МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Невинномысский технологический институт (филиал)

Методические указания по выполнению лабораторных работ по дисциплине «Передача данных в системах управления»

Направление подготовки 15.03.04 Автоматизация технологических процессов и производств
Квалификация выпускника – бакалавр

Невинномысск 2021

Методические указания к выполнению лабораторных работ и по освоению дисциплины «Передача данных в системах управления» предназначены для бакалавров всех форм бучения направления 15.03.04 Автоматизация технологических процессов и производств.

Методические указания составлены на основании федерального государственного образовательного стандарта высшего образования по направлению подготовки 15.03.04 Автоматизация технологических процессов и производств.

Содержание

Введение

Передача данных в системах управления представляет собой дисциплину в которой рассматриваются методы передачи данных и смежные вопросы, рассматривающие направление статистической теории связи, основы которой были заложены классическими трудами Н. Винера, А.Н. Колмогорова, В.А. Котельникова и К. Шеннона.

Круг проблем, составляющих основное содержание теории передачи данных, представляют исследования методов кодирования для экономного представления сообщений различных источников и для надежной передачи сообщений по каналам связи с помехой.

Так в основе теории информации лежит статистическое описание (статистические модели) источников сообщений и каналов связи, а также измерение количества информации, передаваемое по каналам от источников. При этом количество информации определяется только вероятностными свойствами сообщений.

Все задачи, стоящие перед дисциплиной, направлены на решение двух основных проблем. Первая основная проблема – проблема эффективности передачи информации. Проблема эта состоит в том, чтобы передать наибольшее количество сообщений наиболее экономным способом.

Вторая основная проблема – проблема надежности передачи информации. Вследствие влияния помех переданная информация искажается. Теория указывает общие пути повышения достоверности передачи информации.

Требования эффективности и надежности в известной степени противоречивы. Теория помогает отысканию приемлемого компромисса.

Всякий раз, решая подобные проблемы, пытаются найти предельные значения количества двоичных символов, скорость передачи или величины ошибок, а также пытаются найти способ обработки сообщений (кодирование и декодирование), который позволяет достичь указанных пределов.

В данной работе рассмотрены практические вопросы информационной метрики, оптимального кодирования, построение корректирующих кодов и др.

Лабораторная работа №1 Способы хранения, обработки и передачи информации

Цель работы: познакомиться со способами хранения, обработки и передачи информации.

Методические указания.

Сбор и регистрация данных.

Сбор информации – это процесс целенаправленного извлечения и анализа информации о предметной области, в роли которой может выступать тот или иной процесс, объект и т.д. Цель сбора - обеспечение готовности информации к дальнейшему продвижению в информационном процессе.

Операции сбора и регистрации данных осуществляются с помощью различных средств. Различают :

- механизированный;
- автоматизированный;
- автоматический способы сбора и регистрации данных.
    - Механизированный - сбор и регистрация информации осуществляется непосредственно человеком с использованием простейших приборов (весы, счетчики, мерная тара, приборы учета времени и т.д.).
    - Автоматизированный - использование машиночитаемых документов, универсальных систем сбора и регистрации, обеспечивающих совмещение операций формирования первичных документов и получения машинных носителей.
    - Автоматический - используется в основном при обработке данных в режиме реального времени. (Информация с датчиков, учитывающих ход производства - выпуск продукции, затраты сырья, простои оборудования и т.д. - поступает непосредственно в ЭВМ).

Передача данных.

Передача данных – это перенос данных в виде двоичных сигналов из одного пункта в другой средствами электросвязи, как правило, для последующей обработки средствами вычислительной техники.

Технические средства передачи данных включают:

- аппаратуру передачи данных (АПД), которая соединяет средства обработки и подготовки данных с телеграфными, телефонными и широкополосными каналами связи;
- устройства сопряжения ЭВМ с АПД, которые управляют обменом информации - мультиплексоры передачи данных.
- запись и передача информации по каналам связи в ЭВМ имеет следующие преимущества:
- упрощает процесс формирования и контроля информации;
- соблюдается принцип однократной регистрации информации в первичном документе и машинном носителе;
- обеспечивается высокая достоверность информации, поступающей в ЭВМ.

