕ», «ИСКЛЮЧАЮЩЕЕ ИЛИ» ?

and - «И»

or - «ИЛИ»

not - «НЕ»

xor - «ИСКЛЮЧАЮЩЕЕ ИЛИ»

1. Как обозначаются в языке Паскаль операции целочисленного деления и взятия остатка от деления?

div – целочисленное деление

mod – взятие остатка от деления

1. Как описать в языке Паскаль массив A из 100 вещественных элементов?

var

A: array [1..100] of Real;

или

const

N = 100;

type

RealArray = array [1…N] of Real;

var

A: RealArray;

1. Как описать в языке Паскаль двухмерный массив А размером 10 на 20 логических элементов?

Const

N = 10;

M = 20;

Type

vector = array [1..N] of Boolean;

matrix= array [1..M] of vector;

Var

A: matrix;

1. Как описать в языке Паскаль строку из 100 символов?

Const

N = 100;

Var

s: string[N];

N - максимально возможная длина строки - целое число в диапазоне 1..255. Если этот параметр опущен, то по умолчанию он принимается равным 255.

1. Синтаксис оператора IF в языке Паскаль?

IF <условие> THEN <оператор 1>

ELSE <оператор 2>

где <условие> - выражение логического типа;

<оператор 1>, <оператор 2> - любые операторы.

1. Синтаксис оператора FOR в языке Паскаль?

FOR <счётчик цикла>:=<нач. зн.> TO <кон. зн.> DO <оператор>

где <счётчик цикла> - переменная порядкового типа, чаще всего INTEGER.

FOR <счётчик цикла>:=<нач. зн.> DOWNTO <кон. зн.> DO <оператор>

Здесь цикл будет повторяться, пока <счётчик цикла> ≥ <кон. зн.>.

1. Синтаксис оператора WHILE…DO в языке Паскаль?

Это цикл с предварительной проверкой условия повторения цикла. Его структура:

WHILE <условие> DO <оператор>

Пока проверяемое условие справедливо (TRUE) то выполняется оператор, стоящий после слова DO. Если проверяемое условие равно FALSE, то оператор WHILE заканчивает свою работу.

1. Синтаксис оператора REPEAT…UNTIL в языке Паскаль?

Это цикл с постусловием. Он имеет следующую структуру:

REPEAT

<Тело цикла>

UNTIL <условие>

Все операторы, входящие в тело цикла, повторяются, пока не выполнится условие выхода из цикла, стоящее после слова UNTIL. После этого цикл заканчивает свою работу. В тело цикла можно включать любое количество операторов, не оформляя их, как составной оператор.

1. Синтаксис оператора CASE в языке Паскаль?

Оператор выбора позволяет выбрать одно из нескольких возможных продолжений программы. Его структура:

CASE <ключ выбора> OF

<список выбора>

ELSE <оператор>

END

где <ключ выбора> - выражение порядкового типа (обычно целого или символьного); <список выбора> - одна или более конструкций следующего вида: <константа выбора> : <оператор>, где <константа выбора> - константа того же типа, что и выражение <ключ выбора>.

1. Как описать запись в языке Паскаль?

Объявление структуры записи производится в разделе описания типов TYPE или переменных VAR.

VAR

< имя\_записи > : RECORD

< список полей >

END

В описании указывают имя каждого поля и его тип:

Person : RECORD

name: STRING[20];

day, month, year: INTEGER;

END;

Или

Type Person = record … end;

Var student : Person;

1. Для чего используется раздел VAR в языке Паскаль?

Var - обозначает начало блока определения переменных. Всё что будет идти за этим словом, будет рассматриваться компилятором, как объявление переменных.

Формат:

var <идентификатор > : <тип>;

1. Для чего используется раздел CONST в языке Паскаль?

Раздел описания CONST позволяет ввести в программу идентификаторы, являющиеся представителями определенных значений.

CONST

ИмяКонстанты1 = Значение1;

ИмяКонстанты2 = Значение2;

или

ИмяКонстанты = ЗначениеВыражения;

Примеры описания констант:

CONST

Min = 0; { константа - целое число }

Max = 100;{ константа - целое число }

e = 2.7; { константа - вещественное число }

SpecChar = '\'; {константа - символ }

HelpStr = 'Нажмите клавишу F1'; {константа - строка }

OK = True; { логическая константа "истина" }

1. Для чего используется раздел TYPE в языке Паскаль?

Для создания пользовательских типов в программе, объявленных в разделе TYPE по формату:

TYPE

[имя] = [тип]

Программист может создавать свои типы произвольной структурой сложности на основе стандартных типов, либо уже определенных пользователем типов. Количество создаваемых типов неограниченно.

Пример:

TYPE

Matr = аггау[1..10] of integer;

1. Для чего используется раздел LABEL в языке Паскаль?

В разделе Label объявляются метки, которые помечают операторы в соответствующей операторной части.

Синтаксис:

label идентификатор, ..., идентификатор;

Каждая метка должна помечать только один оператор.

В дополнение к идентификаторам, в качестве метки также можно использовать последовательность цифр в диапазоне от 0 до 9999.

Передать управление к метке можно с помощью оператора goto.

Пример

LABEL lbl1, lbl2;

Var s : integer;

begin

readln(s);

if (s = 0) then GOTO lbl1;

else GOTO lbl2;

lbl1 : WriteLn(‘s = 0');

lbl2 : WriteLn('s < > 0');

end.

1. С помощью какого оператора языка Паскаль можно ввести значение переменной с клавиатуры?

Общий вид:

Read(переменная, переменная...)

ReadLn(переменная, переменная...)

При выполнении процедуры Read ожидается ввод перечисленных в скобках значений. Вводимые данные нужно отделить друг от друга пробелами. Присваивание значений идет по очереди.

Часто процедуру ReadLn без параметров применяют в конце программы для задержки: до нажатия на клавишу <Enter> результат выполнения программы остается на экране.

1. С помощью какого оператора языка Паскаль можно вывести текстовое сообщение на экран?

Общий вид:

Write (список выражений)

WriteLn (список выражений)

Процедуры Write и WriteLn используются не только для вывода результата, но и для вывода различных сообщений или запросов.

Оператор WriteLn можно применить и без параметров. В этом случае курсор просто будет переведен в начало следующей строки.

Пример:

Var s : real;

Begin

s := 5.5;

writeln(‘число s = ‘, s:5:2);

end.

**Вопрос №3**

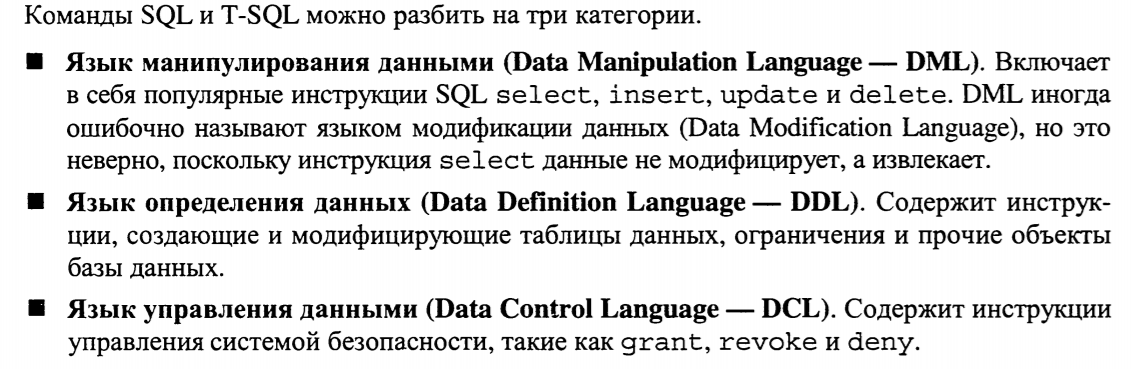
1. Как расшифровывается аббревиатура СУБД?

Система управления базами данных – пакет приклад программ, предназначенных для создания, сопровождения и использования БД.

1. Какие операторы подмножества DML языка SQL вы знаете?

DML (Data Manipulation Language) – язык манипуляции (не манипулирования) данными:

SELECT - отобрать строки из таблиц  
INSERT - добавить строки в таблицу  
UPDATE - изменить строки в таблице  
DELETE - удалить строки в таблице



1. С помощью какого оператора можно изменить значение поля записи в таблице БД?

UPDATE <таблица>

SET <столбец> = <выражение>

[FROM <список\_таблиц>]

[WHERE <условие\_выбора\_строк>]

Пример:

UPDATE D

SET dprice = 0.9 \* dprice

WHERE dnum = 1

1. С помощью какого оператора можно удалить запись из таблицы БД?

DELETE [<таблица>]

FROM <список\_таблиц>

[WHERE <условие\_выбора\_строк>]

1. С помощью какого оператора можно добавить новые записи в таблицу БД?

Вставить строки в указанную таблицу и заполнить конкретными значениями

INSERT INTO <таблица> (<список\_столбцов>)

VALUES (<список\_значений>)

ИЛИ скопировать существующ строки, которые отбирает select

INSERT INTO <таблица> (<список\_столбцов>)

<select оператор>

1. С помощью какой функции можно определить среднее значение поля при выполнении запроса на выборку?

AVG

Пример:

SELECT AVG(dprice) FROM D

1. С помощью какой функции можно определить количество записей в группе при выполнении запроса на выборку?

COUNT

1. С помощью какого оператора можно создать новую таблицу БД?

CREATE TABLE <таблица> (<столбец\_1> <тип>[, ..n])

1. Что такое хранимая процедура?

ХП представляет собой последовательность операторов языка SQL, которые хранятся в БД. 2 типа ХП:

1.Системные: для получения инф из системных таблиц и выполнения служебных операций sp\_\*

2.Пользовательские: создаются разработчиком БД

ХП сохраняются в БД в откомпилированном виде, поэтому скорость их выполнения выше скорости выполнения обычных заепросов

CREATE PROC <имя\_ХП> [(<список\_параметров>)] AS <операторы>

1. Каким оператором выполняется извлечение данных через курсор?

FETCH [NEXT | FIRST | LAST] <имя курсора> INTO <список переменных>

1. Каким оператором создается новое представление?

CREATE VIEW <имя представления>

AS <select оператор>

[WITH CHECK OPTION]

1. Что такое триггер применительно к базам данных?

Триггер – специальный тип ХП, который активизируется при выполнении определенных действий над БД. Каждый триггер привязывается к определенной таблице. С их помощью обеспечивается целостность данных в БД и реализуются бизнес-правила для хранимой в ней информации.

DML-триггеры срабатывают при выполнении команд update, insert, delete

DDL-триггеры срабатывают при выполнении команд create, alter, drop

1. Что такое транзакция?

Транзакцией называют неделимую последовательность действий над базой данных, результаты которой либо полностью принимаются, если все действия выполнены правильно, либо полностью отвергаются, если какое-либо действие выполнено с ошибкой. Транзакции обрабатываются транзакционными системами, в процессе работы которых создаётся история транзакций

1. Каким оператором выполняется фиксация транзакции?

COMMIT TRANSACTION

1. Каким оператором выполняется откат транзакции?

ROLLBACK TRANSACTION

1. Какие бывают уровни изоляции транзакций?

Для этого существуют так называемые уровни изоляции транзакций. В стандарте SQL-92 их четыре:

READ UNCOMMITTED он вообще не изолирует операции чтения других транзакций  
READ COMMITTED имеет две формы. Первая форма применяется в пессимистической модели одновременного конкурентного доступа, а вторая - в оптимистической  
REPEATABLE READ устанавливает разделяемые блокировки на все считываемые данные и удерживает эти блокировки до тех пор, пока транзакция не будет подтверждена или отменена  
SERIALIZABLE не допускает возникновения всех четырех проблем параллельного одновременного конкурентного доступа, перечисленных ранее

Самый высокий уровень изоляции – SERIALIZABLE, самый низкий – READ UNCOMMITTED. Чем выше уровень изоляции, тем меньше проблем совместного выполнения транзакций остается.

1. Как расшифровывается аббревиатура OLTP?

OLTP (Online Transaction Processing) обработка транзакций в реальном времени - способ организации БД, при котором система работает с небольшими по размерам транзакциями, но идущими большим потоком, и при этом клиенту требуется от системы минимальное время отклика.

1. Как расшифровывается аббревиатура OLAP?

OLAP (Online Analytical Processing) интерактивная аналитическая обработка - технология работы с многомерными данными, применяемая чаще всего при составлении разнообразных отчётов.

1. Что такое запрос применительно к БД?

Запрос - это объект, который позволяет пользователю получить нужные данные из одной или нескольких базовых таблиц, и других запросов. В запросе можно указать условия, которым должны удовлетворять данные. Благодаря этому запрос позволяет из большого массива информации, хранимой в БД, извлекать только нужные данные. Для создания запроса используют запрос по образцу (QBE) или инструкции SQL. Можно создавать запросы на выборку, обновление, удаление или на добавление данных. С помощью запросов можно создавать новые таблицы, используя данные одной или нескольких таблиц, которые уже существуют

1. Что такое запрос применительно к SQL?

Запрос - это объект, который позволяет пользователю получить нужные данные из одной или нескольких базовых таблиц, и других запросов. В запросе можно указать условия, которым должны удовлетворять данные. Благодаря этому запрос позволяет из большого массива информации, хранимой в БД, извлекать только нужные данные. Для создания запроса используют запрос по образцу (QBE) или инструкции SQL. Можно создавать запросы на выборку, обновление, удаление или на добавление данных. С помощью запросов можно создавать новые таблицы, используя данные одной или нескольких таблиц, которые уже существуют

1. Что такое первичный ключ?

Потенциальный ключ - это подмножество атрибутов K отношений R такие что:

1.свойство уникальности: в отношении (таблице) R не может быть 2 различных кортежей (строк) с одинаковыми значениями K

2.свойство неизбыточности: никакое подмножество K не обладает свойством уникальности

Первичный ключ – потенциальный ключ, который выбран для уникальной идентификации кортежей.

1. Что такое целостность базы данных?

Целостность БД – свойство, означающее то, что БД содержит полную и непротиворечивую информацию, необходимую и достаточную для корректной работы БД.

1. Что такое группировка применительно к оператору SELECT языка SQL.

SELECT <список столбцов>

FROM <список таблиц>

[GROUP BY <условие группировки строк>]

Может группировать большие множества в несколько небольших подмножеств на основе значений столбцов, упомянутых в этом предложении

1. Чем отличаются предложения WHERE и HAVING в операторе SELECT?

WHERE - это ограничивающее выражение. Оно выполняется до того, как будет получен результат операции.

HAVING - фильтрующее выражение. Оно применяется к результату операции и выполняется уже после того как этот результат будет получен, в отличии от where.

Выражения WHERE используются вместе с операциями SELECT, UPDATE, DELETE, в то время как HAVING только с SELECT и предложением GROUP BY.

1. Что такое подзапрос?

Подзапрос представляет собой оператор, вложенный в тело другого оператора. Внешний оператор использует результаты выполнения внутреннего.

Подзапросы: скалярные и табличные; простые и сложные

**Вопрос №4**

1. Какие принципы лежат в основе объектно-ориентированного программирования?

Абстрагирование - это способ выделить набор значимых характеристик объекта, исключая из рассмотрения незначимые.

Инкапсуляция - это свойство системы, позволяющее объединить данные и методы, работающие с ними, в классе и скрыть детали реализации от пользователя.

Наследование - это свойство системы, позволяющее описать новый класс на основе уже существующего с частично или полностью заимствующейся функциональностью. Класс, от которого производится наследование, называется базовым, родительским или суперклассом. Новый класс - потомком, наследником или производным классом.

Полиморфизм - это свойство системы использовать объекты с одинаковым интерфейсом без информации о типе и внутренней структуре объекта.

1. Что такое класс применительно к объектно-ориентированному программированию?

Класс – производный структурированный тип, введенный программистом на основе уже существующих типов, включающий данные и методы.

1. Как описать новый класс в языке С++?

class <Имя\_класса>

{

public:

конструкторы

деструктор

методы

protected:

private:

поля

}

1. Что такое инкапсуляция?

Инкапсуляция - это свойство системы, позволяющее объединить данные и методы, работающие с ними, в классе и скрыть детали реализации от пользователя

1. Что такое наследование?

Наследование - это свойство системы, позволяющее описать новый класс на основе уже существующего с частично или полностью заимствующейся функциональностью. Класс, от которого производится наследование, называется базовым, родительским или суперклассом. Новый класс - потомком, наследником или производным классом

1. Для чего используются служебные слова class и struct в языке С++?

Структурный тип данных – struct, класс – class. Struct по умолчанию используется открытый доступ, а class - к закрытому доступу

1. Что такое метод класса?

Метод класса – подпрограмма, принадлежащая какому-либо классу, состоящая из операторов для выполнения какого-либо действия и имеющая набор входных данных

1. Что такое конструктор класса в языке С++?

Конструктор – специальный метод, предназначенный для инициализации объекта, в котором обычно присваивают начальные значения полям

1. Что такое стандартный конструктор в языке С++?

Конструктор по умолчанию – конструктор, который может быть вызван без аргументов.

1. Что такое деструктор класса в языке С++?

Деструктор – специальный метод, предназначенный для уничтожения объекта

1. С помощью какой операции можно создать новый экземпляр класса в языке С++?

int main()

{

MyClass \*pC = new MyClass(); // вызов конструктора по умолчанию для инициализации объекта

}

1. С помощью какой операции можно удалить объект в языке С++?

Оператор delete отвечает за освобождение памяти и уничтожение объекта (вызывает деструктор объекта)

1. Что такое перегрузка функции в языке С++?

Под перегрузкой функции понимается, определение нескольких функций (две или больше) с одинаковым именем, но различными параметрами. Наборы параметров перегруженных функций могут отличаться порядком следования, количеством, типом. Таким образом перегрузка функций нужна для того, чтобы избежать дублирования имён функций, выполняющих сходные действия, но с различной программной логикой

1. Что такое перегрузка операции в языке С++?

Перегрузка операции - один из способов реализации полиморфизма, заключающийся в возможности одновременного существования в одной области видимости нескольких различных вариантов применения оператора, имеющих одно и то же имя, но различающихся типами параметров, к которым они применяются

class Complex {

…

Complex operator +(Complex &right) const

{

return Complex(\_re + right.re(), \_im + right.im());

}

}

1. Что такое виртуальная функция в языке С++?

Виртуальная функция — это функция, которая определяется в базовом классе, а любой порожденный класс может ее переопределить. Виртуальная функция вызывается только через указатель или ссылку на базовый класс

1. Что такое абстрактный класс в языке С++?

Абстрактные классы - это классы, которые с одержат или наследуют без переопределения хотя бы одну чистую виртуальную функцию (вирт функцию без определения)

1. С помощью какого стандартного объекта языка С++ выполняется вывод текстового сообщения на экран?

cout << ‘message’; // вывод в поток

printf(‘message’); // в стиле Си

1. С помощью какого стандартного объекта языка С++ выполняется ввод текста с клавиатуры?

cin >> x; // ввод из потока

scanf(x); // в стиле Си

1. Каким служебным словом обозначается шаблон класса в языке С++?

template<typename T>

class MyClass {…}

1. Для чего используется служебное слово friend в языке С++?

Дружественная функция — это функция, которая не является членом класса, но имеет доступ к членам класса, объявленным в полях private или protected

1. С помощью каких операторов реализован механизм исключений в языке С++.

Для реализации обработки исключений в C++ используйте выражения try, throw и catch

1. Что такое множественное наследование в языке С++?

Множественное наследование позволяет порожденному классу наследовать элементы более, чем от одного базового класса

1. Что означают служебные слова public, protected и private применительно к членам класса в языке С++.

Модификаторы доступа

Public — доступ открыт всем, кто видит определение данного класса.

Protected — доступ открыт классам, производным от данного. То есть, производные классы получают свободный доступ к таким свойствам или метода. Все другие классы такого доступа не имеют.

Private — доступ открыт самому классу (т.е. функциям-членам данного класса) и друзьям (friend) данного класса - как функциям, так и классам. Однако производные классы не получают доступа к этим данным совсем. И все другие классы такого доступа не имеют

1. В чем отличие операций . и -> (точка и стрелка) в языке С++?

Доступ к элементам структур и классов может осуществляться двумя операциями: операцией точки (.) или операцией стрелки (->). Если доступ осуществляется через объект, то используется операция точка.

Если доступ осуществляется через указатель на объект, то ->

1. Что такое inline-функция в языке С++?

inline-функция – функция, определенная в заголовочном файле (встроенная функция). inline – совет компилятору, вставлять код функции прямо в место вызова.

**Вопрос №5**

1. Что такое TCP/IP?

TCP/IP — сетевая модель передачи данных, представленных в цифровом виде. Модель описывает способ передачи данных от источника информации к получателю. Название TCP/IP происходит из двух важнейших протоколов семейства — Transmission Control Protocol (TCP) и Internet Protocol (IP), которые первыми были разработаны и описаны в данном стандарте

1. Что такое UDP применительно к сети Интернет?

UDP (англ. User Datagram Protocol — протокол пользовательских датаграмм) — один из ключевых элементов TCP/IP, набора сетевых протоколов для Интернета. С UDP компьютерные приложения могут посылать сообщения (в данном случае называемые датаграммами) другим хостам по IP-сети без необходимости предварительного сообщения для установки специальных каналов передачи или путей данных

1. Что такое HTTP?

HTTP (англ. HyperText Transfer Protocol — «протокол передачи гипертекста») — протокол прикладного уровня передачи данных (изначально — в виде гипертекстовых документов в формате «HTML», в настоящий момент используется для передачи произвольных данных)

1. Что такое POP3?

POP3 (англ. Post Office Protocol Version 3 — протокол почтового отделения, версия 3) — стандартный интернет-протокол прикладного уровня, используемый клиентами электронной почты для получения почты с удалённого сервера по TCP-соединению

1. Что такое SMTP?

SMTP (англ. Simple Mail Transfer Protocol — простой протокол передачи почты) — это широко используемый сетевой протокол, предназначенный для передачи электронной почты в сетях TCP/IP

1. Что такое FTP?

FTP (англ. File Transfer Protocol) — протокол передачи файлов по сети. В отличие от TFTP, гарантирует передачу (либо выдачу ошибки) за счёт применения квитируемого протокола TCP. Стандартный порт управления FTP-соединением — 21. Типичное применение FTP-протокола — загрузка сайтов и других документов с частного устройства разработки на общедоступные сервера хостинга

1. Что такое SSL применительно к сети Интернет?

SSL (англ. Secure Sockets Layer — уровень защищённых cокетов) — криптографический протокол, который подразумевает более безопасную связь. Он использует асимметричную криптографию для аутентификации ключей обмена, симметричное шифрование для сохранения конфиденциальности, коды аутентификации сообщений для целостности сообщений. Протокол широко использовался для обмена мгновенными сообщениями и передачи голоса через IP (англ. Voice over IP — VoIP) в таких приложениях, как электронная почта, интернет-факс и др

1. Что такое SSH применительно к сети Интернет?

SSH (англ. Secure Shell — «безопасная оболочка»[1]) — сетевой протокол прикладного уровня, позволяющий производить удалённое управление операционной системой и туннелирование TCP-соединений (например, для передачи файлов). Схож по функциональности с протоколами Telnet и rlogin, но, в отличие от них, шифрует весь трафик, включая и передаваемые пароли

1. Что такое Telnet?