Существует дистанционная передача данных, которая представляет собой передачу данных в виде электрических сигналов, которые могут быть непрерывными во времени и дискретными, т.е. носить прерывный во времени характер. Наиболее широко используются телеграфные и телефонные каналы связи. Электрические сигналы, передаваемые по телеграфному каналу связи являются дискретными, а по телефонному - непрерывными.

В зависимости от направлений, по которым пересылается информация, различают каналы связи:

- симплексный (передача идет только в одном направлении);
- полудуплексный (в каждый момент времени производится либо передача, либо прием информации);
- дуплексный (передача и прием информации осуществляются одновременно в двух встречных направлениях).

Обработка данных.

Технология обработки данных применяется на уровне операционной (исполнительской) деятельности персонала невысокой квалификации в целях автоматизации некоторых рутинных постоянно повторяющихся операций управленческою труда. Поэтому внедрение информационных технологий и систем на этом уровне существенно повысит производительность труда персонала, освободит его от рутинных операций, возможно, даже приведет к необходимости сокращения численности работников.

Технологический процесс обработки информации с использованием ЭВМ включает в себя следующие операции:

- прием и комплектовка документов (проверка полноты и качества их заполнения, комплектовки и т.д.);
- подготовка и контроль;
- ввод данных в ЭВМ;
- сортировка (если в этом есть необходимость);
- обработка данных;
- получение ответов на всевозможные текущие запросы и их оформление.

Вывод данных.

Заключительным этапом после сбора, регистрации, передачи и обработки данных является их вывод в том или ином формате, как то графическом, табличном или текстовом виде. Непосредственно сам вывод данных может осуществляться через электронный устройства. Таковыми являются:

- Мониторы.
- Принтеры.
- Плоттер.
- Графопостроитель

Задания

1. Набрать в одном из текстовых редакторов текст из 10 предложений на тему «Моя профессия».
2. Вставить в набранный текст рисунок.
3. Сохранить текст на каких-либо носителях.
4. Создать свою электронную почту.
5. Отправить, набранную информацию по электронной почте.
6. Получить информацию по электронной почте.
7. Изменить полученный текст, введя диаграмму.
8. Сохранить текст.

Контрольные вопросы:

1. Как происходит сбор и регистрация данных?
2. Как происходит передача данных?
3. Из каких технологических процессов состоит процесс обработки информации?
4. Как осуществляется вывод данных?

Лабораторная работа №2 Применение систем счисления в задачах передачи данных в системах управления

Представление числовой информации с помощью систем счисления.

Цель работы: познакомиться с алгоритмами представления десятичных целых, отрицательных и вещественных чисел в памяти ЭВМ.

Методические указания.

Все числовые данные хранятся в машине в двоичном виде, т.е. в виде последовательности нулей и единиц, однако формы хранения целых и действительных чисел различны.

Для представления чисел в памяти ПК используются два формата:

-формат с фиксированной точкой (запятой) целые числа;

-формат с плавающей точкой (запятой) вещественные числа.

Представление целых чисел

Множество **целых чисел**, представленных в ЭВМ, ограничено. Диапазон значений зависит от размера ячеек памяти, используемых для их хранения.

Для целых чисел существуют два представления:

-беззнаковое;

-со знаком.

В К-разрядной ячейке может храниться $2^к$ различных значений целых чисел.

Диапазон значений целых беззнаковых чисел (только положительные):

от 0 до $2^к - 1$

для 16-разрядной ячейки от 0 до 65535

для 8-разрядной ячейки от 0 до 255

Диапазон значений целых чисел со знаком (и отрицательные, и положительные в равном количестве):

от $-2^{к-1}$ до $2^{к-1}-1$

для 16-разрядной ячейки от -32768 до 32767

для 8-разрядной ячейки от -128 до 127

Чтобы получить внутреннее представление **целого положительного числа N**, хранящегося в К-разрядной ячейке, необходимо:

1. перевести число N в двоичную систему счисления;
2. полученный результат дополнить слева незначащими нулями до К разрядов.

Пример:

Получить внутреннее представление целого числа 1607 в 2-х байтовой ячейке.

Решение:

$N=1607=11001000111_2$.