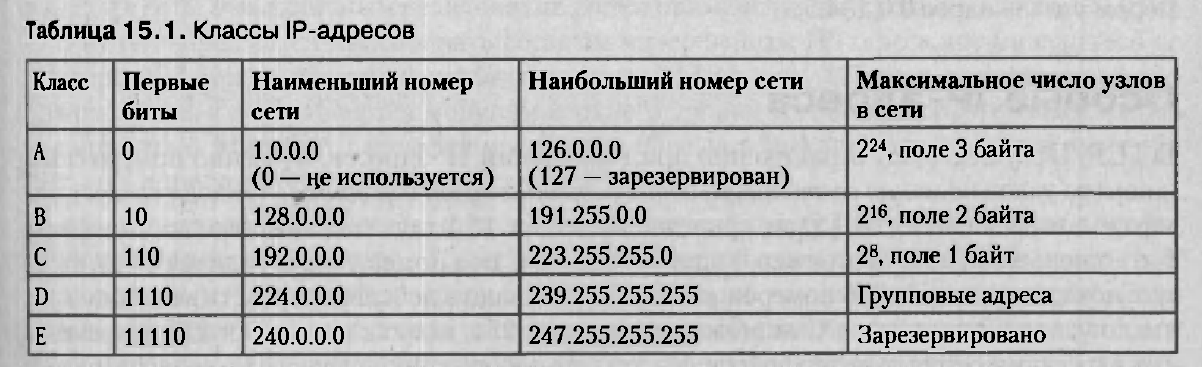
TELNET (сокр. от англ. teletype network) — сетевой протокол для реализации текстового интерфейса по сети (в современной форме — при помощи транспорта TCP). Название «telnet» имеют также некоторые утилиты, реализующие клиентскую часть протокола

1. Что такое ISP применительно к сети Интернет

Интернет-прова́йдер (иногда просто провайдер; от англ. internet service provider, сокр. ISP — поставщик интернет-услуги) — организация, предоставляющая услуги доступа к сети Интернет и иные связанные с Интернетом услуги

1. Что такое IP-адрес?

IP-адрес (ˌaɪ ˈpiː-адрес; является словообразовательной полукалькой английского IP address, буквенной аббревиатурой от англ. Internet Protocol Address «адрес Интернет-протокола») — уникальный сетевой адрес узла в компьютерной сети, построенной на основе стека протоколов TCP/IP



1. Приведите пример IP-адреса?

192.168.0.0

1. Что такое доменное имя применительно к сети Интернет?

Доменное имя — символьное имя, служащее для идентификации областей — единиц административной автономии в сети Интернет — в составе вышестоящей по иерархии такой области. Каждая из таких областей называется доменом. Общее пространство имён Интернета функционирует благодаря DNS — системе доменных имён

1. Что такое хост применительно к сети Интернет?

Хост (от англ. host — «хозяин, принимающий гостей») — любое устройство, предоставляющее сервисы формата «клиент-сервер» в режиме сервера по каким-либо интерфейсам и уникально определённое на этих интерфейсах. В более частном случае под хостом могут понимать любой компьютер, сервер, подключённый к локальной или глобальной сети

1. Что такое TTL применительно к протоколу IP?

TTL — Time to live — время жизни пакета данных в протоколе IP (предельно допустимое время его пребывания в системе)

1. Что такое порт применительно к протоколу TCP?

Порт источника идентифицирует приложение клиента, с которого отправлены пакеты. Ответные данные передаются клиенту на основании этого номера.

Порт назначения идентифицирует порт, на который отправлен пакет.

1. Что такое proxy-сервер?

Прокси-сервер (от англ. proxy — «представитель, уполномоченный») — промежуточный сервер (комплекс программ) в компьютерных сетях, выполняющий роль посредника между пользователем и целевым сервером (при этом о посредничестве могут как знать, так и не знать обе стороны), позволяющий клиентам как выполнять косвенные запросы (принимая и передавая их через прокси-сервер) к другим сетевым службам, так и получать ответы

1. Что такое DNS-сервер?

DNS-сервер, Domain name system — приложение, предназначенное для ответов на DNS-запросы по соответствующему протоколу. Также DNS-сервером могут называть хост, на котором запущено приложение.

1. Что такое HTML?

HTML (от англ. HyperText Markup Language — «язык гипертекстовой разметки») — стандартизированный язык разметки документов во Всемирной паутине. Большинство веб-страниц содержат описание разметки на языке HTML (или XHTML). Язык HTML интерпретируется браузерами; полученный в результате интерпретации форматированный текст отображается на экране монитора компьютера или мобильного устройства

1. Что такое CSS?

CSS (Cascading Style Sheets — каскадные таблицы стилей) — формальный язык описания внешнего вида документа, написанного с использованием языка разметки

1. Что такое Javascript?

JavaScript— мультипарадигменный язык программирования. Поддерживает объектно-ориентированный, императивный и функциональный стили. Является реализацией языка ECMAScript

1. Что такое PHP?

PHP (Hypertext Preprocessor —препроцессор гипертекста; первоначально Personal Home Page Tools — «Инструменты для создания персональных веб-страниц») — скриптовый язык общего назначения, интенсивно применяемый для разработки веб-приложений. В настоящее время поддерживается подавляющим большинством хостинг-провайдеров и является одним из лидеров среди языков, применяющихся для создания динамических веб-сайтов

1. Что такое XML?

XML (eXtensible Markup Language) — расширяемый язык разметки. Рекомендован Консорциумом Всемирной паутины (W3C). Спецификация XML описывает XML-документы и частично описывает поведение XML-процессоров (программ, читающих XML-документы и обеспечивающих доступ к их содержимому). XML разрабатывался как язык с простым формальным синтаксисом, удобный для создания и обработки документов программами и одновременно удобный для чтения и создания документов человеком, с подчёркиванием нацеленности на использование в Интернете

1. Что такое Apache применительно веб-технологиям?

Apache HTTP (HyperText Transfer Protocol) Server – свободный веб-сервер, разработанный Apache Software Foundation

Веб-сервер - сервер, принимающий HTTP-запросы от клиентов, обычно веб-браузеров, и выдающий им HTTP-ответы, как правило, вместе с HTML-страницей,

1. Что такое хостинг-провайдер?

Хостинг-провайдер — это компания, оказывающая услуги по предоставлению дискового пространства для размещения файлов на своем сервере. Такая фирма предоставляет место для размещения сайта. На сервере хостинг-провайдера стоит ПО, необходимое для обработки запросов к файлам сайта, которые посылают пользователи интернета

**Вопрос №6**

1. Язык SQL. Предложение SELECT оператора SELECT.

В своей базовой форме инструкция SELECT сообщает серверу, какие данные следует извлечь, в частности, какие столбцы и строки из каких таблиц получить и как сортировать данные.

Ниже приведен упрощенный синтаксис инструкции SELECT.

SELECT <столбцы или выражения>

[FROM <список таблиц>]

[JOIN <таблица>

ON <условия>]

[WHERE <условия выбора строк>]

[GROUP BY <условия группировки строк>]

[HAVING <условия выбора групп>]

[ORDER BY <условия сортировки>]

Инструкция SELECT начинается со списка столбцов или выражений. Как минимум обязательно наличие хотя бы одного выражения, все остальное - необязательно. Простейшая из возможных инструкций SELECT выглядит следующим образом: SELECT 1;

1. Язык SQL. Предложение WHERE оператора SELECT.

Предложение WHERE фильтрует строки набора данных, собранного предложением FROM, на основе некоторых условий.

1. Язык SQL. Предложение GROUP BY оператора SELECT.

Агрегатные функции выполняют в наборе данных итоговые подсчеты. Предложение GROUP BY может группировать большие множества в несколько небольших подмножеств на основе значений столбцов, упомянутых в этом предложении.

1. Язык SQL. Предложение HAVING оператора SELECT.

Затем агрегатные функции применяются к этим небольшим подмножествам данных, после чего результаты агрегатных функций фильтруются с помощью предложения HAVING

1. Язык SQL. Конструкция UNION оператора SELECT.

Число и порядок извлекаемых столбцов должны совпадать во всех объединяемых запросах;

Типы данных в соответствующих столбцах должны быть совместимы.

UNION применяется для объединения результатов двух SQL-запросов в единую таблицу, состоящую из схожих строк.

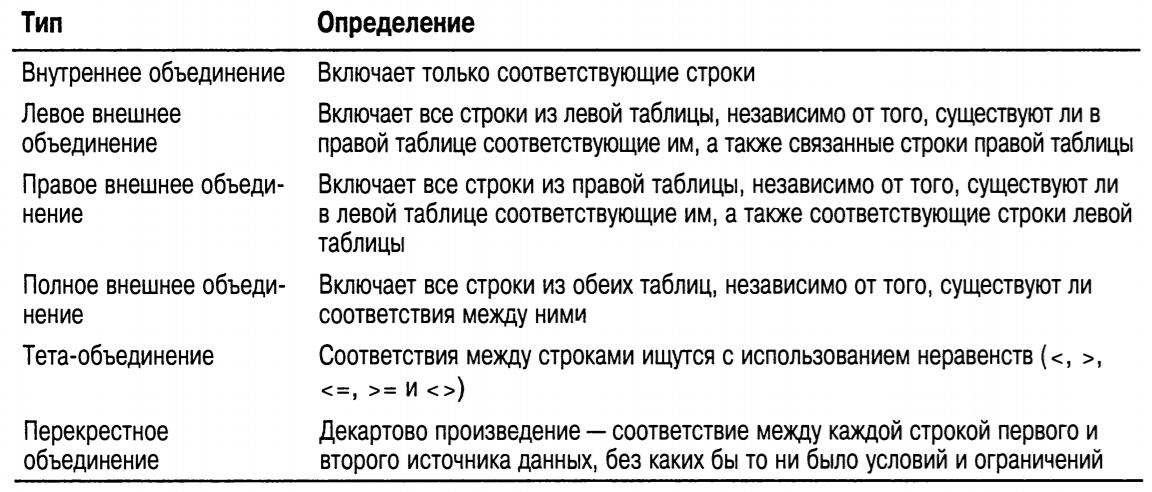
1. Язык SQL. Предложение ORDER BY оператора SELECT.

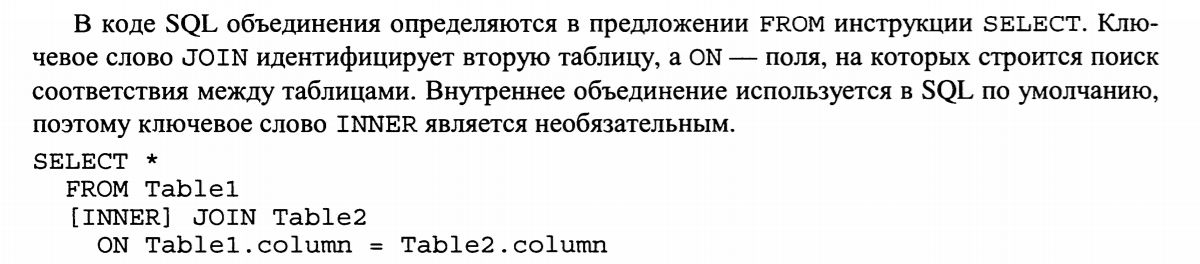
предложение ORDER BY определяет порядок сортировки результирующего набора данных

1. Язык SQL. Предложение FROM оператора SELECT. Декартово произведение и внутреннее соединение.

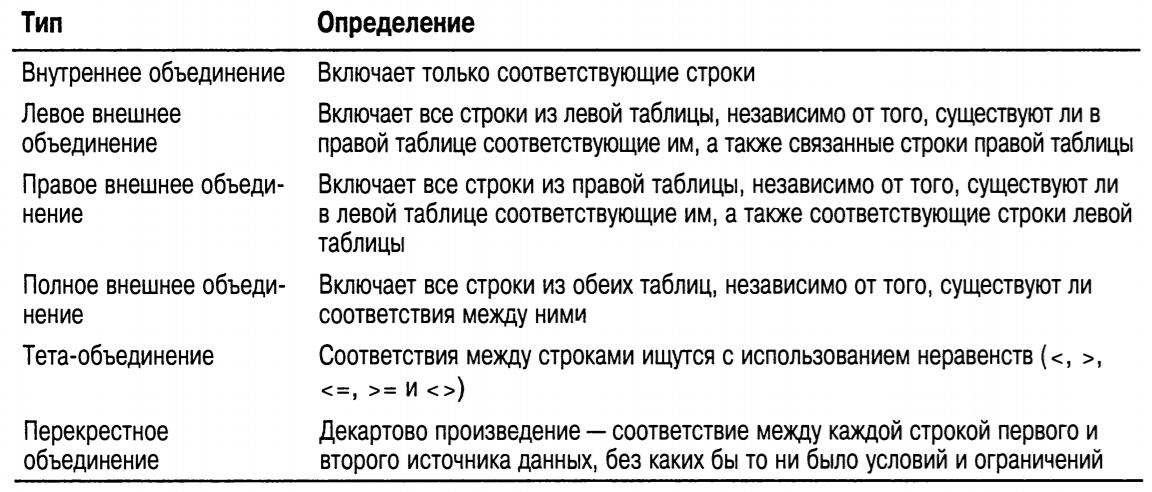
Предложение FROM B инструкции SELECT собирает все источники данных в единый набор, над которым будет работать остальная часть инструкции. В предложении FROM может участвовать множество таблиц, которые ссылаются друг на друга с помощью одного из нескольких типов объединений.

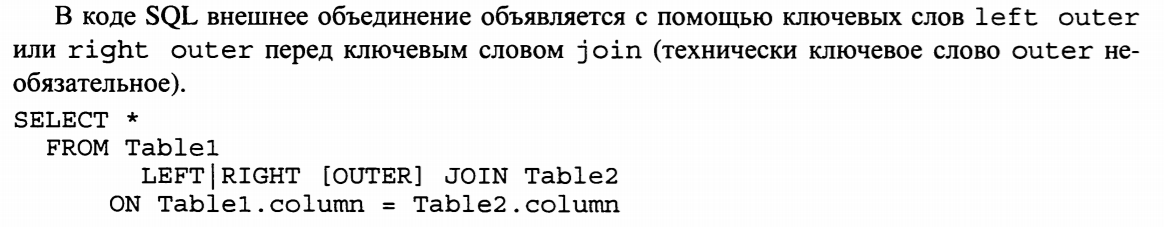
Декартово произведение отношений (таблиц) A и B – отношение (таблица), заголовок которого является сцеплением заголовков A и B, а тело представляет собой результат сцепления каждого кортежа (строки) из отношения А с каждым кортежем из отношения B.





1. Язык SQL. Предложение FROM оператора SELECT. Левое внешнее и правое внешнее соединения.





1. Язык SQL. Использование подзапросов.

Подзапрос представляет собой оператор, вложенный в тело другого оператора. Внешний оператор использует результаты выполнения внутреннего.

Подзапрос – законченный оператор SELECT, внедрённый в тело другого оператора SELECT. Внешний (второй) оператор SELECT использует результат выполнения внутреннего (первого) оператора для определения содержания окончательного результата всей операции. Внутренние запросы могут находиться в конструкциях WHERE и HAVING внешнего оператора SELECT — в этом случае они получают название подзапросов, или вложенных запросов. Кроме того, внутренние операторы SELECT могут использоваться в операторах INSERT, UPDATE и DELETE

Подзапросы: скалярные и табличные; простые и сложные

1. Язык SQL. Оператор INSERT.

Оператор INSERT вставляет новые записи в таблицу. При этом значения столбцов могут представлять собой литеральные константы, либо являться результатом выполнения подзапроса. В первом случае для вставки каждой строки используется отдельный оператор INSERT; во втором случае будет вставлено столько строк, сколько возвращается подзапросом.

Вставить строки в указанную таблицу и заполнить конкретными значениями

INSERT INTO <таблица> (<список\_столбцов>)

VALUES (<список\_значений>)

ИЛИ скопировать существующ строки, которые отбирает select

INSERT INTO <таблица> (<список\_столбцов>)

<select оператор>

1. Язык SQL. Оператор UPDATE.

UPDATE <таблица>

SET <столбец> = <выражение>

[FROM <список\_таблиц>]

[WHERE <условие\_выбора\_строк>]

Пример:

UPDATE D

SET dprice = 0.9 \* dprice

WHERE dnum = 1

1. Язык SQL .Оператор DELETE.

DELETE [<таблица>]

FROM <список\_таблиц>

[WHERE <условие\_выбора\_строк>]

1. Язык SQL. Создание, удаление и изменение структуры таблиц.

CREATE TABLE <таблица> (<столбец\_1> <тип>[, ..n])

ALTER TABLE <таблица> ADD <имя столбца> <тип> [<ограничение>]

ALTER TABLE <таблица> ADD [CONSRTAINT <имя ограничения>] <ограничение>

ALTER TABLE <таблица> ALTER COLUMN

ALTER TABLE <таблица> DROP COLUMN

ALTER TABLE <таблица> DROP CONSTRAINT

удаление

DROP TABLE <таблица>

1. Язык SQL. Создание ограничения первичного ключа.

Таблица может иметь только одно ограничение PRIMARY KEY, причем ни один из включенных в первичный ключ столбцов не может принимать значение NULL. При попытке использовать в качестве первичного ключа столбец (или группу столбцов), для которого ограничения первичного ключа не выполняются, первичный ключ создан не будет, а система выдаст сообщение об ошибке

Create table D (nu mint PRIMARY KEY)

или

alter table D add constraint PK\_D PRIMARY KEY (num)

1. Язык SQL. Создание ограничения внешнего ключа.

alter table PD add constraint FK\_PD\_P FOREIGN KEY (pnum) REFERENCES P(pnum) [поддержание ссылочной целостности]

ON DELETE, ON UPDATE

1.NO ACTION стратегия ограничения

2.CASCADE стратегия каскадирования

3.SET NULL установка в нулл

4.SET DEFAULT установка по умолчанию

1. Язык SQL. Создание ограничений умолчания, проверки и уникальности.

CREATE TABLE D (

c1 int [NOT NULL | NULL]

c2 int UNIQUE // уникальность

price money CHECK (price < 50) // проверка

price money2 DEFAULT 10 ) // умолчание

пользовательское умолчание

CREATE DEFAULT <имя умолчания> AS <умолчание>

CREATE DEFAULT D1 AS 10

EXEC sp\_bindefault ‘D1’, ‘D.price’

1. Метод ER-диаграмм: основные понятия.

Суть метода состоит в построении ER-диаграммы, отображающей в графической форме основные объекты предметной области и связи между ними и в определении характеристик этих связей. Затем по определенным правилам делается переход от ER-диаграммы к таблицам БД. Результатом проектирования является схема БД.

Определения:

Сущность — это объект, информация о котором должна быть представлена в БД.

Экземпляр сущности — информация о конкретном представителе объекта.

Связь — соединение между двумя и более сущностями.

Экземпляр связи — информация о конкретной связи между конкретными представителями объектов.

Атрибут — представляет собой свойство сущности.

Ключ сущности — множество атрибутов для однозначной идентификации сущности.

1. Избыточное дублирование и аномалии в отношениях БД.

Схема базы данных — это структура связей между полями и таблицами, получаемая в результате проектирования базы данных. Основная проблема, возникающая при проектировании — избыточность хранимых в ней данных, которую необходимо минимизировать во избежание связанных с ней сложностей.

При работе с отношениями, содержащими избыточные данные, возникают проблемы, которые называют аномалиями модификации. Причины таких аномалий – избыточность данных, порождённая тем, что в отношении находится разнородная информация. Различают следующие виды аномалий:

аномалия вставки; аномалия удаления; аномалия обновления.



В рассматриваемом примере аномалии модификации проявляется следующим образом:

1) Аномалия вставки. В отношение нельзя вставить информацию о студенте, который находится в академическом отпуске, или информацию о преподавателе, который в текущем семестре не принимает ни одного экзамена.

2) Аномалия удаления. Пусть при изменении учебного плана будет исключён некоторый предмет. Если за преподавателем не закреплён больше ни один предмет, то данные о преподавателе теряются.

3) Аномалия обновления. Если изменится номер группы студента или должность преподавателя, то такое изменение необходимо выполнить во всех кортежах, где записана эта информация. Устранить аномалии можно, выполнив нормализацию отношений

1. Зависимости между атрибутами в отношениях БД.

Определение **функциональной** зависимости.

Пусть К — отношение, а X,Y — произвольные атрибуты (множества атрибутов) отношения R. Y функционально зависит от X (X функционально определяет Y, X → Y) если одному значению X соответствует ровно одно значение Y. Обратное не обязательно. X называют детерминантом функциональной зависимости, а Y – зависимой частью.

В приведённом примере можно выделить такие зависимости: [ФИО студента] → [Номер группы], {[ФИО студента], Предмет} → Оценка.

Определение. Функциональная зависимость X → Y называется полной, если Y не зависит от любого собственного подмножетства X. Если детерминант функциональной зависимости представляет собой один атрибут, то ФЗ автоматически является полной.

Определение **транзитивной функциональной** зависимости.

Функциональная зависимость X → Y называется транзитивной, если есть такой атрибут Z, что существуют функциональные зависимости X → Z, Z → Y, и не существует функциональной зависимости Z → X. Тогда говорят, что Y транзитивно зависит от X, или что X транзитивно определяет Y.

Определение. **Неключевым атрибутом** называется любой атрибут отношения, который не входит в состав потенциальных ключей.

Определение. Атрибуты называются **взаимонезависимыми**, если ни один из них не является функционально зависимым от другого.

1. Метод нормальных форм. 1НФ.

Нормализация – формальный метод анализа отношений на основе первичных или потенциальных ключей и существующих зависимостей между атрибутами. Процесс нормализации заключается в последовательном переводе отношений из первой нормальной формы 1НФ в нормальные формы более высокого порядка по определённым правилам. Каждой нормальной форме соответствует определённый набор ограничений, и отношения находятся в некоторой нормальной форме, если удовлетворяют свойственному ей набору ограничений.

Отношение находится в 1НФ, если в нём на пересечении каждой строки и столбца находится ровно одно значение

1. Метод нормальных форм. 2НФ.

Отношение находится в 2НФ, если оно находится в 1НФ и каждый неключевой атрибут функционально полно зависит от любого потенциального ключа, в частности, от первичного.