Внутреннее представление этого числа будет: 0000 0110 0100 0111. Шестнадцатеричная форма внутреннего представления числа: 0647.

Для представления целого отрицательного числа используется **дополнительный код**.

Дополнительным кодом двоичного числа X в N-разрядной ячейке является число, дополняющее его до значения 2^N.

Получение дополнительного кода:

1. получить внутреннее представление положительного числа N (прямой код);
2. получить обратный код этого числа заменой 0 на 1 или 1 на 0 (обратный код);
3. к полученному числу прибавить 1.

Положительное число в прямом, обратном и дополнительном кодах не меняют свое изображение.

Использование дополнительного кода позволяет заменить операцию вычитания на операцию сложения.

A-B=A+(-B).

Процессору достаточно уметь лишь складывать числа.

Старший, К-й разряд во внутреннем представлении любого положительного числа равен 0, отрицательного числа равен 1. Поэтому этот разряд называется **знаковым разрядом**

Пример:

Получить внутреннее представление целого отрицательного числа - 1607.

Решение:

1. Внутреннее представление положительного числа: 000 0110 0100 0111;
2. Обратный код: 1111 1001 1011 1000;
3. Дополнительный код: 1111 1001 1011 1001 - внутреннее двоичное представление числа.

16-ричная форма: F9B9.

Представление вещественных чисел

Вещественные числа представляются в ПК в форме с плавающей точкой.

Этот формат использует представление вещественного числа R в виде произведения мантиссы m на основание системы счисления p в некоторой целой степени n которую называют порядком:

$R=m*p^n$

Представление числа в форме с плавающей точкой неоднозначно.

Например: $25.324=25324*10^1=0.0025324*10^4=2532.4*10^{-2}$

В ЭВМ используют **нормализованное** представление числа в форме с плавающей точкой. Мантисса в нормализованном представлении должна удовлетворять условию: $0.1_p \leqslant m<1_p$

Иначе говоря, мантисса меньше 1 и первая значащая цифра - не 0.

В памяти компьютера мантисса представляется как целое число, содержащее только значащие цифры (0 целых и запятая не хранится). Следовательно, внутреннее представление вещественного числа сводиться к представлению пары целых чисел: мантиссы и порядка.

Например: 4-х байтовая ячейка памяти. В ячейке должна содержаться следующая информация о числе:

- знак числа;
- порядок;
- значащие цифры мантиссы.

±	МАН	ТИ	ССА
1-й байт	2-й байт	3-й байт	4-й байт

В старшем бите 1-го байта хранятся знак числа: 0 обозначает плюс, 1 - минус.

Оставшиеся 7 бит 1-го байта содержат машинный порядок. В следующих трех байтах хранятся значащие цифры мантиссы (24 разряда).

В семи двоичных разрядах помещаются двоичные числа в диапазоне от 0000000 до 1111111. Значит, машинный порядок изменяется в диапазоне от 0 до 127 (в десятичной системе счисления). Всего 128 значений. Порядок, очевидно, может быть как положительным так и отрицательным. Разумно эти 128 значений разделить поровну между положительным и отрицательным значениями порядка: от -64 до 63.

Машинный порядок смещен относительно математического и имеет только положительные значения. Смещение выбирается так, чтобы минимальному математическому значению порядка соответствовал нуль.

Связь между машинным порядком (Mp) и математическим (p) в рассматриваемом случае выражается формулой:

$$Mp = p + 64$$

Полученная формула записана в десятичной системе. В двоичной системе формула имеет вид: $Mp_2=p_2+1000000_2$

Для записи внутреннего представления вещественного числа необходимо:

1) перевести модуль данного числа в двоичную систему счисления с 24 значащими цифрами;

2) нормализовать двоичное число;

3) найти машинный порядок в двоичной системе счисления;

4) учитывая знак числа, выписать его представление в 4-х байтовом машинном слове.

Пример

Записать внутреннее представление числа 250,1875 в форме с плавающей точкой.

Решение:

1) Приведем его в двоичную систему счисления с 24 значащими цифрами: $250.1875_{10}=11111010, 001100000000000000_2$.

2) Запишем в форме нормализованного двоичного числа с плавающей точкой: $0,111110100011000000000000*10_2^{1000}$. Здесь мантисса, основание системы счисления ($2_{10}=10_2$) и порядок ($8_{10}=1000_2$) записаны в двоичной системе.

3) Вычислим машинный порядок в двоичной системе счисления: $Мр_2$= 1000 + 100 0000 =100 1000.

4) Запишем представление числа в 4-х байтовой ячейке памяти с учетом знака числа:

	1001000	11111010		00110000	00000000	
1	24	23		0		

Шестнадцатеричная форма: 48FA3000.

Пример.

По шестнадцатеричной форме внутреннего представления числа в форме с плавающей точкой C9811000 восстановить само число.

Решение: 1) Перейдем к двоичному представлению числа в 4-х байтовой ячейке, заменив каждую шестнадцатеричную цифру 4-мя двоичными цифрами:

1100 1001 1000 0001 0001 0000 0000 0000

1	1001001	10000001		00010000	00000000	
31		23		0		

2) Заметим, что получен код отрицательного числа, поскольку в старшем разряде с номером 31 записана 1. Получим порядок числа: $p=1001001_2 -1000000_2=1001_2=9_{10}$.

3) Запишем в форме нормализованного двоичного числа с плавающей точкой с учетом знака числа:

$-0,10000001000100000000000 *2^{1001}$

4) Число в двоичной системе счисления имеет вид: -100000010.001_2.

5) Переведем число в десятичную систему счисления:

$-100000010.001_2= -(1*2^8+1*2^1+1*2^{-3})= -258.125_{10}$

Задание для решений №1

1) Получить двоичную форму внутреннего представления целого числа в 2-х байтовой ячейке.

2) Получить шестнадцатеричную форму внутреннего представления целого числа 2-х байтовой ячейке.

3) По шестнадцатеричной форме внутреннего представления целого числа в 2-х байтовой ячейке восстановить само число.

	Номера заданий		
№ Варианта	**1**	**2**	**3**
1	1450	-1450	F67D

2	1341	-1341	F7AA
3	1983	-1983	F6D7
4	1305	-1305	F700
5	1984	-1984	F7CB
6	1453	-1453	F967
7	1833	-1833	F83F
8	2331	-2331	F6E5
9	1985	-1985	F8D7
10	1689	-1689	FA53
11	2101	-2101	F840
12	2304	-2304	FAE7
13	2345	-2345	F841
14	2134	-2134	FAC3
15	2435	-2435	FA56

Задание для решений №2

1) Получить шестнадцатеричную форму внутреннего представления числа в формате с плавающей точкой в 4-х байтовой ячейке.

2) По шестнадцатеричной форме внутреннего представления вещественного числа в 4-х байтовой ячейке восстановить само число.

	Номера заданий	
№ Варианта	**1**	**2**
1	26.28125	C5DB0000
2	-29.625	45D14000
3	91.8125	C5ED0000
4	-27.375	47B7A000
5	139.375	C5D14000
6	-26.28125	488B6000
7	27.375	C7B7A000
8	-33.75	45DB0000
9	29.265	C88B6000
10	-139.375	45ED0000
11	333.75	C6870000
12	-333.75	46870000
13	224.25	C9A6E000
14	-91.8125	49A6E000
15	33.75	48E04000

Контрольные вопросы:

1. Как представляют целые числа?
2. Что используется для представления целого отрицательного числа?
3. Какие числа не меняют изображения?

4. Какой разряд называется знаковым разрядом?
5. Что называется нормализованным представление числа в форме с плавающей точкой?

Применение правил недесятичной арифметики

Цель работы: познакомиться с правилами недесятичной арифметики. Выработать навыки перевода чисел.

Методические указания.

Перевод чисел из одной системы счисления в другую.

Перевод чисел в десятичную систему осуществляется путем составления степенного ряда с основанием той системы, из которой число переводится. Затем подсчитывается значение суммы.

Двоичная арифметика.

При сложении двоичных чисел в каждом разряде производится сложение цифр слагаемых и переноса из соседнего младшего разряда, если он имеется. При этом необходимо учитывать, что 1+1 дают нуль в данном разряде и единицу переноса в следующий.

Восьмеричная система счисления. Используется восемь цифр: 0, 1, 2, 3, 4, 5, 6, 7.