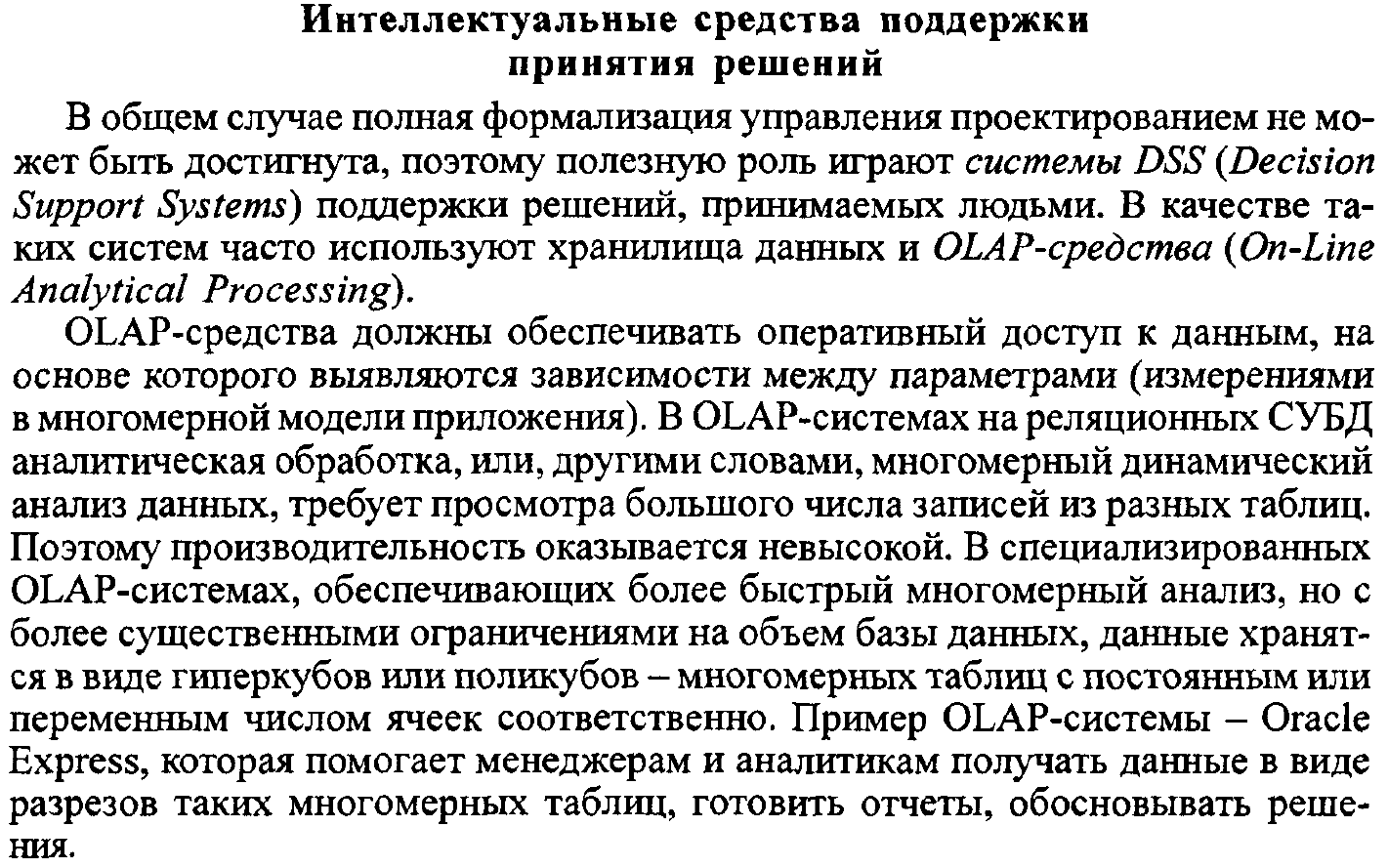
1. Метод нормальных форм. 3НФ.

Отношение находится в 3НФ, если оно находится в 2НФ и если в нём нет транзитивных зависимостей неключевых атрибутов от любого потенциального ключа (в частности, первичного)

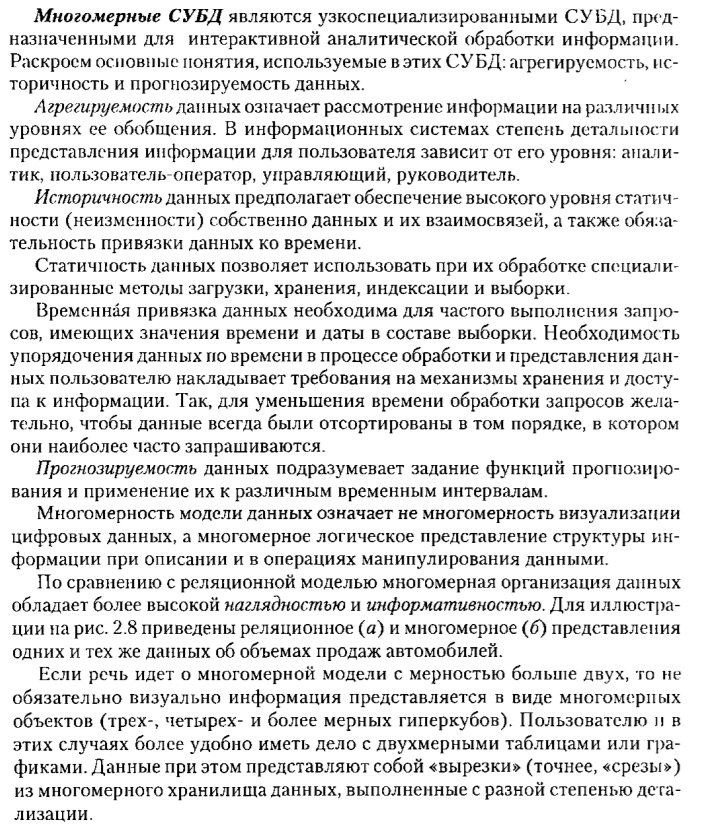
1. Метод нормальных форм. Нормальная форма Бойса-Кодда.

Отношение находится в БКНФ, если оно находится в 3НФ и если детерминанты всех функциональных зависимостей являются потенциальными ключами. БКНФ часто называют усиленной 3НФ

1. Особенности проектирования БД для OLTP и OLAP информационных систем.



|  |  |  |
| --- | --- | --- |
| Характеристика | OLTP | OLAP |
| Преобладающие операции | Ввод данных, поиск | Анализ данных |
| Характер запросов | Много простых транзакций | Сложные транзакции |
| Хранимые данные | Оперативные, детализированные | Охватывающие большой период времени, агреги­рованные |
| Вид деятельности | Оперативная, тактическая | Аналитическая, страте­гическая |
| Тип данных | Структурированные | Разнотипные |



1. Понятие транзакции.

При работе с БД нередко возникает потребность согласовать выполнение нескольких логически связанных между собой действий. Для этого применяют механизм транзакций. Транзакцией называют неделимую последовательность действий над базой данных, результаты которой либо полностью принимаются, если все действия выполнены правильно, либо полностью отвергаются, если какое-либо действие выполнено с ошибкой.

Транзакции в основном применяют для поддержания целостности данных в корпоративных БД. Большинство современных серверных СУБД, в том числе и MS SQL Server, поддерживают механизм транзакций.

При работе с транзакциями используют следующие основные действия.

1. Начать транзакцию. Все последующие действия над БД до завершения транзакции войдут в состав транзакции.

2. Фиксация транзакции. Транзакция завершается, а результаты всех действий над БД в составе транзакции навсегда сохраняются в БД. Фиксацию применяют в том случае, когда все действия в составе транзакции выполнены правильно.

3. Откат транзакции. Результаты всех действий, которые успела совершить транзакция, отменяются, и БД возвращается в то состояние, в котором она была до начала выполнения транзакции. Откат применяют в том случае, когда какое-либо действие в составе транзакции выполнено с ошибкой. Например, не удалось добавить запись в таблицу БД или не найдена искомая запись в таблице. Если во время работы с БД произойдет аппаратный или программный сбой, то после восстановления работоспособности сервера произойдет автоматический откат всех ранее начатых, но незавершенных транзакций.

Язык Transact-SQL содержит следующие основные операторы для работы с транзакциями.

1. BEGIN TRAN – начинает новую явную транзакцию.

2. COMMIT TRAN – выполняет фиксацию транзакции.

3. ROLLBACK TRAN – выполняет откат транзакции

**Вопрос №7**

1. Архитектура ЭВМ.

Электронная вычислительная машина - физическая система, служащая для автоматизации процесса вычислений и алгоритмической обработки информации путем выполнения программы, хранимой в памяти машины.

Архитектура определяется обычно совокупностью свойств компьютера, существенных для пользователя: логическое построение (перечень и формат команд, формы представления данных, механизмы ввода/вывода, способы адресации памяти …). Архитектура не затрагивает вопросы физического построения (число регистров, емкость памяти, тактовая частота цп…)

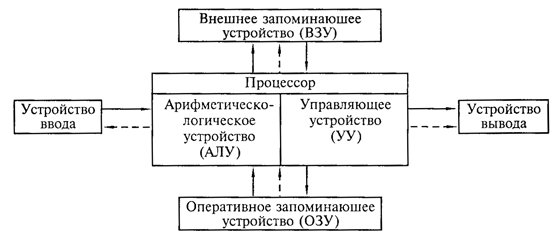


Рис. Архитектура фон Неймана

Большинство ЭВМ построены на основе некоторых общих принципов‚ заложенных американским учёным Джоном фон Нейманом, основными из которых являются:

1.двоичное кодирование информации и деление её на слова фиксированной длины;

2.принцип произвольного доступа к основной памяти (ОП).

Структурно ОП имеет линейно-адресную организацию, т.е. состоит из последовательно расположенных друг за другом ячеек (байтов) - адресного пространства. Это максимальное количество ячеек основной памяти, которое может быть адресовано непосредственно процессором.

3.представление алгоритма программой, состоящей из команд;

4.команды и данные хранятся в одной памяти. При этом адресное пространство памяти делится на две области: область программ и область данных;

5.вычислительный процесс определяется как последовательное выполнение команд в порядке, определяемом программой;

6.жёсткость структуры, т.е. неизменность ее в процессе работы ЭВМ;

7.принцип программного управления: решение любой задачи выполняется автоматически по программе, хранимой в памяти.

8.принцип хранимой программы: программа хранится в памяти наравне с данными, а, следовательно, над командами можно выполнять всевозможные операции, как над числами, что ведет к повышению быстродействия. Для решения любой задачи на ЭВМ в неё надо ввести данные и программу.

1. Арифметические основы ЭВМ (системы счисления, перевод из одной системы в другую, арифметические операции, коды прямой, дополнительный и обратный).

Система счисления – совокупность цифр (символов) и правил для обозначения чисел. Бывают непозиционные и позиционные.

Непозиционная система счисления - система, в которой символы, используемые для представления числа, не меняют своего значения в зависимости от местоположения (позиции) в записи числа (римская). Позиционная наоборот – наша десятичная.

Упорядоченный набор символов (цифр): а0, а1‚ ...‚аn‚ используемых для представления любых чисел в заданной позиционной системе счисления, называют ее алфавитом, число символов (цифр алфавита) - ее основанием (q), a саму систему счисления называют q-ичной.

Основание системы счисления q показывает, во сколько раз увеличивается значение цифры в числе при смещении ее на одну позицию влево. Внутри компьютера информация представляется в виде чисел, записанных в той или иной системе счисления (СС), кратной степени двойки (двоичной, 16-ричной и др.). Такие операции, как ввод, вывод и обработка чисел на компьютере связаны с их преобразованием из одной СС в другую. Различают перевод целых, дробных и смешанных чисел.

Перевод целых чисел: Перевод из десятичной в q-ичную систему счисления:

Перевод можно выполнить различными способами - либо делением на основание системы счисления, либо методом "поразрядного взвешивания" или методом взвешенного кодирования (используется для перевода из десятичной системы счисления в двоичную). При переводе по первому способу пользуются следующим правилом: исходное число Х и последовательно получающиеся частные делят на основание новой системы счисления q до получения частного, меньшего q. Остатки, полученные при делении, являются разрядами числа в новой q-ичной системе счисления. Последний остаток, за который принимается последнее частное, является старшим разрядом числа, т.е. для записи числа Xq остатки записываются в порядке, обратном их получению.

Перевод дробных чисел: Перевод из десятичной в q-ичную систему счисления:

При переводе дробных чисел говорят о переводе с заданной точностью и используют метод последовательного умножения на основание новой системы счисления q: исходное число Х и получающиеся дробные части последовательно умножают на q до получения дробной части, равной нулю (при точном переводе), или до получения нужного количества цифр q-ичной записи (при переводе с заданной точностью). Число Х в q-ичной системе образуется как последовательность целых частей произведений.

Для представления чисел со знаком используются машинные коды.

Прямой код n-разрядного двоичного числа X = ±0 Xn-2 Xn-3 … X1 X0:

Используется для представления мантиссы числа, операндов при делении и умножении.

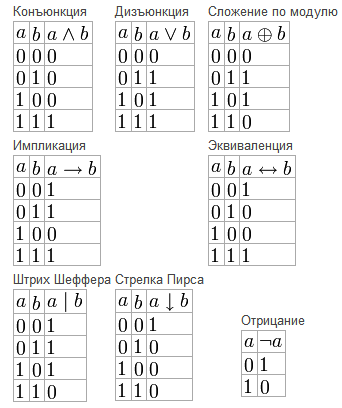
Обратный код практически не используется:

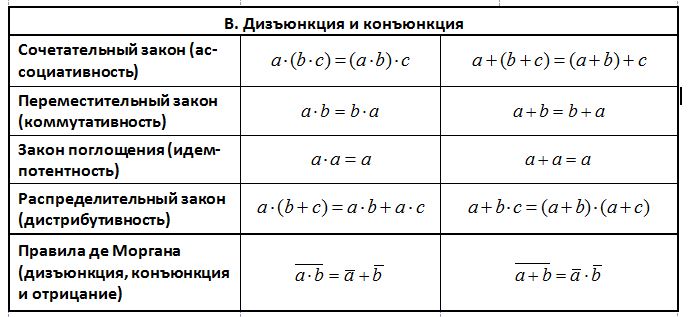
Дополнительный код: Длина числа может быть увеличена путем тиражирования знакового разряда слева.

При нем знак и значащие двоичные цифры при сложении (вычитании) обрабатываются одинаково, поэтому он используется для хранения и выполнения арифм операций над числами.

1. Алгебра логики. Логические функции. Преобразование логических выражений.

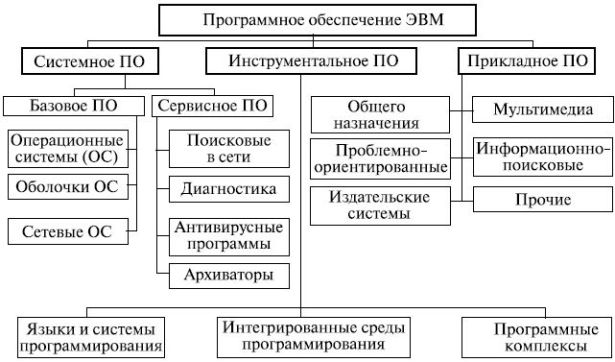
Логическими высказываниями являются утвердительные предложения, о которых можно судить, истинны они или ложны. Причем они не могут быть истинными и ложными одновременно.





1. Программное обеспечение ЭВМ (прикладное и системное).





Программное обеспечение, можно условно разделить на три категории:

1. системное ПО (программы общего пользования), выполняющие различные вспомогательные функции, например создание копий используемой информации, выдачу справочной информации о компьютере, проверку работоспособности устройств компьютера и т.д.
2. прикладное ПО, обеспечивающее выполнение необходимых работ на ПК: редактирование текстовых документов, создание рисунков или картинок, обработка информационных массивов и т.д.
3. инструментальное ПО (системы программирования), обеспечивающее разработку  новых программ для компьютера на языке программирования.

#### **Системное ПО**

Это программы общего пользования не связаны с конкретным применением ПК и выполняют традиционные функции: планирование и управление задачами, управления вводом-выводом и т.д.

Другими словами, системные программы выполняют различные вспомогательные функции, например, создание копий используемой информации, выдачу справочной информации о компьютере, проверку работоспособности устройств компьютера и т.п.

К системному ПО относятся:

* операционные системы (эта программа загружается в ОЗУ при включении компьютера);
* программы – оболочки (обеспечивают более удобный и наглядный способ общения с компьютером, чем с помощью командной строки DOS, например, Norton Commander);
* операционные оболочки – интерфейсные системы, которые используются для создания графических интерфейсов, мультипрограммирования и.т.;
* Драйверы (программы, предназначенные для управления портами периферийных устройств, обычно загружаются в оперативную память при запуске компьютера);
* утилиты (вспомогательные или служебные программы, которые представляют пользователю ряд дополнительных услуг).

К утилитам относятся:

* диспетчеры файлов или файловые менеджеры;
* средства динамического сжатия данных (позволяют увеличить количество информации на диске за счет ее динамического сжатия);
* средства просмотра и воспроизведения;
* средства диагностики; средства контроля позволяют проверить конфигурацию компьютера и проверить работоспособность устройств компьютера, прежде всего жестких дисков;
* средства коммуникаций (коммуникационные программы) предназначены для организации обмена информацией между компьютерами;
* средства обеспечения компьютерной безопасности (резервное копирование, антивирусное ПО).

Необходимо отметить, что часть утилит входит в состав операционной системы, а другая часть функционирует автономно. Большая часть общего (системного) ПО входит в состав ОС. Часть общего ПО входит в состав самого компьютера (часть программ ОС и контролирующих тестов записана в ПЗУ или ППЗУ, установленных на системной плате). Часть общего ПО относится к автономными программам и поставляется отдельно.

#### **Прикладное ПО**

Прикладные программы могут использоваться  автономно или в составе программных комплексов или пакетов.

Прикладное ПО – программы,  непосредственно обеспечивающие выполнение необходимых работ на ПК: редактирование текстовых документов, создание рисунков или картинок, создание электронных таблиц и т.д.

Пакеты прикладных программ – это система программ, которые по сфере применения делятся на проблемно – ориентированные, пакеты общего назначения и интегрированные пакеты. Современные интегрированные пакеты содержат до пяти функциональных компонентов: тестовый и табличный процессор, СУБД, графический редактор, телекоммуникационные средства.

К прикладному ПО, например, относятся:

1. Комплект офисных приложений MS OFFICE.
2. Бухгалтерские системы.
3. Финансовые аналитические системы.
4. Интегрированные пакеты делопроизводства.
5. CAD – системы (системы автоматизированного проектирования).
6. Редакторы HTML или Web – редакторы.
7. Браузеры – средства просмотра Web - страниц.
8. Графические редакторы.
9. Экспертные системы.

И так далее.

#### **Инструментальное ПО**

Инструментальное ПО или системы программирования  - это системы для автоматизации разработки новых программ на языке программирования.

В самом общем случае для создания программы на выбранном языке программирования (языке системного программирования) нужно иметь следующие компоненты:

1. Текстовый редактор для создания файла с исходным текстом программы.
2. Компилятор или интерпретатор. Исходный текст с помощью программы-компилятора переводится в промежуточный объектный код. Исходный текст большой программы состоит из нескольких модулей(файлов с исходными текстами). Каждый модуль компилируется в отдельный файл с объектным кодом, которые затем надо объединить в одно целое.
3. Редактор связей или сборщик, который выполняет связывание объектных модулей и формирует на выходе работоспособное приложение – исполнимый код. Исполнимый код – это законченная программа, которую можно запустить на любом компьютере, где установлена операционная система, для которой эта программа создавалась. Как правило, итоговый файл имеет расширение .ЕХЕ или .СОМ.
4. В последнее время получили распространение визуальный методы программирования (с помощью языков описания сценариев), ориентированные на создание Windows-приложений. Этот процесс автоматизирован в средах быстрого проектирования. При этом используются готовые визуальные компоненты, которые настраиваются с помощью специальных редакторов.

Наиболее популярные редакторы (системы программирования программ с использованием визуальных средств) визуального проектирования:

1. Borland Delphi - предназначен для решения практически любых задачи прикладного программирования.
2. Borland C++ Builder – это отличное средство для разработки DOS и Windows приложений.
3. Microsoft Visual Basic – это популярный инструмент для создания Windows-программ.

Microsoft Visual C++ - это средство позволяет разрабатывать любые приложения, выполняющиеся в среде ОС типа Microsoft Windows.

1. Операционные системы, их функции, классификация, типы.

Программы, составляющие ПО, можно разделить на три группы: системное ПО, системы программирования, прикладное ПО. Ядром системного ПО является операционная система (ОС).

ОС - это неотъемлемая часть ПО, управляющая техническими средствами компьютера (hardware).. Операционная система - это программа, координирующая действия вычислительной машины; под ее управлением осуществляется выполнение программ.

Основные функции операционной системы:

* 1.      Обмен данными между компьютером и различными периферийными устройствами (терминалами, принтерами, гибкими дисками, жесткими дисками и т.д.). Такой обмен данными называется "ввод/вывод данных".
* 2.      Обеспечение системы организации и хранения файлов.
* 3.      Загрузка программ в память и обеспечение их выполнения.
* 4.      Организация диалога с пользователем.

По количеству одновременно работающих пользователей:

• Однопользовательские ОС позволяют работать на компьютере только одному человеку.

• Многопользовательские ОС поддерживают одновременную работу на ЭМВ нескольких пользователей за различными терминалами.

По числу процессов, одновременно выполняемых под управлением системы:

• Однозадачные ОС поддерживают выполнение только одной программы в отдельный момент времени, то есть позволяют запустить одну программу в основном режиме.

• Многозадачные ОС (мультизадачные) поддерживают параллельное выполнение нескольких программ, существующих в рамках одной вычислительной системы на некотором отрезке времени, то есть позволяют запустить одновременно несколько программ, которые будут работать параллельно, не мешая друг другу.

Операционные системы классифицируются по:

* количеству одновременно работающих пользователей: однопользовательские, многопользовательские;
* числу процессов, одновременно выполняемых под управлением системы;
* количеству решаемых задач: однозадачные, многозадачные;
* количеству поддерживаемых процессоров: однопроцессорные, многопроцессорные;
* разрядности кода ОС: 8-разрядные, 16-разрядные, 32-разрядные, 64-разрядные;
* типу интерфейса: командные (текстовые) и объектно-ориентированные (графические);
* типу доступа пользователя к ЭВМ: с пакетной обработкой, с разделением времени, реального времени;

типу использования ресурсов: сетевые, локальные.

1. Трансляторы, компиляторы и интерпретаторы.

Важнейшее место среди системных обрабатывающих программ занимают трансляторы или языковые процессоры. В общем случае транслятором называется программа, которая осуществляет перевод текста с одного языка на другой. Существуют несколько различных видов трансляторов:

1.Компиляторы - переводит исходную программу, представленную на языке программирования высокого уровня, в эквивалентную объектную программу, представленную на языке машинных команд. Реальная программа обычно состоит из нескольких частей, которые компилируются отдельно и независимо друг от друга. В этом случае результат компиляции обычно называется объектным модулем.

2.Интерпретаторы - это программа, которая считывает исходную программу небольшими фрагментами (построчно или отдельными операторами), анализирует их и после распознавания конструкций исходного языка сразу же исполняет их. Объектный код при этом не создается.

С помощью компиляторов и интерпретаторов реализуются языки программирования высокого уровня. Машинно-ориентированные языки (языки ассемблера) реализуются с помощью ассемблеров.

3.Ассемблер представляет собой особый вид компилятора и преобразует исходную программу, представленную на машинно-ориентированном языке (языке ассемблера), в объектный код ЭВМ по принципу “один в один”, т.е. каждый оператор языка ассемблера заменяется одной машинной командой.

4.Макропроцессор представляет собой программу для символьной обработки, которая короткие фразы исходного текста (макровызовы) заменяет соответствующими длинными фразами в результирующем тексте.