Употребляется в ЭВМ как вспомогательная для записи информации в сокращенном виде. Для представления одной цифры восьмеричной системы используется три двоичных разряда (триада) (Таблица 1).

Шестнадцатеричная система счисления. Для изображения чисел употребляются 16 цифр. Первые десять цифр этой системы обозначаются цифрами от 0 до 9, а старшие шесть цифр латинскими

буквами: 10=A,
11=B,
12=C,
13=D,
14=E,
15=F.

Шестнадцатеричная система используется для записи информации в сокращенном виде. Для представления одной цифры шестнадцатеричной системы счисления используется четыре двоичных разряда (тетрада)

Задание №1

Переведите данное число из десятичной системы счисления в двоичную, восьмеричную и шестнадцатеричную системы счисления.

772(10).
71(10).
284.375(10).
876.5(10).
281.86(10).

Задание № 2.

Переведите данное число в десятичную систему счисления.

1000001111(2).
1010000110(2).
101100110.011011(2).
100100110.101011(2).
1022.2.
53.9(16).

Задание № 3.
Сложите числа.
1100111(2)+1010111000(2).
1101111010(2)+1000111100(2).
1111101110.01(2)+1110001.011(2)
153.3(8)+1347.2(8).
e0.2(16)+1e0.4(16).

Задание № 4.
Выполните вычитание.
1010101110(2)-11101001(2).
1000100010(2)-110101110(2).
1010100011.011(2)-1000001010.001(2).
1517.64(8)-1500.30(8).
367.6(16)-4a.c(16).

Задание № 5.
Выполните умножение.
1100110(2)*101111(2).
1272.3(2)*23.14(8).
48.4(16)*5.a(16).

Контрольные вопросы:

1. Какая система счисления называется двоичной?
2. Какая система счисления называется восьмеричной?
3. Какая система счисления называется шестнадцатеричной?
4. Как производится перевод из одной системы счисления в другой?

Перевод из одной системы счисления в другую.

Цель: научиться переводить числа из одной системы счисления в другую.

Методические указания.

Под системой счисления понимается способ представления любого числа с помощью некоторого алфавита символов, называемых цифрами.

Все системы счисления делятся на позиционные и непозиционные.

Непозиционными системами являются такие системы счисления, в которых каждый символ сохраняет свое значение независимо от места его положения в числе. Примером непозиционной системы счисления является римская система. К недостаткам таких систем относятся наличие большого количества знаков и сложность выполнения арифметических операций.

Система счисления называется позиционной, если одна и та же цифра имеет различное значение, определяющееся позицией цифры в последовательности цифр, изображающей число. Это значение меняется в однозначной зависимости от позиции, занимаемой цифрой, по некоторому закону. Примером позиционной системы счисления является десятичная система, используемая в повседневной жизни.

Количество *p* различных цифр, употребляемых в позиционной системе определяет название системы счисления и называется основанием системы счисления *"p"*.

В десятичной системе используются десять цифр: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9; эта система имеет основанием число десять.

Задание 1. Запишите развернутую и краткую формы записи любого числа.

В ЭВМ применяют позиционные системы счисления с недесятичным основанием: двоичную, восьмеричную, шестнадцатеричную. В аппаратной основе ЭВМ лежат двухпозиционные элементы, которые могут находиться только в двух состояниях; одно из них обозначается 0, а другое 1. Поэтому основной системой счисления применяемой в ЭВМ является двоичная система.

Двоичная система счисления. Используется две цифры: 0 и 1.

Восьмеричная система счисления. Используется восемь цифр: 0, 1, 2, 3, 4, 5, 6, 7.

Употребляется в ЭВМ как вспомогательная для записи информации в сокращенном виде. Для представления одной цифры восьмеричной системы используется три двоичных разряда (триада) (Таблица 1).

Шестнадцатеричная система счисления. Для изображения чисел употребляются 16 цифр. Первые десять цифр этой системы обозначаются цифрами от 0 до 9, а старшие шесть цифр латинскими

буквами: 10=A,
11=B,
12=C,
13=D,
14=E,
15=F.

Шестнадцатеричная система используется для записи информации в сокращенном виде. Для представления одной цифры шестнадцатеричной системы счисления используется четыре двоичных разряда (тетрада) (Таблица 1).