5.Дизассемблеры - вспомогательная программа, которая выполняет перевод программы в машинных кодах обратно на язык ассемблера. Обычно используется при отладке и модификации готовых программ в машинных кодах

6.Языковые конверторы являются относительно новым видом трансляторов и используются для перевода текста программы с одного языка на другой. Существуют конверторы для перевода программ как с одного языка высокого уровня на другой язык высокого уровня, так и конверторы с некоторых специализированных входных языков различных систем на более универсальные промежуточные (внутренние) языки, для которых существуют готовые компиляторы или интерпретаторы

7.Кросс-системы наиболее часто реализуются в виде кросс-компиляторов и кросс-ассемблеров. Они обеспечивают получение на одной ЭВМ, называемой инструментальной, машинных программ, предназначенных для выполнения на другой ЭВМ, называемой целевой.

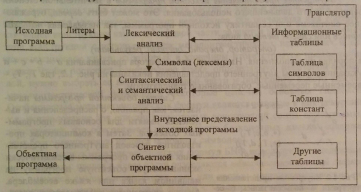


Рис. Функциональная модель транслятора

1. Виды сетей ЭВМ. Архитектура сетей.

Системы обработки данных классифицируются на две группы: сосредоточенные (централизованные) и распределенные. Распределенная обработка данных – обработка данных, выполняемая на независимых, но связанных между собой компьютерах, представляющих распределенную систему. Для реализации такой обработки были созданы многомашинные ассоциации, структура которых разрабатывается по одному из следующих направлений: многомашинные вычислительные комплексы (МВК), компьютерные (вычислительные) сети.

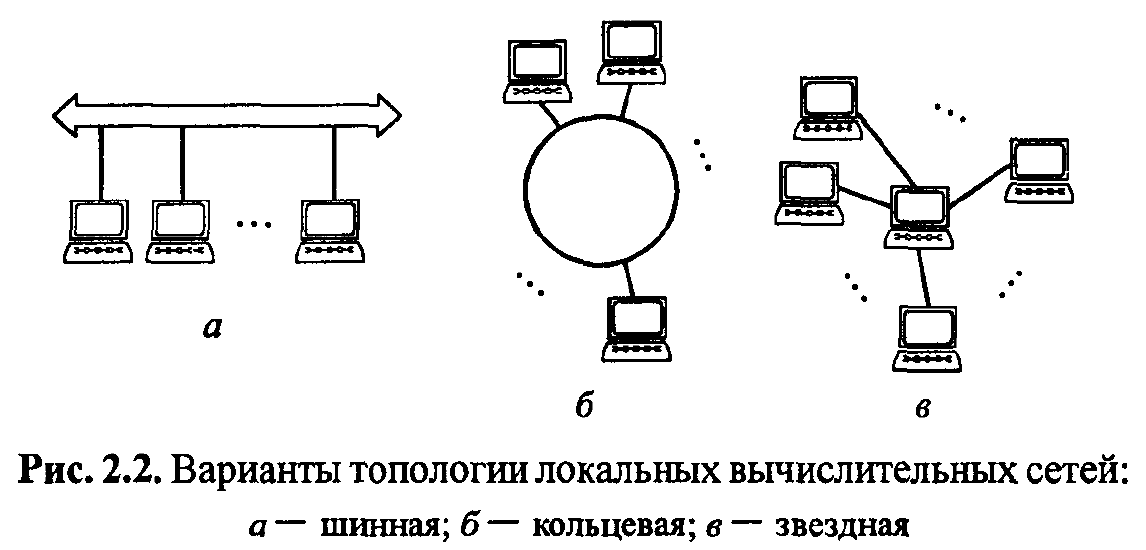
МВК - группа установленных рядом вычислительных машин, объединенных с помощью специальных средств сопряжения и выполняющих совместно единый вычислительный процесс.

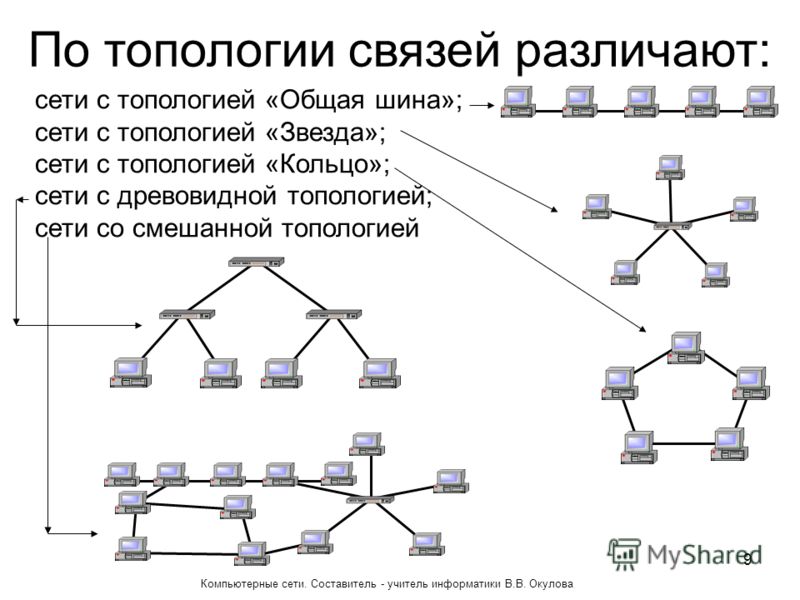
Компьютерная (вычислительная) сеть - совокупность компьютеров и терминалов, соединенных с помощью каналов связи в единую систему, удовлетворяющую требованиям распределенной обработки данных.

Классификация:

Масштаб: глобальные (Wide Area Network - WAN), региональные (Metropolitan AN), локальные (Local AN)

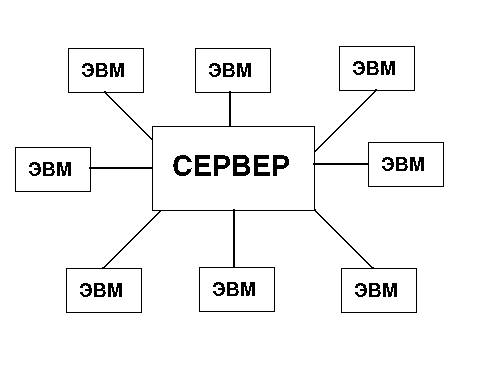
Топология: шина (магистральная), кольцо, звезда, иерархия, смешанная.





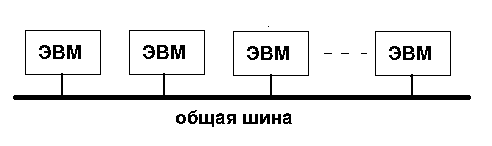
По топологии (способу объединения ЭВМ) различают:

* звездообразную топологию



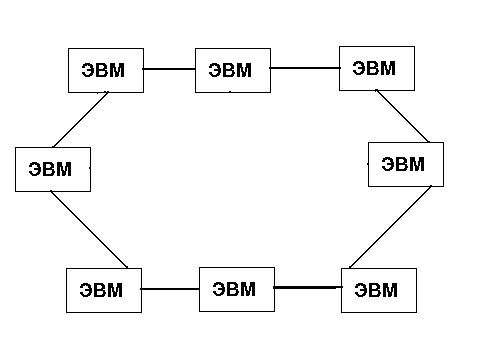
При таком способе обмен данными между ЭВМ осуществляется через более мощную ЭВМ - сервер. Недостатком такого соединения является низкая живучесть сети - выход из строя сервера означает прекращение функционирование сети. Однако, простота и дешевизна реализации сделала эту структуру популярной в локальных сетях.

* топологию с общей шиной



При этом способе обмен данными происходит через общую шину, которую используя механизм прерывания может "захватывать" тот или иной компьютер. Характерной особенностью здесь является отсутствие сервера. Очень часто используется в локальных сетях, а уж в "домашних" повсеместно.

* кольцевая топология



В этой структуре каждая ЭВМ используя механизм прерывания работает в качестве ретранслятора. Обратите внимание, живучесть сети повышена - при одиночном обрыве связи между соседними ЭВМ сеть продолжает функционировать.

1. Понятие о стандарте OSI. Уровни OSI и их назначение.

Для удобства модернизации сложных информационных систем их делают максимально открытыми, т. е. приспособленными для внесения изменении в некоторую часть системы при сохранении неизменными остальных частей. В отношении вычислительных сетей реализация концепции открытости привела к появлению эталонной модели взаимосвязи открытых систем (ЭМВОС), предложенной Международной организацией стандартизации (ISO — International Standard Organization). В этой модели дано описание общих принципов, правил, соглашений, обеспечивающих взаимодействие информационных систем и называемых протоколами.

Информационную сеть в ЭМВОС рассматривают как совокупность функций (протоколов), которые подразделяют на группы, называемы уровнями.

Именно разделение на уровни позволяет вносить изменения в средства реализации одного уровня без перестройки средств других уровней, что значительно упрощает и удешевляет модернизацию средств по мере развития техники. Различают семь уровней модели:

На **физическом** (physical) уровне осуществляется представление информации в виде электрических или оптических сигналов, преобразование формы сигналов, выбор параметров физических сред передачи данных, организуется передача информации через физические среды.

На **канальном** (link) уровне выполняется обмен данными между соседними узлами сети, т. е. узлами, непосредственно связанными физическими соединениями без других промежуточных узлов. Отметим, что пакеты канального уровня обычно называют кадрами.

На **сетевом** (network) уровне происходит формирование пакетов по правилам тех промежуточных сетей, через которые проходит исходный пакет, и маршрутизация пакетов, т. е. определение и реализация маршрутов, по которым передаются пакеты. Другими словами, маршрутизация сводится к образованию логических каналов. Логическим каналом называют виртуальное соединение двух или более объектов сетевого уровня, при котором возможен обмен данными между этими объектами. Еще одной важной функцией сетевого уровня после маршрутизации является контроль нагрузки на сеть с целью предотвращения перегрузок, отрицательно влияющих на работу сети.

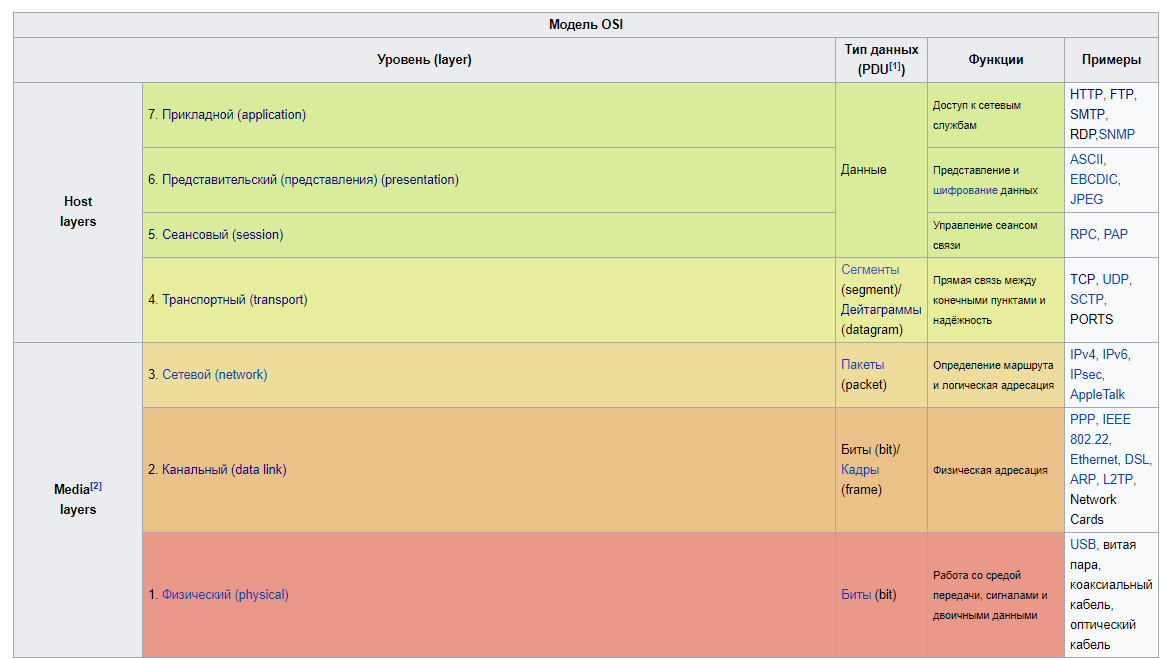
На **транспортном** (transport) уровне обеспечивается связь между оконечными пунктами (в отличие от предыдущего сетевого уровня, на котором обеспечивается передача данных через промежуточные компоненты сети). К функциям транспортного уровня относятся мультиплексирование и демультиплексирование (сборка-разборка сообщений на пакеты в конечных пунктах), обнаружение и устранение ошибок в переданных данных, задание требуемого уровня услуг (например, заказанных скорости и надежности передачи).

На **сеансовом** (session) уровне определяются тип связи (дуплекс или полудуплекс), начало и окончание заданий, последовательность и режим обмена запросами и ответами взаимодействующих партнеров.

На **представительном** (presentation) уровне реализуются функции представления данных (кодирование, форматирование, структурирование). Например, на этом уровне выделенные для передачи данные преобразуются из одного кода в другой, в частности, с целью шифрования.

На **прикладном** (application) уровне определяются и оформляются в сообщения те данные, которые подлежат передаче по сети.

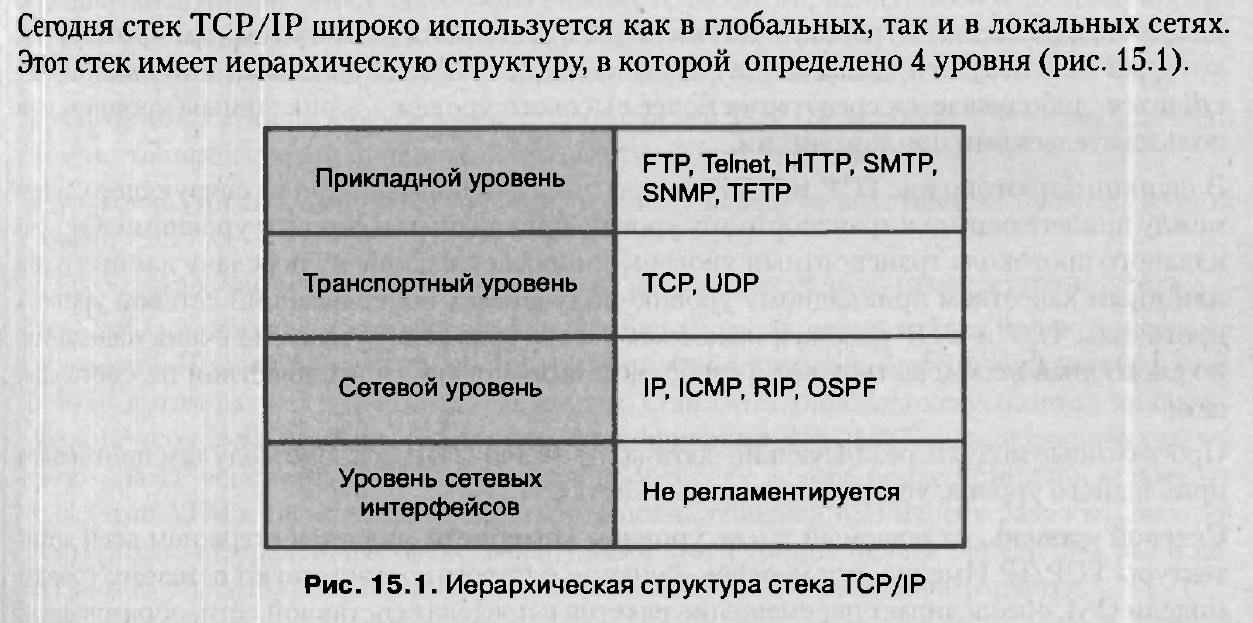
В конкретных случаях может возникать потребность в реализации лишь части названных функций, тогда соответственно сеть будет содержать лишь часть уровней. Так, в простых (неразветвленных) ЛВС отпадает необходимость в средствах сетевого и транспортного уровней.



1. Понятие протокола, стек протоколов, протокол TCP/TP (общие характеристики).

Протокол – набор правил, определяющий взаимодействие двух одноименных уровней модели взаимодействия открытых систем в различных абонентских ЭВМ. Правила и последовательность выполнения действий при обмене информацией, определенные протоколом, должны быть реализованы в программе. Все компьютеры, участвующие в обмене, должны работать по одним и тем же протоколам, чтобы по завершении передачи информация восстанавливалась в первоначальном виде.





**Прикладной** уровень стека TCP/IP соответствует трем верхним уровням модели OSI: прикладному, представления и сеансовому. Он объединяет сервисы, предоставляемые системой пользовательским приложениям. Протоколы прикладного уровня развертываются на хостах.

**Транспортный** уровень стека TCP/IP может предоставлять вышележащему уровню два типа сервиса:

1.гарантированную доставку обеспечивает протокол управления передачей (Transmission Control Protocol, ТСР);

2.доставку по возможности, или c максимальными усилиями, обеспечивает протокол пользовательских дейтаграмм (User Datagram Protocol, UDP).

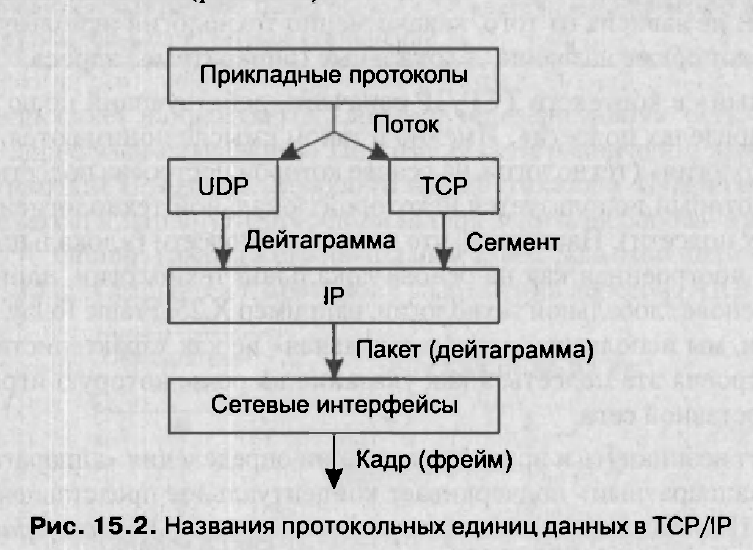
Программные модули, реализующие протоколы ТСР и UDP, подобно модулям протоколов прикладного уровня, устанавливаются на хостах.

**Сетевой** уровень, называемый также уровнем Интернета, является стержнем всей архитектуры ТСР/IP. Именно этот уровень, функции которого соответствуют сетевому уровню модели OSI, обеспечивает перемещение пакетов в пределах составной сети, образованной объединением нескольких подсетей. Протоколы сетевого уровня поддерживают интерфейс с вышележащим транспортным уровнем, получая от него запросы на передачу данных по составной сети, а также с нижележащим уровнем сетевых интерфейсов.

У **уровня сетевых интерфейсов** задача существенно проще - он отвечает только за организацию взаимодействия с подсетями разных технологий, входящими в составную сеть. TCP/IP рассматривает любую подсеть, входящую в составную сеть, как средство транспортировки пакетов между двумя соседними маршрутизаторами. Задачу организации интерфейса между технологией TCP/ IP и любой другой технологией промежуточной сети упрощенно можно свести к двум задачам:

1.упаковка (инкапсуляция) IР-пакета в единицу передаваемых данных промежуточной сети;

2.преобразование сетевых адресов в адреса технологии данной промежуточной сети.



1. Сетевое программное обеспечение (браузеры, фаерволы, поисковики).

Браузер, или веб-обозреватель (web browser) — прикладное программное обеспечение для просмотра веб-страниц, содержания веб-документов, компьютерных файлов и их каталогов; управления веб-приложениями; а также для решения других задач. В глобальной сети браузеры используют для запроса, обработки, манипулирования и отображения содержания веб-сайтов. Браузер находит веб-страницы и отдельные объекты по адресам URL (Uniform Resource Locator), состоящих из трех частей (тип протокола доступа, dns-имя сервера, путь к объекту. Пример: http://www.rsreu.ru)

Многие современные браузеры также могут использоваться для обмена файлами с серверами ftp, а также для непосредственного просмотра содержания файлов многих графических форматов (gif, jpeg, png, svg), аудио-видео форматов (mp3, mpeg), текстовых форматов (pdf, djvu) и других файлов.

Межсетевой экран, сетевой экран — программный или программно-аппаратный элемент компьютерной сети, осуществляющий контроль и фильтрацию проходящего через него сетевого трафика в соответствии с заданными правилами. Другие названия: брандмауэр (нем. Brandmauer — противопожарная стена); файервол (англ. Firewall — противопожарная стена).

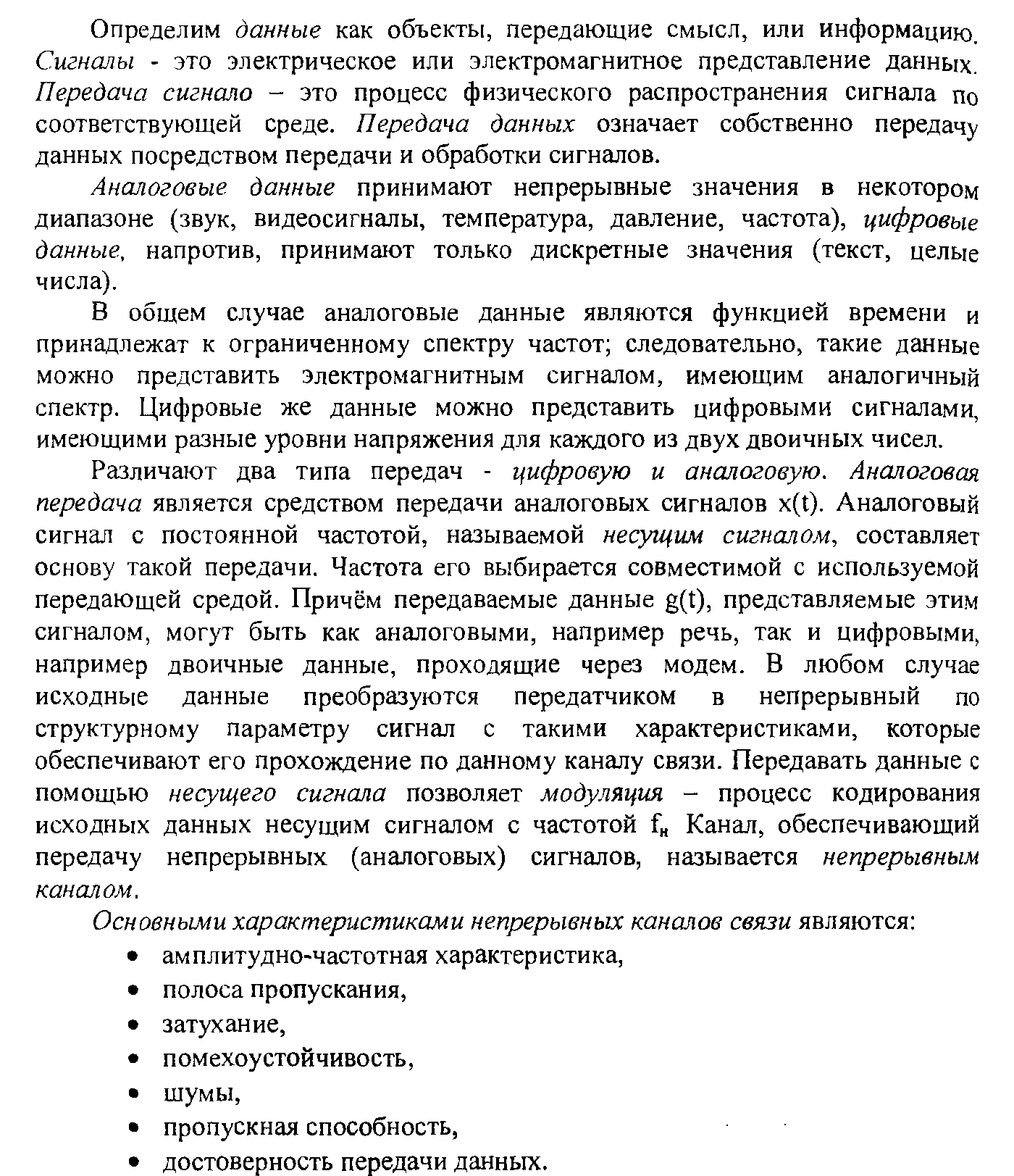
Среди задач, которые решают межсетевые экраны, основной является защита сегментов сети или отдельных хостов от несанкционированного доступа с использованием уязвимых мест в протоколах сетевой модели OSI или в программном обеспечении, установленном на компьютерах сети. Межсетевые экраны пропускают или запрещают трафик, сравнивая его характеристики с заданными шаблонами.

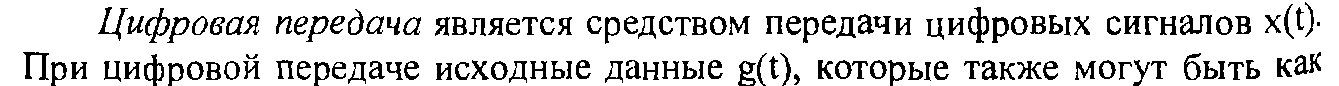
Наиболее распространённое место для установки межсетевых экранов — граница периметра локальной сети для защиты внутренних хостов от атак извне. Однако атаки могут начинаться и с внутренних узлов — в этом случае, если атакуемый хост расположен в той же сети, трафик не пересечёт границу сетевого периметра, и межсетевой экран не будет задействован. Поэтому в настоящее время межсетевые экраны размещают не только на границе, но и между различными сегментами сети, что обеспечивает дополнительный уровень безопасности.

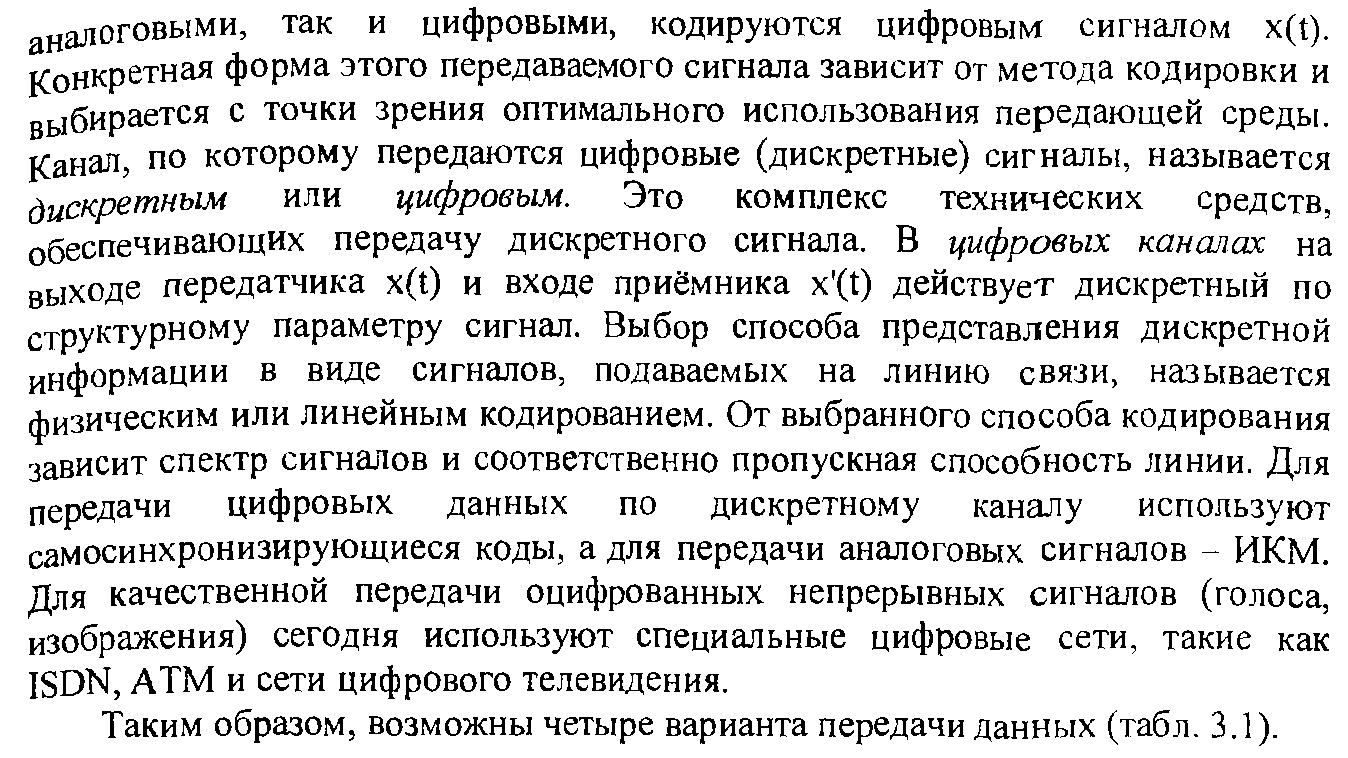
Поиско́вая систе́ма (англ. search engine) — это компьютерная система, предназначенная для поиска информации. Одно из наиболее известных применений поисковых систем — веб-сервисы для поиска текстовой или графической информации во Всемирной паутине. Существуют также системы, способные искать файлы на FTP-серверах, товары в интернет-магазинах, информацию в группах новостей Usenet.

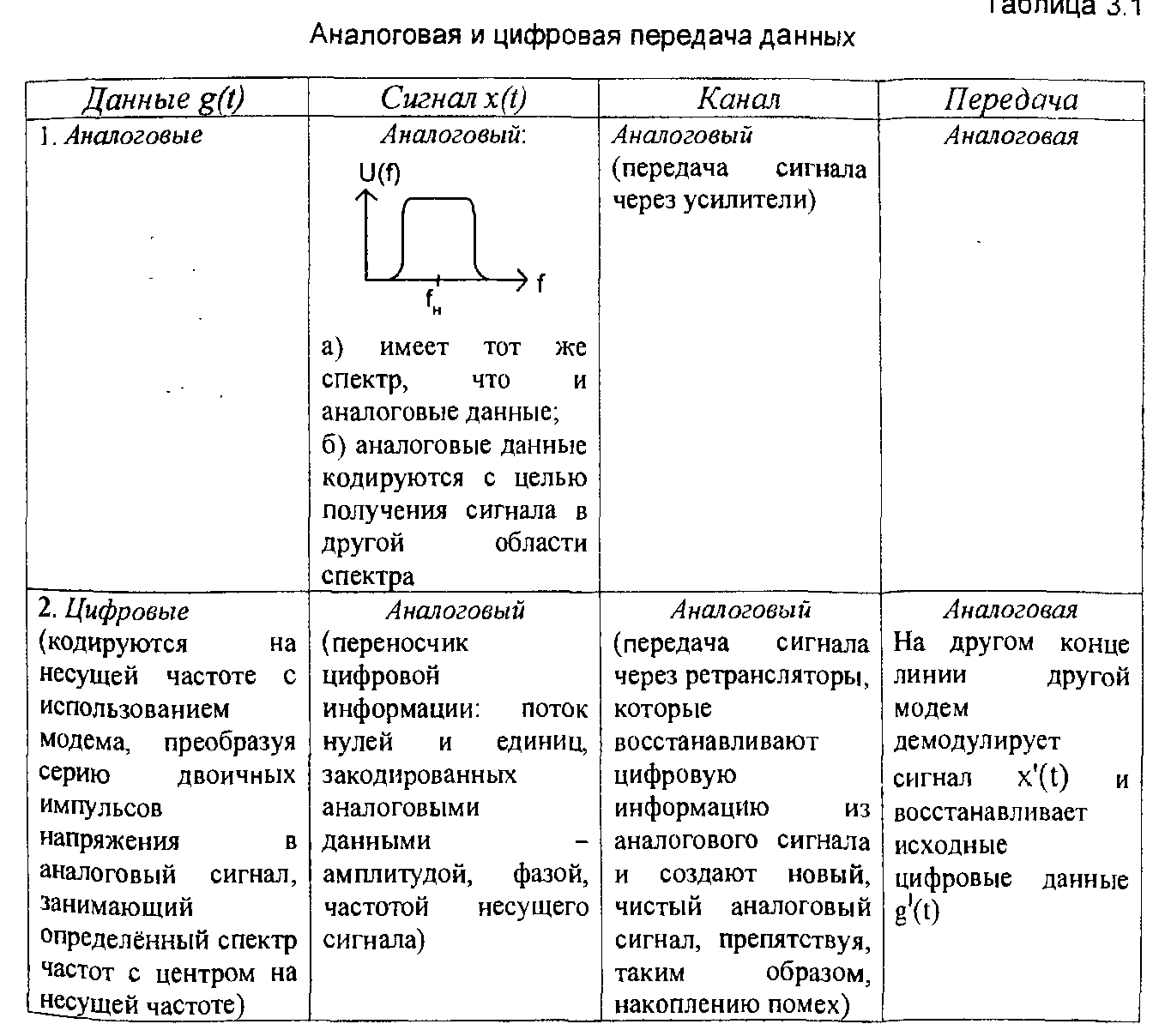
Для поиска информации с помощью поисковой системы пользователь формулирует поисковый запрос. Работа поисковой системы заключается в том, чтобы по запросу пользователя найти документы, содержащие либо указанные ключевые слова, либо слова, как-либо связанные с ключевыми словами. При этом поисковая система генерирует страницу результатов поиска. Такая поисковая выдача может содержать различные типы результатов, например: веб-страницы, изображения, аудиофайлы. Некоторые поисковые системы также извлекают информацию из подходящих баз данных и каталогов ресурсов в Интернете.

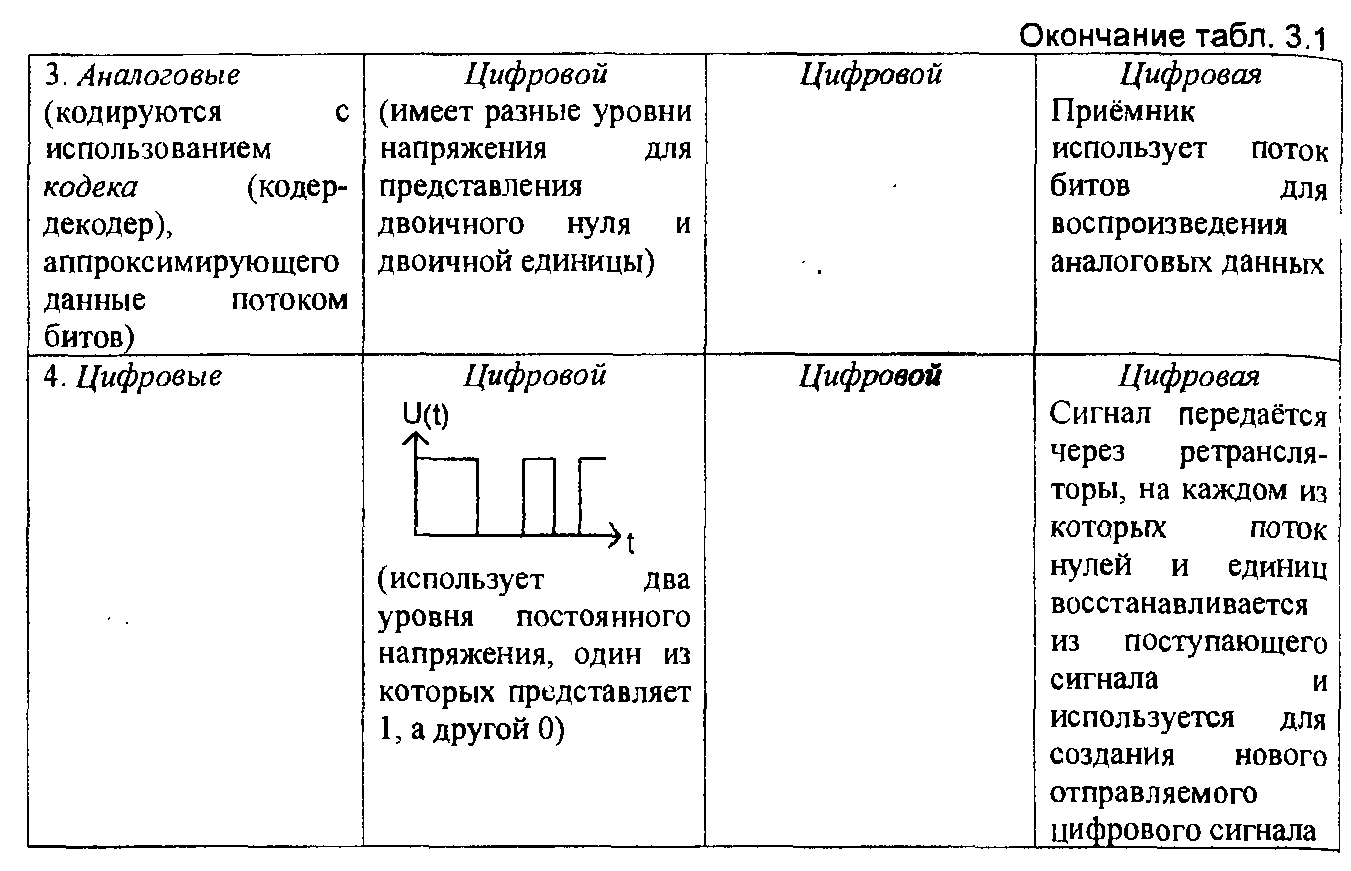
1. Способы передачи данных: модуляция, защита от ошибок, сжатие данных.

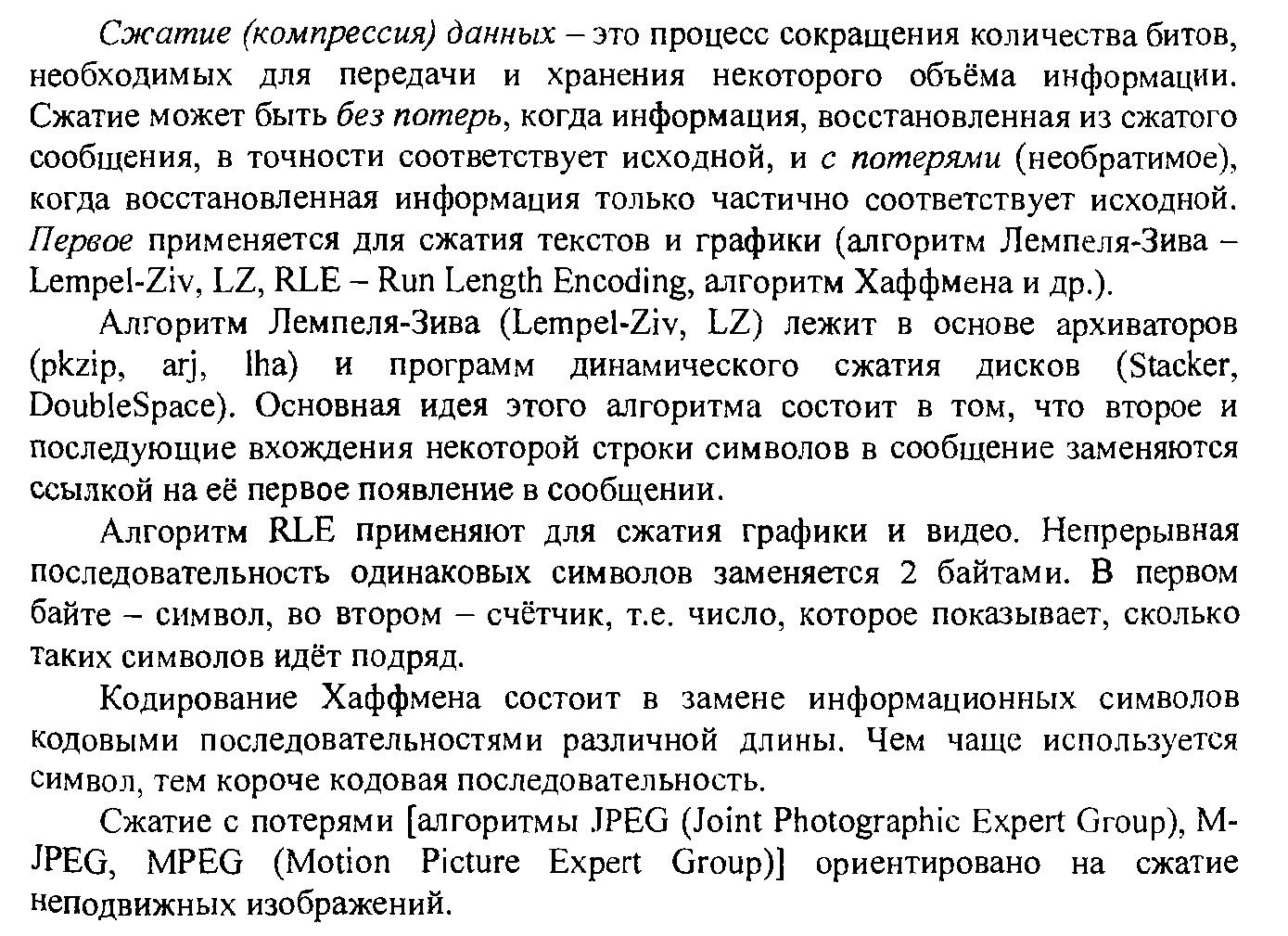






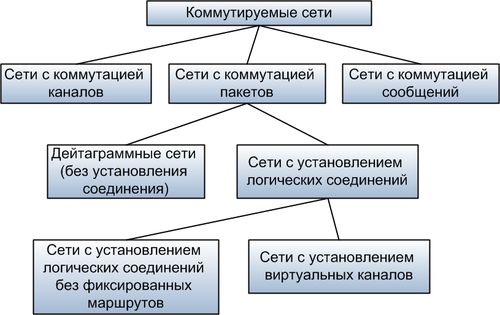






1. Сети с коммутацией каналов.

*Коммутация* — это процесс соединения различных абонентов коммуникационной сети через транзитные узлы. Коммуникационные сети должны обеспечивать связь своих абонентов между собой. Абонентами могут выступать ЭВМ, сегменты локальных сетей, факс-аппараты или телефонные собеседники.



***Коммутация каналов***

Коммутация каналов подразумевает образование непрерывного составного физического канала из последовательно соединенных отдельных канальных участков для прямой передачи данных между узлами. Отдельные каналы соединяются между собой специальной аппаратурой - коммутаторами, которые могут устанавливать связи между любыми конечными узлами сети. В сети с коммутацией каналов перед передачей данных всегда необходимо выполнить процедуру установления соединения, в процессе которой и создается составной канал.

Время передачи сообщения при этом определяется пропускной способностью канала, длинной связи и размером сообщения.

Коммутаторы, а также соединяющие их каналы должны обеспечивать одновременную передачу данных нескольких абонентских каналов. Для этого они должны быть высокоскоростными и поддерживать какую-либо технику мультиплексирования абонентских каналов.

*Достоинства коммутации каналов:*

1) постоянная и известная скорость передачи данных;

2) правильная последовательность прихода данных;

3) низкий и постоянный уровень задержки передачи данных через сеть.

*Недостатки коммутации каналов:*

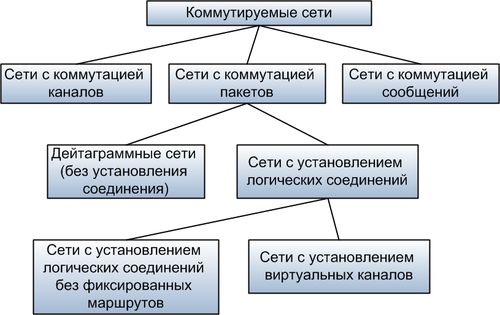
1) возможен отказ сети в обслуживании запроса на установление соединения;

2) нерациональное использование пропускной способности физических каналов, в частности невозможность применения пользовательской аппаратуры, работающей с разной скоростью. Отдельные части составного канала работают с одинаковой скоростью, так как сети с коммутацией каналов не буферизуют данные пользователей;

3) обязательная задержка перед передачей данных из-за фазы установления соединения.

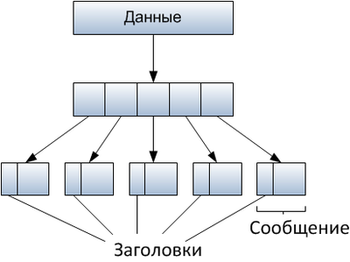
1. Сети с коммутацией сообщений.

*Коммутация* — это процесс соединения различных абонентов коммуникационной сети через транзитные узлы. Коммуникационные сети должны обеспечивать связь своих абонентов между собой. Абонентами могут выступать ЭВМ, сегменты локальных сетей, факс-аппараты или телефонные собеседники.



***Коммутация сообщений***

Коммутация сообщений – разбиение информации на сообщения, каждый из которых состоит из заголовка и информации.



Это способ взаимодействия, при котором создается логический канал, путем последовательной передачи сообщений через узлы связи по адресу указанному в заголовке сообщения.

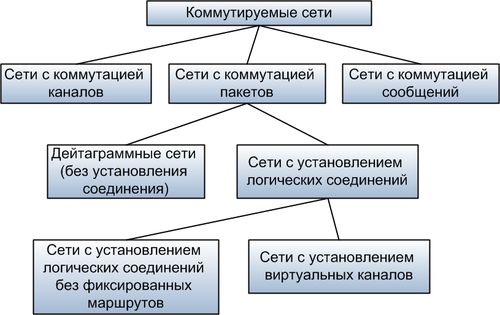
При этом каждый узел принимает сообщение, записывает в память, обрабатывает заголовок, выбирает маршрут и выдает сообщение из памяти в следующий узел.

Время доставки сообщения определяется временем обработки в каждом узле, числом узлов и пропускной способности сети. Когда заканчивается передача информации из узла А в узел связи В, то узел А становится свободным и может участвовать в организации другой связи между абонентами, поэтому канал связи используется более эффективно, но система управления маршрутизации будет сложной.

Сегодня коммутация сообщений в чистом виде практически не существует.

1. Сети с коммутацией пакетов.

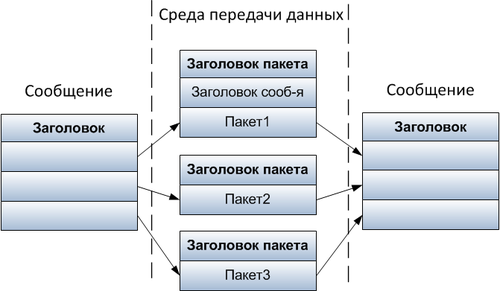
*Коммутация* — это процесс соединения различных абонентов коммуникационной сети через транзитные узлы. Коммуникационные сети должны обеспечивать связь своих абонентов между собой. Абонентами могут выступать ЭВМ, сегменты локальных сетей, факс-аппараты или телефонные собеседники.