Таблица 1. Наиболее важные системы счисления.

Двоичная (Основание 2)	Восьмеричная (Основание 8)		Десятичная (Основание 10)	Шестнадцатиричная (Основание 16)	
		триады			тетрады
0	0	000	0	0	0000
1	1	001	1	1	0001
	2	010	2	2	0010
	3	011	3	3	0011
	4	100	4	4	0100
	5	101	5	5	0101
	6	110	6	6	0110
	7	111	7	7	0111
			8	8	1000
			9	9	1001
				A	1010
				B	1011
				C	1100
				D	1101
				E	1110
				F	1111

Перевод чисел из одной системы счисления в другую.

Перевод чисел в десятичную систему осуществляется путем составления степенного ряда с основанием той системы, из которой число переводится. Затем подсчитывается значение суммы.

Задание 2.

Перевести 10101101.101 из «2» в «16», «8» и «10» с.с.

При одновременном использовании нескольких различных систем счисления основание системы, к которой относится число, указывается в виде нижнего индекса.

Задание 3. Переведите самостоятельно.

а) Перевести 703.048 из «10» в «2», затем в «8» и наконец, в «16»

б) Перевести B2E.416 из «16» в «10», затем в «8».

Перевод целых десятичных чисел в недесятичную систему счисления осуществляется последовательным делением десятичного числа на основание той системы, в которую оно переводится, до тех пор, пока не получится частное меньшее этого основания. Число в новой системе записывается в виде остатков деления, начиная с последнего.

Задание 4.

а) Перевести 18110 из «10» в «2».

б) Перевести 62210 из «8» в «2», затем в «10».

Перевод правильных дробей из десятичной системы счисления в недесятичную.

Для перевода правильной десятичной дроби в другую систему эту дробь надо последовательно умножать на основание той системы, в которую она переводится. При этом умножаются только дробные части. Дробь в новой системе записывается в виде целых частей произведений, начиная с первого.

Задание 5. Перевести 0.312510

Замечание. Конечной десятичной дроби в другой системе счисления может соответствовать бесконечная (иногда периодическая) дробь. В этом случае количество знаков в представлении дроби в новой системе берется в зависимости от требуемой точности.

Задание 6. Перевести 0.6510 из «10» в «2» с.с. Точность 6 знаков.

Для перевода неправильной десятичной дроби в систему счисления с недесятичным основанием необходимо отдельно перевести целую часть и отдельно дробную.

Задание 7.

Перевести 23.12510 из «10» в «2» с.с.

Необходимо отметить, что целые числа остаются целыми, а правильные дроби дробями в любой системе счисления. Для перевода восьмеричного или шестнадцатеричного числа в двоичную форму достаточно заменить каждую цифру этого числа соответствующим трехразрядным двоичным числом (триадой) (Таб. 1) или четырехразрядным двоичным числом (тетрадой) (Таб. 1), при этом отбрасывают ненужные нули в старших и младших разрядах.

Задание 8.

а)Перевести 305.47 из «8» в «10» с.с.

б)Перевести 7B2.E16 из «16» в «10».

Для перехода от двоичной к восьмеричной (шестнадцатеричной) системе поступают следующим образом: двигаясь от точки влево и вправо, разбивают двоичное число на группы по три (четыре) разряда, дополняя при необходимости нулями крайние левую и правую группы. Затем триаду (тетраду) заменяют соответствующей восьмеричной (шестнадцатеричной) цифрой.

Двоичная арифметика.

При сложении двоичных чисел в каждом разряде производится сложение цифр слагаемых и переноса из соседнего младшего разряда, если он имеется. При этом необходимо учитывать, что 1+1 дают нуль в данном разряде и единицу переноса в следующий.

Задание 11. Выполнить сложение двоичных чисел:

а) X=1101, Y=101;

б) X=1101, Y=101, Z=111;

При вычитании двоичных чисел в данном разряде при необходимости занимается 1 из старшего разряда. Эта занимаемая 1 равна двум 1 данного разряда.

Задание 12. Заданы двоичные числа X=10010 и Y=101. Вычислить X-Y.

Умножение двоичных чисел производится по тем же правилам, что и для десятичных с помощью таблиц двоичного умножения и сложения.