***Коммутация пакетов***

Коммутация пакетов - это особый способ коммутации узлов сети, который специально создавался для наилучшей передачи компьютерного трафика (пульсирующего трафика). Опыты по разработке самых первых компьютерных сетей, в основе которых лежала техника коммутации каналов, показали, что этот вид коммутации не предоставляет возможности получить высокую пропускную способность вычислительной сети. Причина крылась в пульсирующем характере трафика, который генерируют типичные сетевые приложения.

При коммутации пакетов все передаваемые пользователем сети сообщения разбиваются в исходном узле на сравнительно небольшие части, называемые пакетами. Необходимо уточнить, что сообщением называется логически завершенная порция данных - запрос на передачу файла, ответ на этот запрос, содержащий весь файл, и т. п. Сообщения могут иметь произвольную длину, от нескольких байт до многих мегабайт. Напротив, пакеты обычно тоже могут иметь переменную длину, но в узких пределах, например от 46 до 1500 байт (EtherNet). Каждый пакет снабжается заголовком, в котором указывается адресная информация, необходимая для доставки пакета узлу назначения, а также номер пакета, который будет использоваться узлом назначения для сборки сообщения.



Коммутаторы пакетной сети отличаются от коммутаторов каналов тем, что они имеют внутреннюю буферную память для временного хранения пакетов, если выходной порт коммутатора в момент принятия пакета занят передачей другого пакета.

*Достоинства коммутации пакетов:*

1) более устойчива к сбоям;

2) высокая общая пропускная способность сети при передаче пульсирующего трафика;

3) возможность динамически перераспределять пропускную способность физических каналов связи.

*Недостатки коммутации пакетов:*

1) неопределенность скорости передачи данных между абонентами сети;

2) переменная величина задержки пакетов данных;

3) возможны потери данных из-за переполнения буферов;

4) возможны нарушения последовательности прихода пакетов.

В компьютерных сетях применяется коммутация пакетов.

1. Маршрутизация в телекоммуникационных системах.

Теперь мы вплотную подошли к проблеме выбора оптимального маршрута доставки информации от отправителя к получателю. Этот процесс называется ***маршрутизацией***, а выбор маршрута производится в соответствии с используемым ***алгоритмом маршрутизации***.

ПРИМЕЧАНИЕ

Если говорить о сетях, то в подсети, использующей дейтаграммную службу выбор маршрута для каждого пакета должен производиться заново, т.к. оптимальный маршрут мог измениться. Если используются виртуальные каналы, маршрут выбирается только при создании нового виртуального канала [12].

***Основную цель маршрутизации*** можно определить следующим образом – доставка пакета получателю за минимально возможное время, при сохранении требуемой пропускной способности и минимальных потерях информации. Здесь приходится учитывать и топологию сети и то, что маршрутизаторы (коммуникационные устройства, осуществляющие пересылку и маршрутизацию), а также линии связи могут выйти из строя.

В отечественной литературе, например, в [2] выделяют три способа маршрутизации: ***централизованная маршрутизация*** (выбор маршрута для каждого пакета осуществляется в центре управления сетью, что чревато отказом всей сети при выходе из строя центрального узла); ***распределенная*** ***маршрутизация*** (функции управления маршрутизацией распределены между узлами сети, а значит, обеспечивается большая гибкость);***смешанная маршрутизация*** сочетает принципы централизованной и распределенной маршрутизации.

Почему вопросами создания алгоритмов маршрутизации до сих пор занимается огромное количество ученых и инженеров, несмотря на то, что технология с успехом работает много лет? Ответ предельно просто и заключается в следующем – мы не знаем как будет изменяться нагрузка в сети в следующий момент времени. Хотя крайне важными параметрами остаются топология сети и ее изменение (в результате отказов узлов и линий связи, а также при подключении новых), а также различная пропускная способность участков сети. Но оба эти параметра можно определить и передать маршрутизаторам. А значит, во всех случаях алгоритмы маршрутизации выполняются в условиях неопределенности текущего и будущего состояний телекоммуникационной системы [2].

Рассмотрим несколько наиболее известных и широко применяемых алгоритмов маршрутизации. Сначала давайте проведем черту между существующими алгоритмами. Они разделяются на два больших класса:***адаптивные*** и ***неадаптивные***. Вместо того чтобы учитывать топологию сети и изменение ее состояния, а также измерять текущий трафик ***неадаптивные алгоритмы*** выбирают маршруты заранее. Полученный список маршрутов загружается в маршрутизаторы на этапе загрузке сети. Эта процедура называется ***статической маршрутизацией***. ***Адаптивные алгоритмы*** напротив охотно пользуются всеми параметрами, которые можно измерить.

Вне зависимости от топологии сети и интенсивности трафика все алгоритмы маршрутизации базируются на***принципе оптимальности*** и ***концепции кратчайшего пути***. В соответствии с ***принципом оптимальности***, если маршрутизатор *B* располагается на оптимальном маршруте от маршрутизатора *A* к маршрутизатору *C*, то оптимальный маршрут от маршрутизатора *B* к маршрутизатору *C* совпадает с частью первого маршрута [12].

ПРИМЕЧАНИЕ

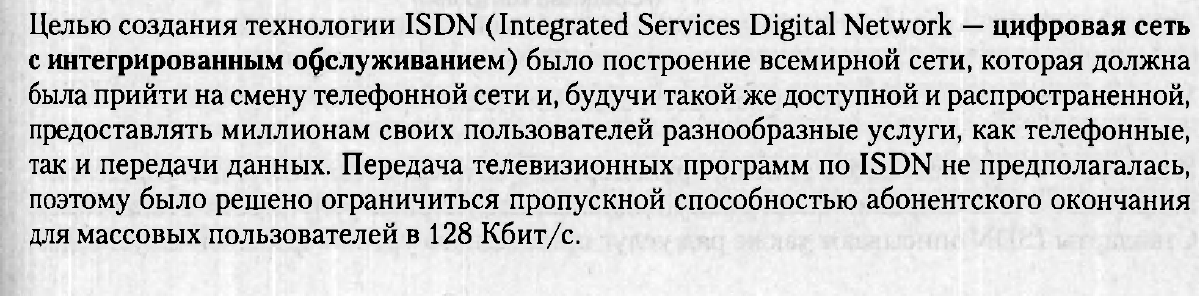
Концепция кратчайшего пути, чаще всего поясняется на графах, где каждый узел сети – вершина графа, а дуга – линия связи. В этом случае для поиска кратчайшего пути на графах используется, например, алгоритм Дейкстры.

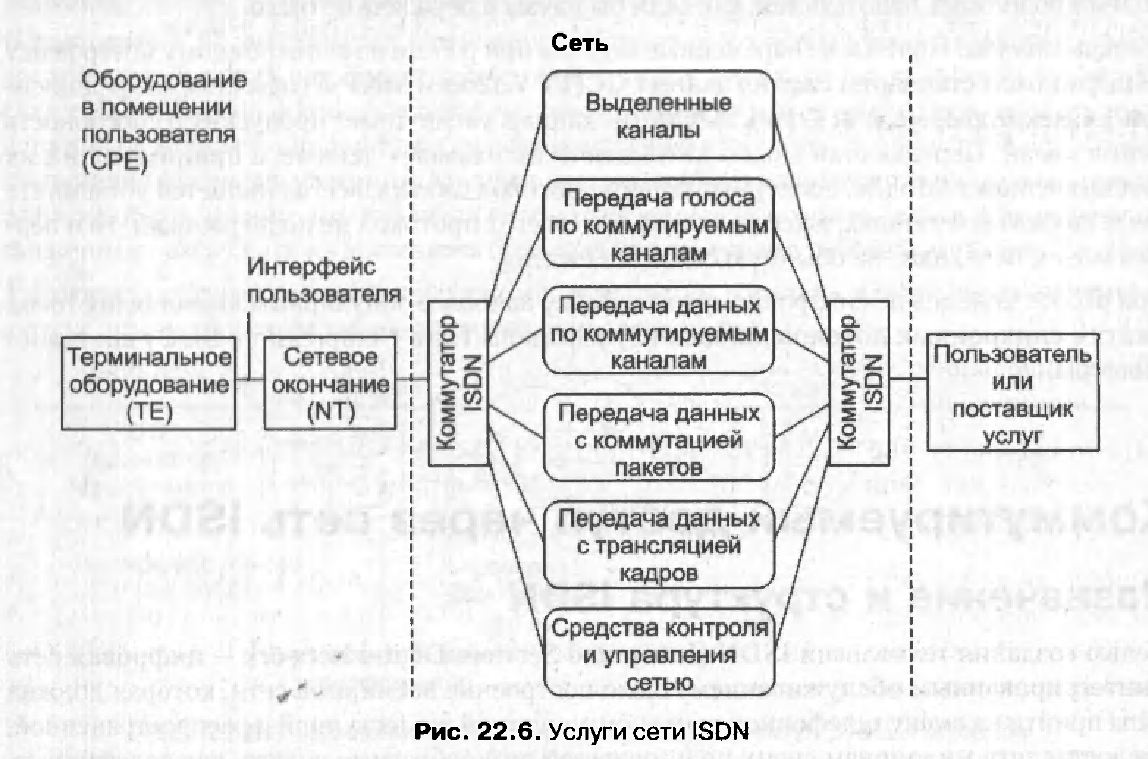
Наиболее простой статический алгоритм маршрутизации – ***заливка***. Здесь можно провести аналогию с методом полного перебора из методов оптимизации и поиска экстремума функции. Суть ***заливки***заключается в том, что каждый пришедший пакет посылается на все исходящие линии, кроме той, по которой он поступил. Это порождает бесконечное число дублированных пакетов. Для ограничения количества тиражируемых пакетов используются счетчики, которые помещаются в заголовок пакета и уменьшаются при прохождении каждого маршрутизатора. Если счетчик обнуляется, то такой пакет удаляется. Кроме данного метода применяются еще ряд вариаций на тему счетчиков и помещения в заголовки номера пройденного узла.

Однако на практике применяется ***выборочная заливка***. Отличие его состоит в том, что пакеты посылаются не на все исходящие линии, а только на те, которые идут в приблизительно правильном направлении. Такой «громоздкий» алгоритм может подойти военным, где вероятность потери части сети из-за боевых действий велика, а также для тестирования других алгоритмов. Напомним, что также как и метод полного перебора, заливка рано или поздно найдет оптимальный маршрут.

На практике, современные системы телекоммуникаций применяю не статические, а динамические алгоритмы маршрутизации.

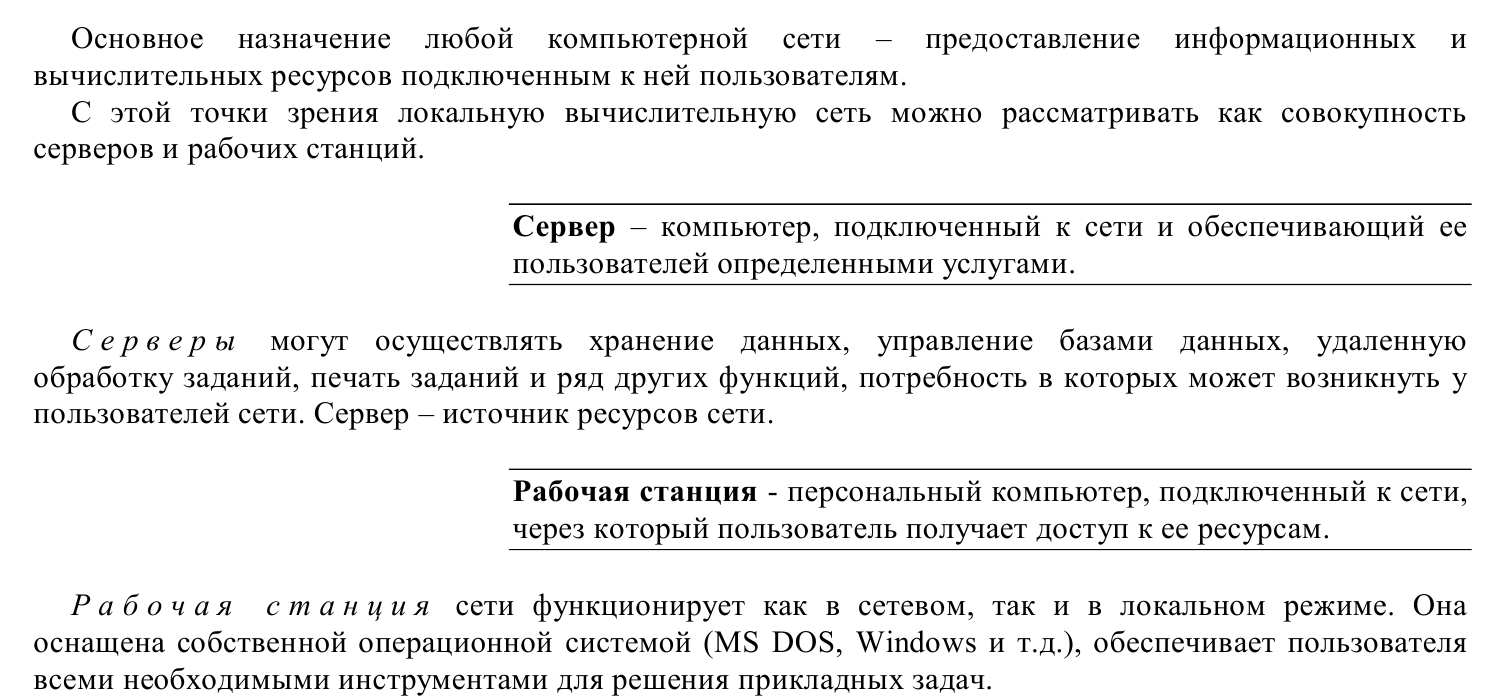
1. Цифровые сети с интеграцией служб (ISDN).



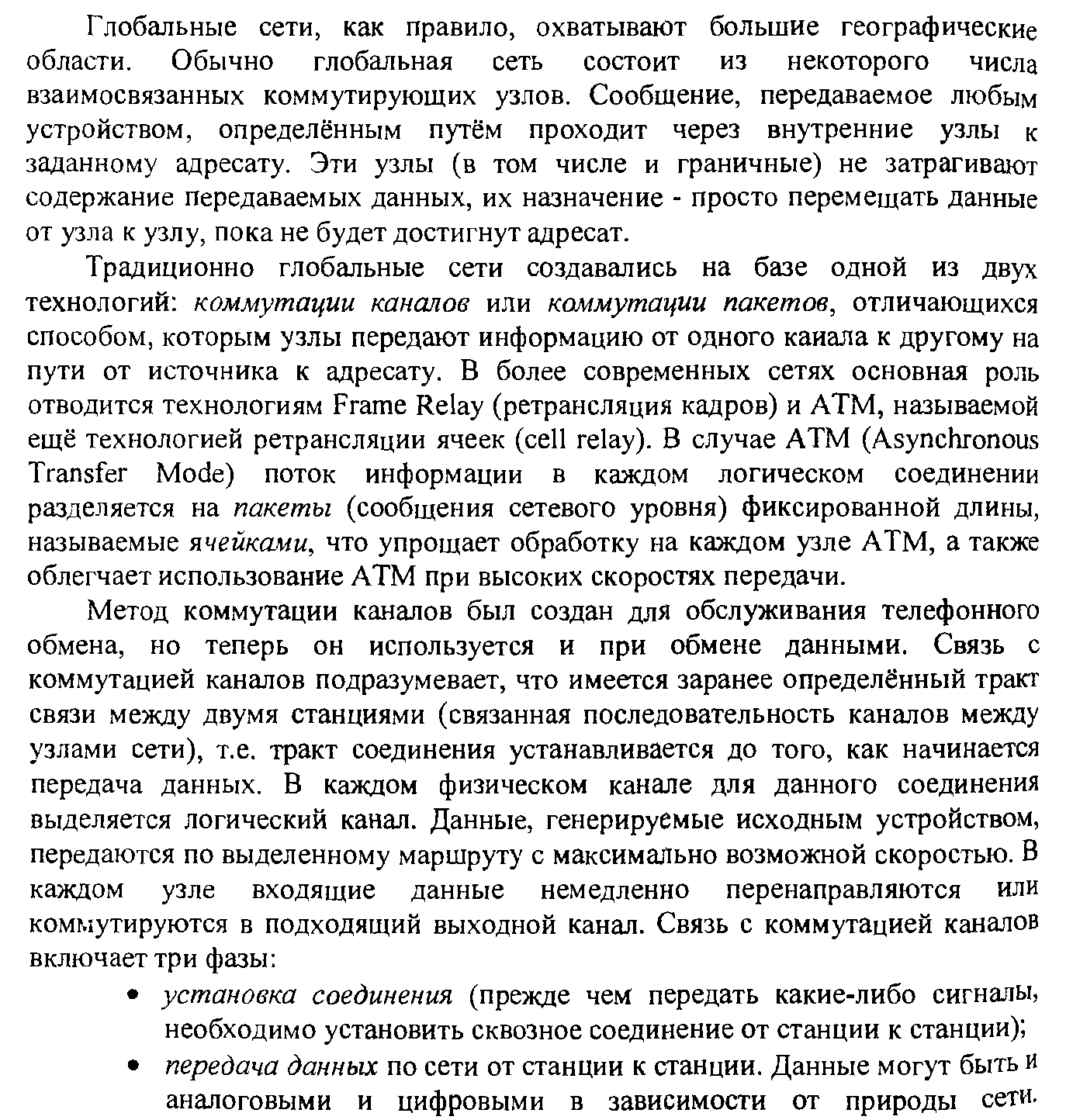


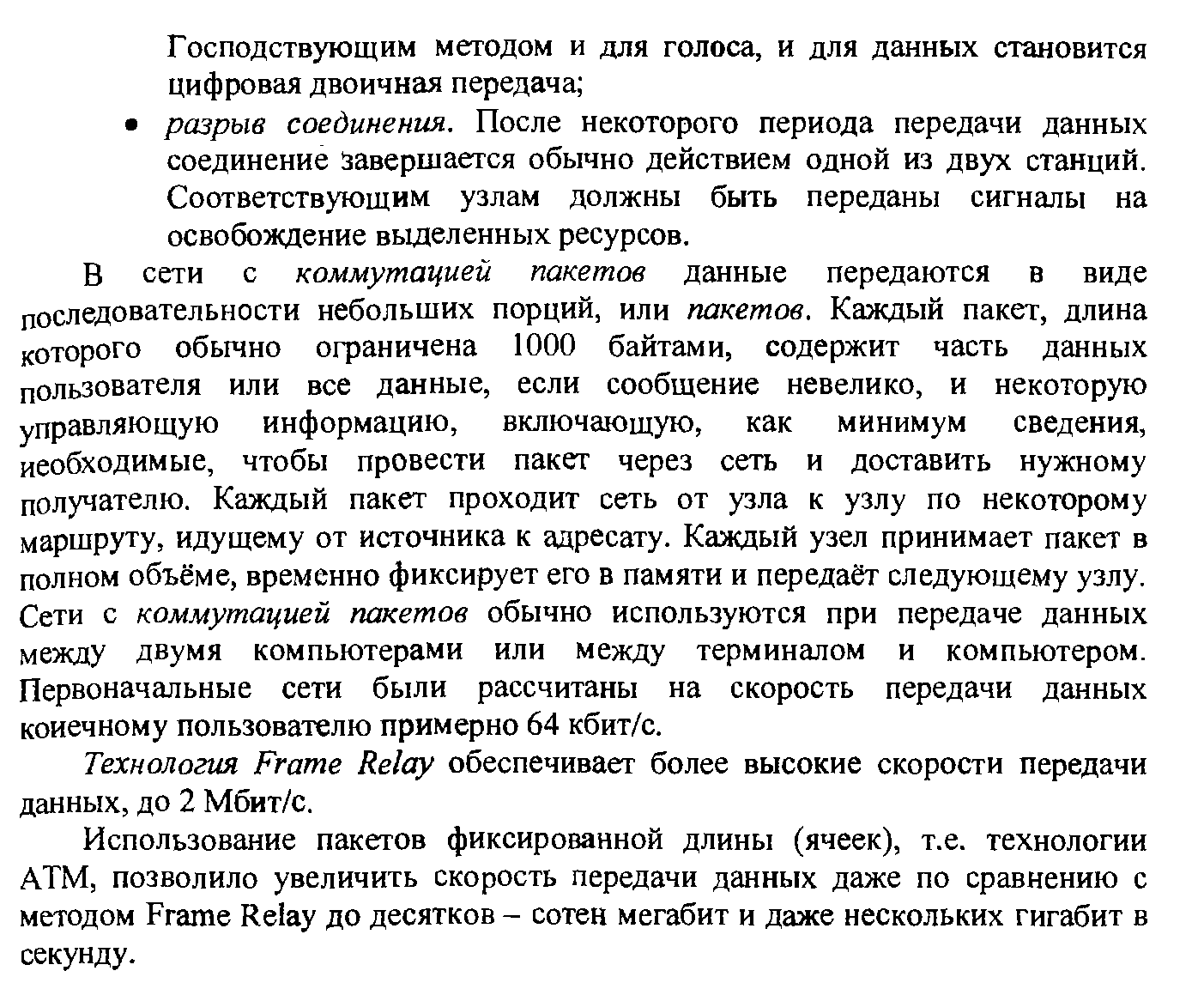
1. Локальные вычислительные сети.

Локальной называется вычислительная сеть, системы которой расположены на небольшом расстоянии друг от друга. При таких расстояниях наиболее эффективное средство связи между системами, объединенными в сеть, – последовательный интерфейс. В качестве передающей среды используются витая пара проводников, коаксиальные и волоконно-оптические кабели, при этом в большинстве случаев достаточно иметь один канал – моноканал, обслуживающий все системы ЛВС в режиме мультиплексирования. На основе моноканала строятся две основные конфигурации: магистральная (шинная) и кольцевая. В ЛВС могут использоваться и другие конфигурации связей между системами – звездообразная и произвольная.



1. Глобальные вычислительные сети.





1. Топология сетей ЭВМ.

Термин топология, или более точно, сетевая топология – это способ организации связей компьютеров в сети. Топология – это стандартный термин, которым пользуются специалисты, когда характеризуют физические или логические связи компьютеров в сети.

Следует различать физическую и логическую топологии.

Физическая топология определяет конфигурацию физических связей, т.е. электрических соединений компьютеров в сети.

Логическая топология определяет маршруты передачи данных между компонентами сети и определяется способом работы коммуникационного оборудования.

Далее рассматриваются базовые физические топологии сети.

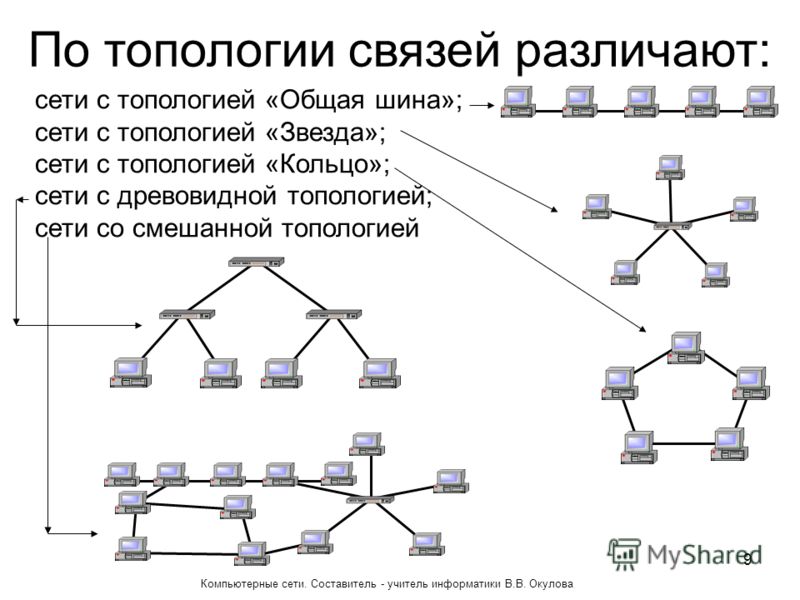
Структура связей в ЛВС легко прослеживается.