Самостоятельная работа.

Выполнить перевод числа в соответствии с вариантом.

1. Перевести десятичное число A=121 в двоичную систему счисления.
2. Перевести двоичное число A=10001010111,01 в десятичную систему счисления.
3. Перевести десятичное число A=135,656 в двоичную систему счисления с

точностью до пяти знаков запятой.
4. Перевести двоичное число A=10111011 в десятичную систему счисления методом деления на основание.
5. Перевести восьмеричное число A=345,766 в двоичную систему счисления.
6. Записать десятичное число A=79,346 в двоичнодесятичной форме.
7. Перевести десятичную дробь 64
A = 63 9 в двоичную систему счисления.
8. Перевести десятичное число A=326 в троичную систему счисления.
9. Перевести десятичную дробь 40
A = 63 5 в двоичную систему счисления.
10. Перевести десятичное число A=15,647 в двоичную систему счисления.
11. 12. Перевести десятичную дробь A=0,625 в двоичную систему счисления.
13. Перевести двоичную дробь A=0,1101 в десятичную систему счисления.
14. Перевести десятичное число A=113 в двоичную систему счисления.
15. Перевести двоичное число A=11001,01 в десятичную систему счисления.
16. Перевести десятичное число A=96 в троичную систему счисления.

Контрольные вопросы:

1. Как переводят правильную дробь из десятичной системы счисления в недесятичную?
2. Как осуществляется перевод из восьмеричной в двоичную?
3. Как осуществляется перевод в 64 систему счисления?
4. Как осуществляется перевод в 3 систему счисления?

Как осуществляется перевод в 5 систему счисления?

Лабораторная работа № 3 Использование формулы Хартли при решении задач на определение количества информации

Цель работы: экспериментальное изучение количественных аспектов информации.

Методические указания.

Понятие количество информации отождествляется с понятием информация. Эти два понятия являются синонимами. Мера информации должна монотонно возрастать с увеличением длительности сообщения (сигнала), которую естественно измерять числом символов в дискретном сообщении и временем передачи в непрерывном случае. Кроме того, на содержание количества информации должны влиять и статистические характеристики, так как сигнал должен рассматриваться как случайный процесс.

При этом наложено ряд ограничений:

1. Рассматриваются только дискретные сообщения.
2. Множество различных сообщений конечно.
3. Символы, составляющие сообщения равновероятны и независимы.

Хартли впервые предложил в качестве меры количества информации принять логарифм числа возможных последовательностей символов.

$I=\log m^k=\log N$ (1)

К.Шеннон попытался снять те ограничения, которые наложил Хартли. На самом деле в рассмотренном выше случае равной вероятности и независимости символов при любом k все возможные сообщения оказываются также равновероятными, вероятность каждого из таких сообщений равна P=1/N. Тогда количество информации можно выразить через вероятности появления сообщений I=–log P.

В силу статистической независимости символов, вероятность сообщения длиной в k символов равна

$$P=\prod_{i=1}^{k} p_i$$

Если i–й символ повторяется в данном сообщении k_i раз, то

$$P=\prod_{i=1}^{m} p_i^{k_i}$$

так как при повторении i символа k_i раз k уменьшается до m. Из теории вероятностей известно, что, при достаточно длинных сообщениях (большое число символов k) $k_i \approx k \cdot p_i$ и тогда вероятность сообщений будет равняться

$$P=\prod_{i=1}^{m} p_i^{kp_i}$$

Тогда окончательно получим

$$I=-\log P=-k\sum_{i=1}^{m} p_i \log p_i \quad (2)$$

Данное выражение называется формулой Шеннона для определения количества информации.

Формула Шеннона для количества информации на отдельный символ сообщения совпадает с энтропией. Тогда количество информации сообщения состоящего из k символов будет равняться $I=k \cdot H$

Количество информации, как мера снятой неопределенности

При передаче сообщений, о какой либо системе происходит уменьшение неопределенности. Если о системе все известно, то нет смысла посылать сообщение. Количество информации измеряют уменьшением энтропии.