Обычно выделяют три базовых (основных) топологии сети:

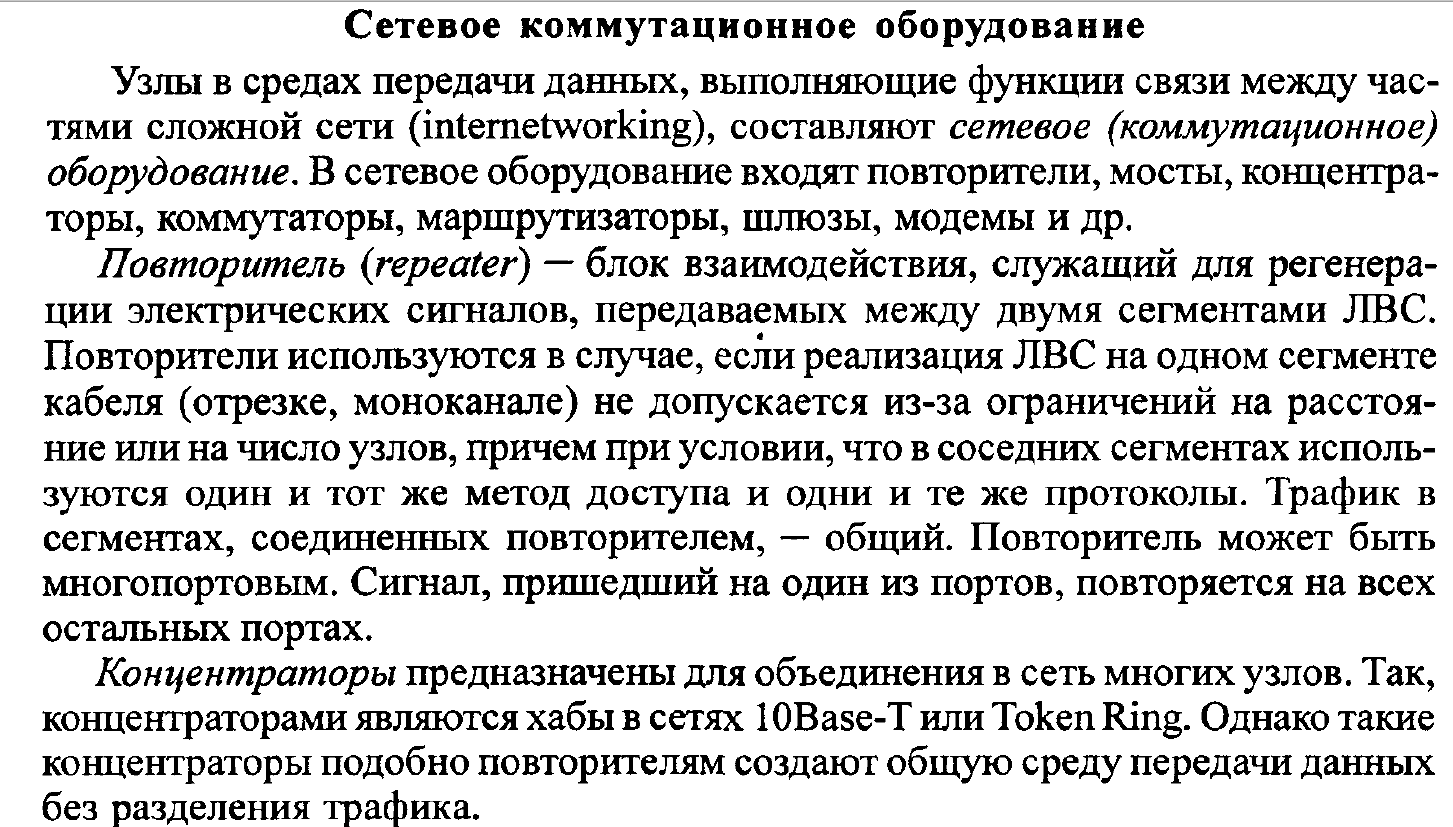
Шинная (bus), при которой все рабочие станции подключаются к общей разделяемой (совместно используемой - share) передающей среде - магистрали (шине) с помощью сетевых адаптеров (устройств сопряжения компьютера с линией связи); конструктивно адаптер представляет собой плату - карту, встраиваемую в компьютер. Аналогичным образом к шине подключаются и другие сетевые устройства.

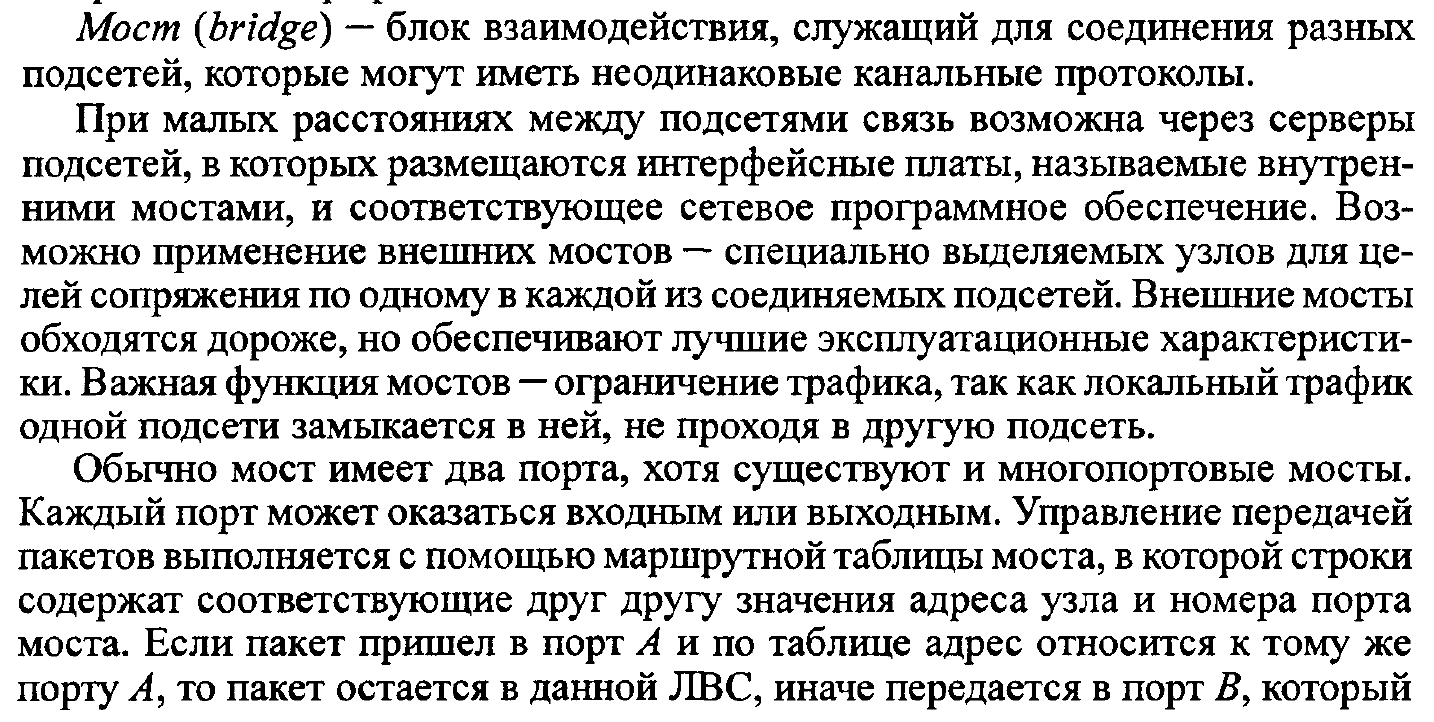
Звездообразная (star) или радиальная, при которой к одному центральному узлу присоединяются все остальные компьютеры, причем каждый из них использует свою отдельную линию связи. Сообщения от передающей станции поступают на адаптеры всех рабочих станций, однако воспринимаются только адаптером той станции, которой они адресованы.

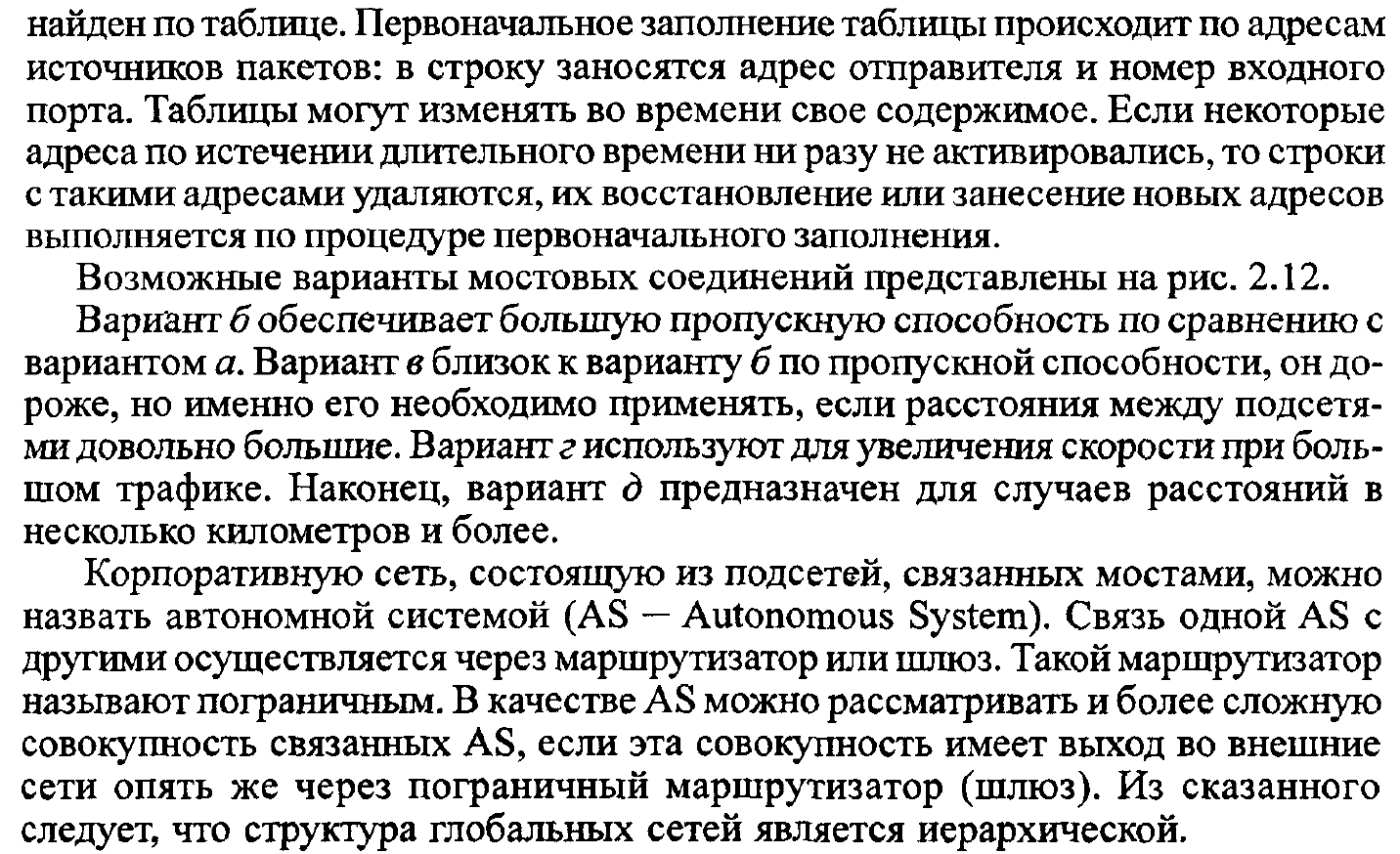
Кольцевая (ring), при которой компьютеры объединены в замкнутое однонаправленное кольцо

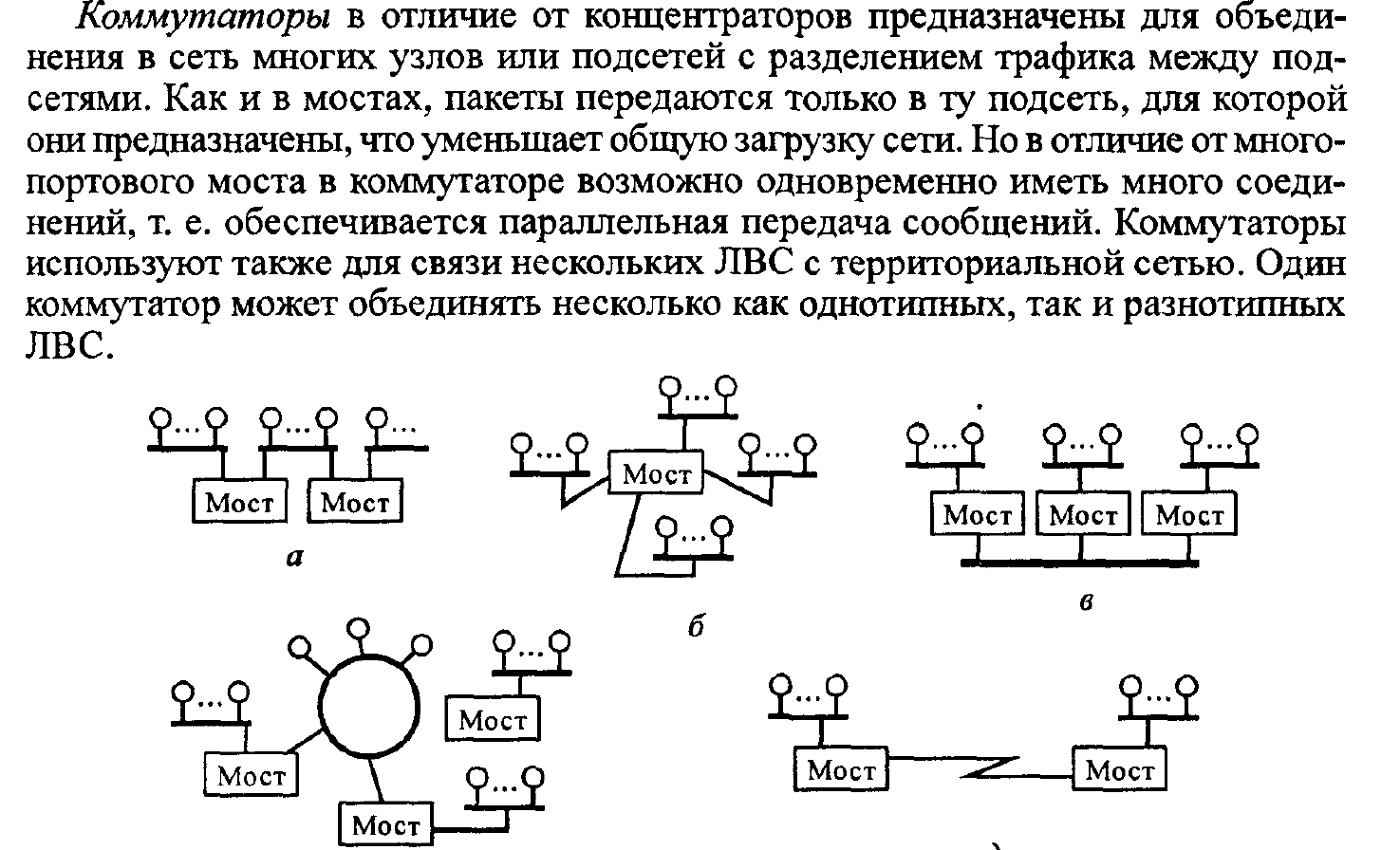


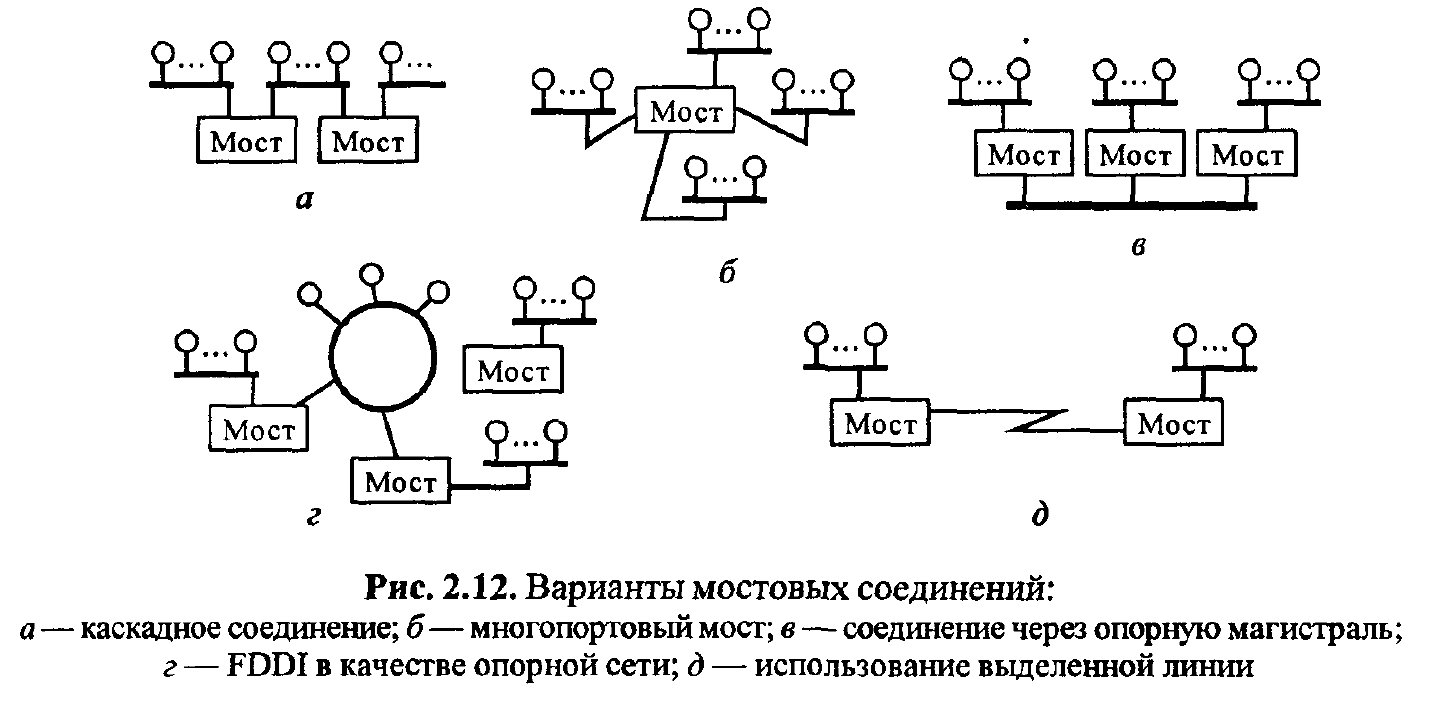
1. Основные типы сетевых устройств.

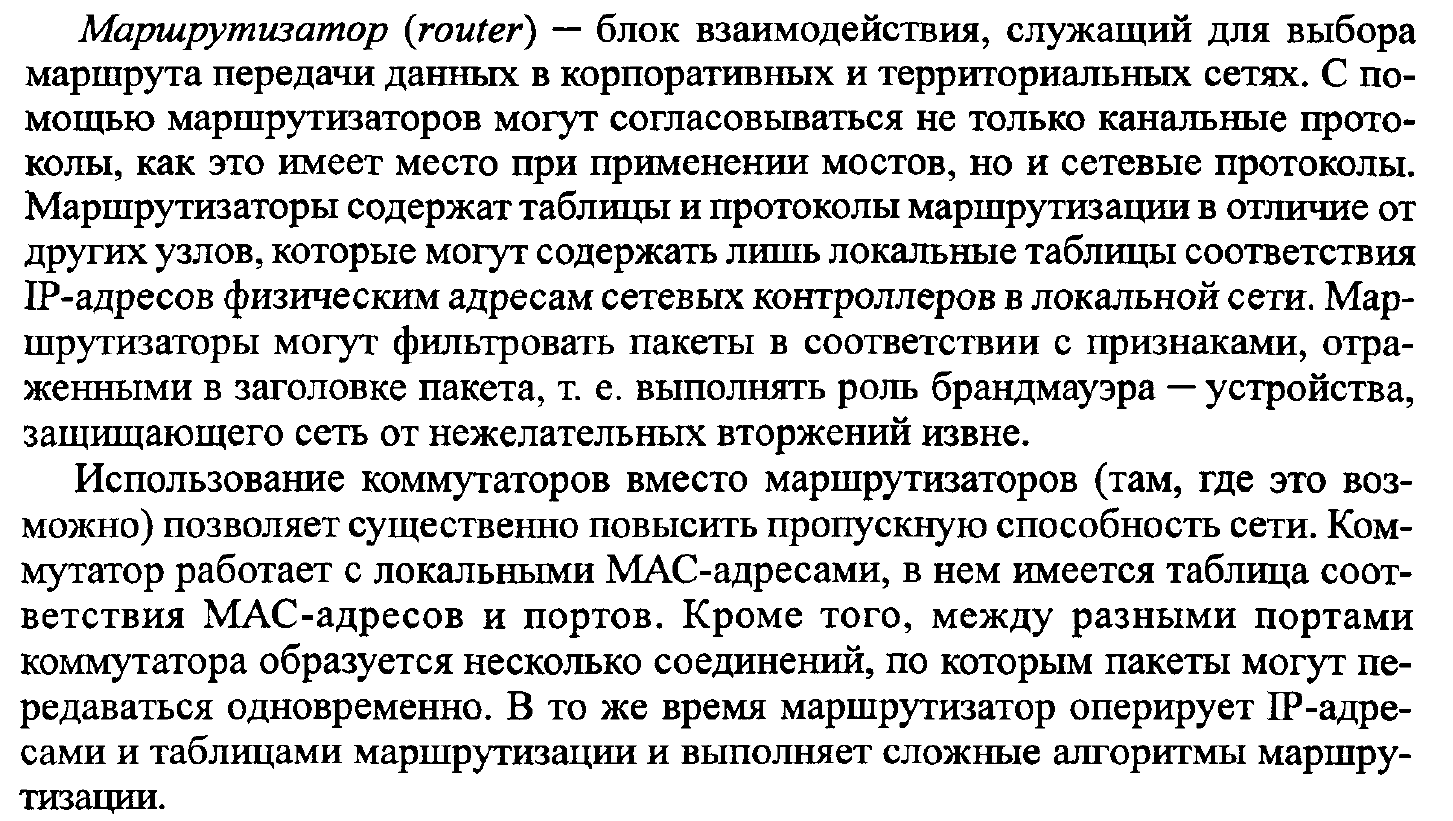


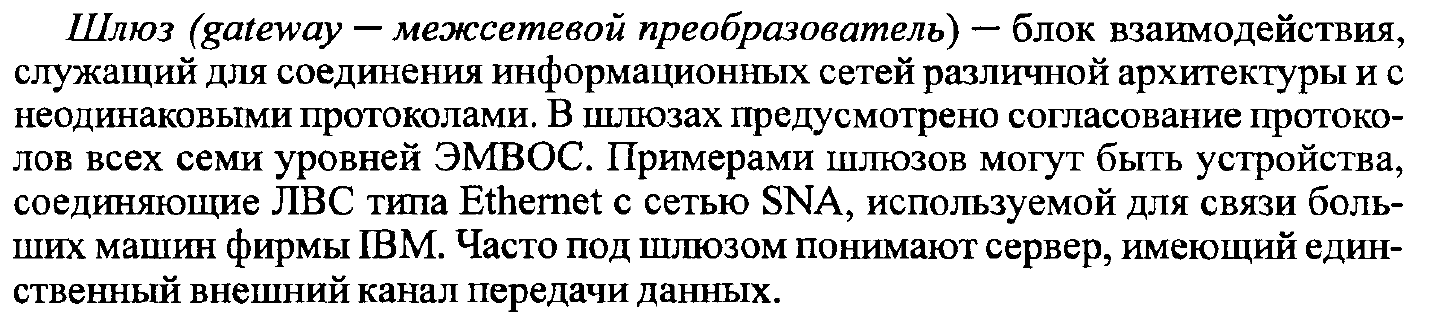




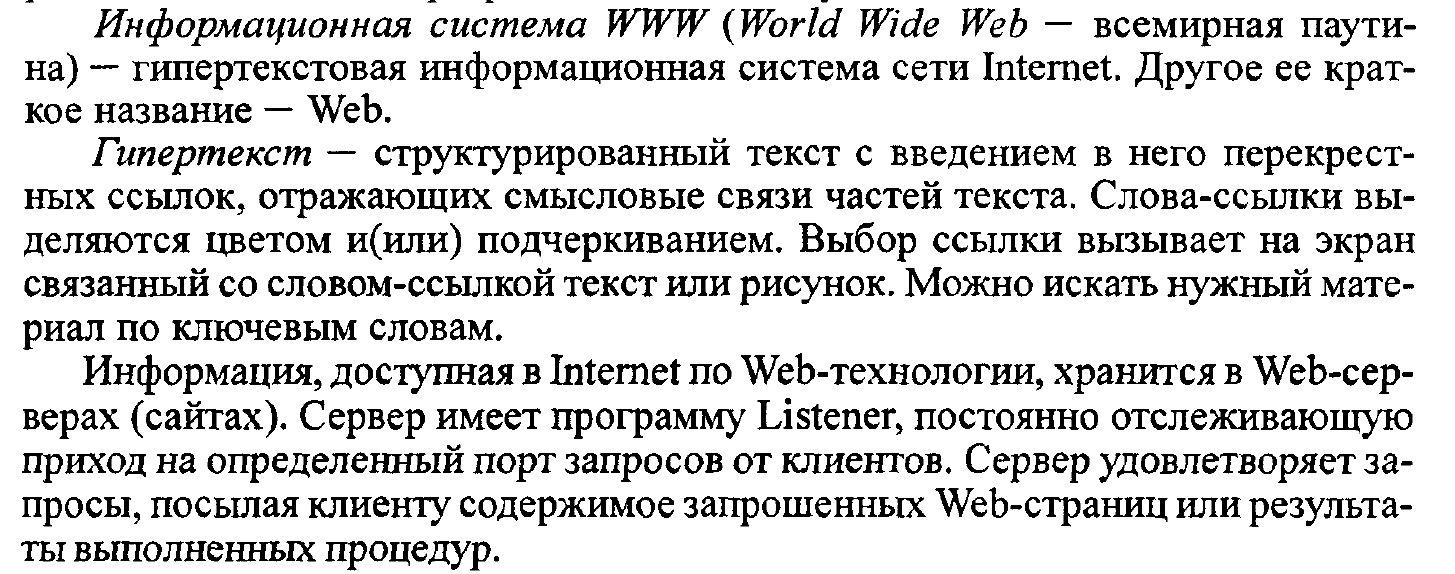


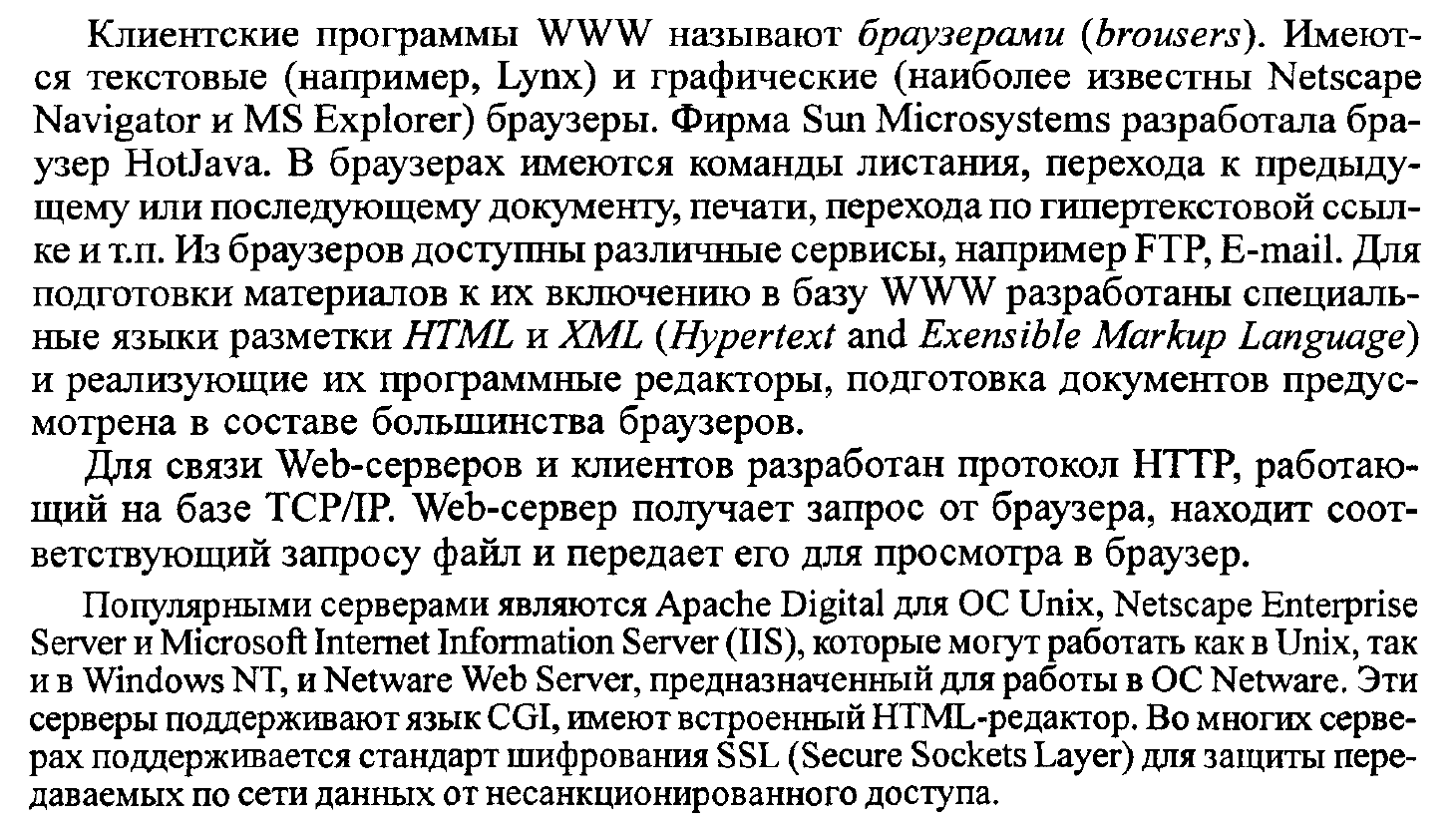






1. Основные технологии WWW.





1. Технология PHP. Встраивание PHP в HTML.

PHP — язык, работающий на стороне сервера. Это может показаться немного сложным для понимания, особенно если до сих пор вы создавали сайты исключительно с помощью клиентских языков, таких как HTML, CSS и JavaScript. Серверные языки во многом похожи на JavaScript, они точно так же позволяют встраивать небольшие программы (скрипты) в HTML-код веб-страницы. По сравнению с HTML такой подход дает больше возможностей для контролирования содержимого окна браузера. Главное отличие между JavaScript и PHPпроявляется на этапе загрузки веб-страницы, когда эти встроенные программы запускаются. Код, который написан на JavaScript и подобных ему языках, работающих на стороне клиента, считывается браузером и выполняется уже после загрузки вебстраницы с сервера. В то время как код, написанный с помощью серверных языков вроде PHP, выполняется веб-сервером до того, как страница будет отправлена в браузер. Клиентские языки позволяют управлять поведением страницы, которая уже отображается в браузере, серверные — генерируют модифицированные страницы налету, до того как они будут отправлены на сторону клиента. Вместо встроенного PHP-кода веб-сервер вставляет в страницу результаты его выполнения. Получив эту страницу, браузеры видят стандартный HTML-код — отсюда термин серверный язык. Давайте вернемся к примеру today.php, о котором шла речь в главе 1 (chaptеr3/today.php).

<!DOCTYPE html>

<html lang="en''>

<head>

<meta charset="utf-8">

<title>Сегодняшняя дата</title>

</head>

<body>

<р>Сегодняшняя дата (согласно данному веб-серверу):

**<?php**

**echo date('l, F jS Y.');**

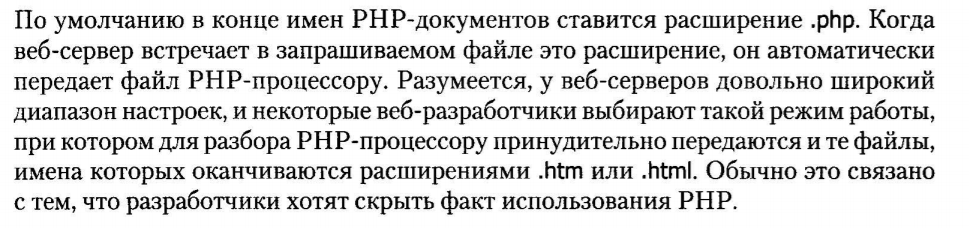
**?>**

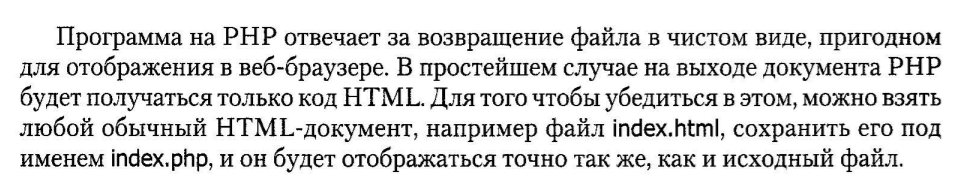
</p>

</body>

</html>

Значительная часть данного примера написана на языке HTML, и только участок между <?php и ?> содержит PHP-код. Выражение <?php обозначает начало встроенного PHP-скрипта, а ?> — его окончание. Веб-сервер интерпретирует все, что находится между этими разделителями, преобразует результат в обычный HTML-код и отправит полученную веб-страницу в браузер, который посылал запрос.





Интерпретатор РНР просмотрел сценарий и заменил его строками вывода. Следовательно, из среды РНР можно создать чистый код HTML, пригодный для просмотра в любом браузере — иначе говоря, применяемый пользователем браузер не обязательно должен понимать РНР.

Это служит хорошей иллюстрацией концепции создания серверных сценариев. PHP-код интерпретируется и выполняется на Web-сервере, в отличие от JavaScript и других технологий клиентской стороны, которые интерпретируются и выполняются в среде Web-браузера на компьютере пользователя.

1. Каскадные таблицы стилей CSS.

Каскадной стилевой таблицей CSS (Cascading Style Sheets) называется набор правил – параметров форматирования, которые применяются к элементам документа, чтобы изменить их внешний вид. Стилизация страницы обычно применяется после разработки структуры и наполнения ее содержанием средствами HTML.

Параметры форматирования объединяются в правила вида:

селектор1, селектор2, ... {

свойство1: значение1;

свойство2: значение2; ...

}

Селекторы определяют теги, к которым применяется данное правило. Каждый селектор может применяться к одному или нескольким тегам веб-страницы.

При загрузке веб-страницы сначала загружаются внешние CSS файлы и к тегам веб-страницы применяются все определенные в них стили, затем загружаются страничные стили, наконец, последними применяются локальные стили. При этом более специфичные правила переопределяют менее специфичные. Таким образом, все правила, описанные в каскадных таблицах стилей, применяются последовательно ко всем встречаемым тегам, дополняя и переопределяя их свойства. Входящие в правила селекторы могут состоять из имен тегов, классов, идентификаторов, псевдоклассов и псевдоэлементов.

Имена тегов применяют, когда необходимо определить стиль для отдельных тегов веб-страницы. Для этого используется следующий синтаксис правил:

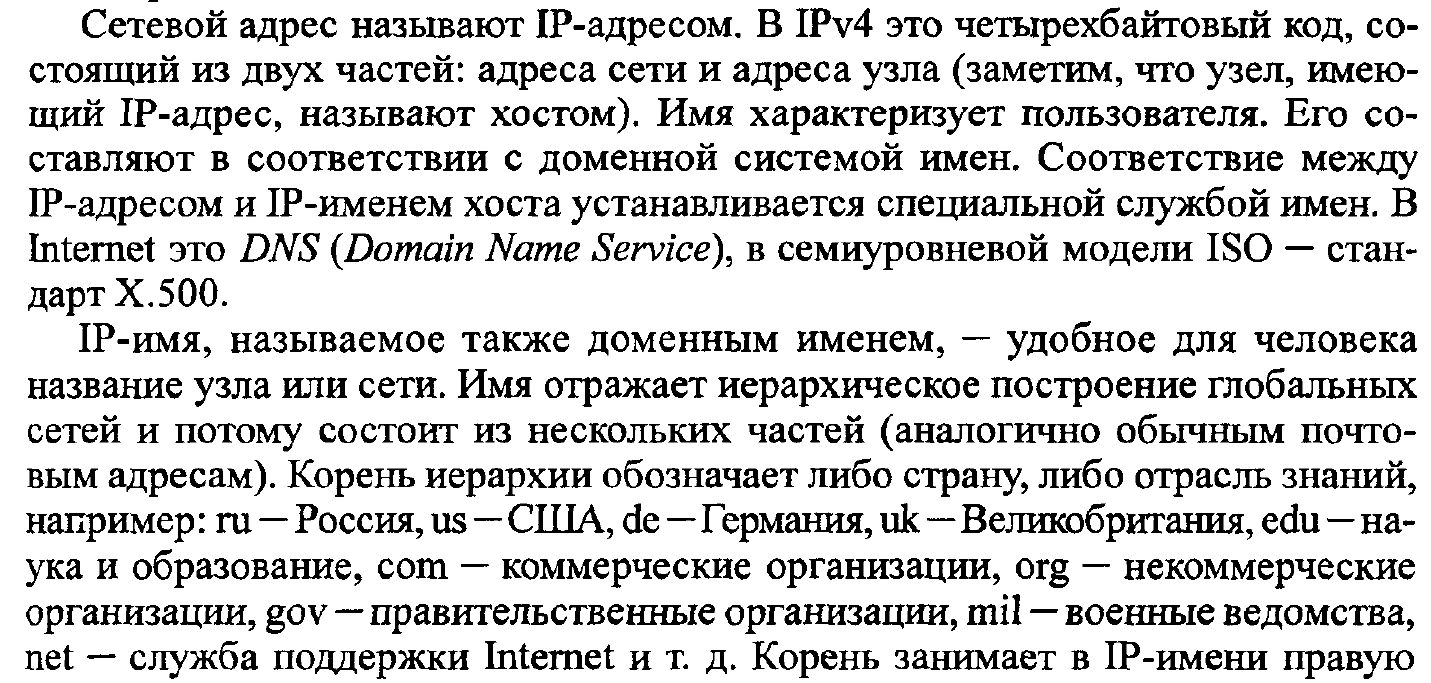
тег { свойство: значение; ... }

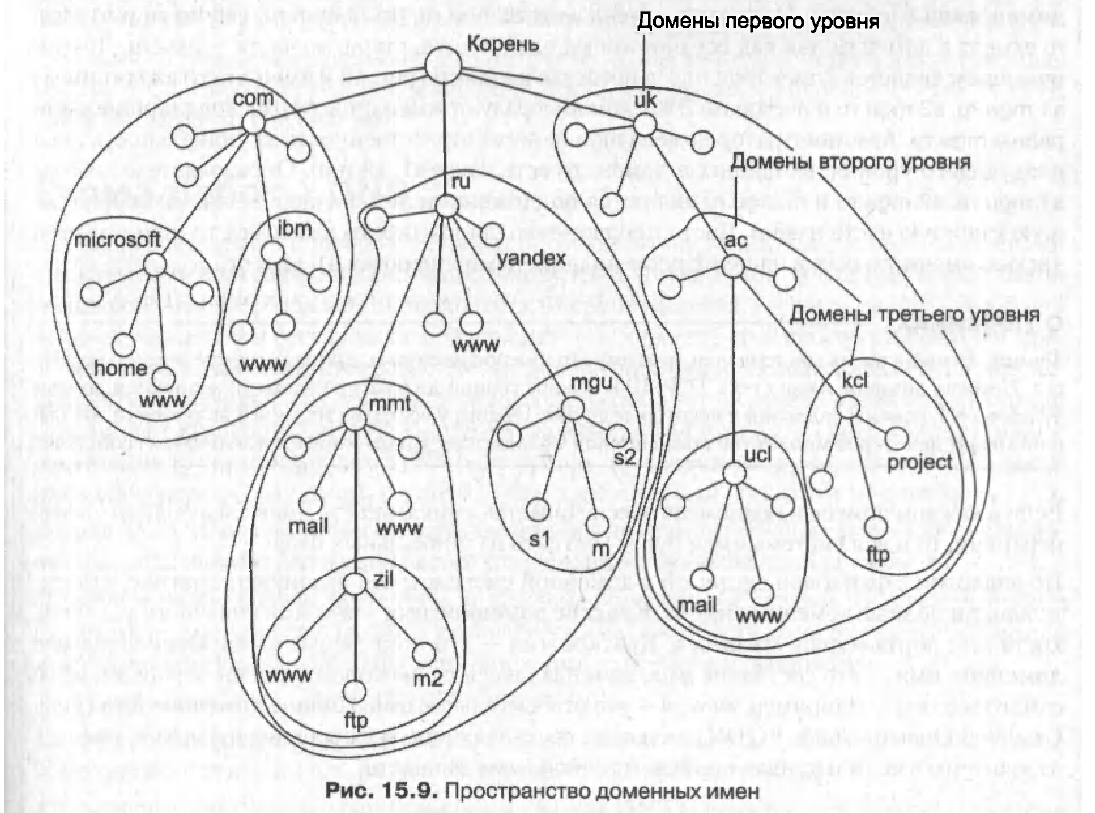
Пример описания CSS стиля:

div { text-align: center; } /\* Для всех тегов div \*/

При этом структура веб-страницы остается неизменной, а все содержание тегов div будет выровнено по центру:

1. Система доменных имен DNS.





1. Графика в web-дизайне.

Графика в HTML

Изображения

Люди устроены так, что первым делом обращают внимание на рисунки, а не на текст. Поэтому именно графическое оформление web - страницы во многом определяет, уделит ли читатель ей время или пройдет мимо. Недаром говорят, что лучше один раз увидеть, чем сто раз услышать.

Существует огромное множество графических форматов, однако для публикаций в Internet используются, в основном, только три - GIF ( Graphics Interchange Format), PNG (Portable Network Graphics) и JPEG ( Joint Potographic Expert Group). WEB - броузер несложно настроить и на другие форматы, такие как TIFF (Tagged Image File Format) и EPS (Encapsulate PostScript), однако практически все web - документы содержат только встроенные изображения в формате GIF, PNG и указатели на большие графические файлы высокого разрешения в формате JPEG .

Формат **GIF**

GIF (Graphics Interchange Format) — формат графических файлов, широко применяемый при создании сайтов. GIF использует 8-битовый цвет и эффективно сжимает сплошные цветные

области, при этом сохраняя детали изображения.

Особенности

Количество цветов в изображении может быть от 2 до 256, но это могут быть любые цвета из 24-битной палитры.

Файл в формате GIF может содержать прозрачные участки. Если используется отличный от белого цвета фон, он будет проглядывать сквозь «дыры» в изображении.

Поддерживает покадровую смену изображений, что делает формат популярным для создания баннеров и простой анимации.

Использует свободный от потерь метод сжатия

Область применения: текст, логотипы, иллюстрации с четкими краями, анимированные рисунки, изображения с прозрачными участками, баннеры.

Формат **JPEG**

JPEG (Joint Photographic Experts Group) — популярный формат графических файлов, широко применяемый при создании сайтов и для хранения изображений. JPEG поддерживает 24-битовый цвет и сохраняет яркость и оттенки цветов в фотографиях неизменными. Данный формат называют сжатием с потерями, поскольку алгоритм JPEG выборочно отвергает данные. Метод сжатия может внести искажения в рисунок, особенно содержащий текст, мелкие детали или четкие края. Формат JPEG не поддерживает прозрачность. Когда вы сохраняете фотографию в этом формате, прозрачные пиксели заполняются определенным цветом.

Особенности

Количество цветов в изображении — около 16 миллионов, что вполне достаточно для сохранения фотографического качества изображения.

Основная характеристика формата — качество, позволяющее управлять конечным размером файла.

Поддерживает технологию, так называемый прогрессивный JPEG, в котором версия рисунка с низким разрешением появляется в окне просмотра до полной загрузки самого изображения.

Область применения: используется преимущественно для фотографий. Не очень подходит для рисунков содержащих прозрачные участки, мелкие детали или текст.

Формат **PNG-24**

PNG-24 — формат, аналогичный PNG-8, но использующий 24-битную палитру цвета Подобно формату JPEG, сохраняет яркость и оттенки цветов в фотографиях. Подобно GIF и формату PNG-8, сохраняет детали изображения, как, например, в линейных рисунках, логотипах, или иллюстрациях

Особенности

Использует примерно 16,7 млн. цветов в файле, из-за чего этот формат применяется для полноцветных изображений.

Поддерживает многоуровневую прозрачность, это позволяет создавать плавный переход от прозрачной области изображения к цветной, так называемый градиент.

Из-за того, что используемый алгоритм сжатия сохраняет все цвета и пикселы в

изображении неизменными, если сравнивать с другими форматами, то у PNG-24

конечный объем графического файла получается наибольшим.

Область применения: фотографии, рисунки, содержащие прозрачные и полупрозрачные участки, рисунки с большим количеством цветов и четкими краями изображений.

**Добавление рисунка**

Для встраивания графических файлов используется тег <img>.

Синтаксис: <img src="URL" alt="альтернативный текст">

Пример: <img src="/pictures/mypicture.jpg" alt="Моя рисунок">