Количество информации, приобретаемое при полном выяснении состояния некоторой физической системы, равно энтропии этой системы:

$$I = -\sum_{i=1}^{n} p_i \log p_i$$

Количество информации I – есть осредненное значение логарифма вероятности состояния. Тогда каждое отдельное слагаемое $-\log p_i$ необходимо рассматривать как частную информацию, получаемую от отдельного сообщения, то есть

$$I_i = -\log p_i$$

Избыточность информации

Если бы сообщения передавались с помощью равновероятных букв алфавита и между собой статистически независимых, то энтропия таких сообщений была бы максимальной. На самом деле реальные сообщения строятся из не равновероятных букв алфавита с наличием статистических связей между буквами. Поэтому энтропия реальных сообщений $-H_p$, оказывается много меньше оптимальных сообщений – H_o. Допустим, нужно передать сообщение, содержащее количество информации, равное I. Источнику, обладающему энтропией на букву, равной H_p, придется затратить некоторое число n_p, то есть

$$I = n_p H_p$$

Если энтропия источника была бы H_0, то пришлось бы затратить меньше букв на передачу этого же количества информации

$$n_o = \frac{I}{H_o} < n_p$$

$I = n_0 H_0$

Таким образом, часть букв $n_p – n_o$ являются как бы лишними, избыточными. Мера удлинения реальных сообщений по сравнению с оптимально закодированными и представляет собой избыточность D.

$$D = 1 - \frac{H_p}{H_o} = 1 - \frac{n_o}{n_p} = \frac{n_p - n_o}{n_p} \qquad (3)$$

Но наличие избыточности нельзя рассматривать как признак несовершенства источника сообщений. Наличие избыточности способствует повышению помехоустойчивости сообщений. Высокая избыточность естественных языков обеспечивает надежное общение между людьми.

Частотные характеристики текстовых сообщений

Важными характеристиками текста являются повторяемость букв, пар букв (биграмм) и вообще m-ок (m-грамм), сочетаемость букв друг с другом, чередование гласных и согласных и некоторые другие. Замечательно, что эти характеристики являются достаточно устойчивыми.

Идея состоит в подсчете чисел вхождений каждой n^m возможных m-грамм в достаточно длинных открытых текстах $T=t_1t_2\ldots t_l$, составленных из букв алфавита $\{a_1, a_2, \ldots, a_n\}$. При этом просматриваются подряд идущие m-граммы текста

$t_1t_2\ldots t_m$, $t_2t_3\ldots t_{m+1}$, ..., $t_{i-m+1}t_{l-m+2}\ldots t_l$.

Если $\vartheta(a_{i1}a_{i2}\ldots a_{im})$ – число появлений m-граммы $a_{i1}a_{i2}\ldots a_{im}$ в тексте T, а L общее число подсчитанных m-грамм, то опыт показывает, что при достаточно больших L частоты

$$\frac{\vartheta(a_{i1}a_{i2}\ldots a_{im})}{L}$$

для данной m-граммы мало отличаются друг от друга.

В силу этого, относительную частоту считают приближением вероятности $P(a_{i1}a_{i2}\ldots a_{im})$ появления данной m-граммы в случайно выбранном месте текста (такой подход принят при статистическом определении вероятности).

Для русского языка частоты (в порядке убывания) знаков алфавита, в котором отождествлены Е с Ё, Ь с Ъ, а также имеется знак пробела (-) между словами, приведены в таблице 1.

Таблица 1

-	0.175	О	0.090	Е, Ё	0.072	А	0.062
И	0.062	Т	0.053	Н	0.053	С	0.045
Р	0.040	В	0.038	Л	0.035	К	0.028
М	0.026	Д	0.025	П	0.023	У	0.021
Я	0.018	Ы	0.016	З	0.016	Ь, Ъ	0.014
Б	0.014	Г	0.013	Ч	0.012	Й	0.010
Х	0.009	Ж	0.007	Ю	0.006	Ш	0.006
Ц	0.004	Щ	0.003	Э	0.003	Ф	0.002

Некоторая разница значений частот в приводимых в различных источниках таблицах объясняется тем, что частоты существенно зависят не только от длины текста, но и от его характера.

Устойчивыми являются также частотные характеристики биграмм, триграмм и четырехграмм осмысленных текстов.

Задание

1. Определить количество информации (по Хартли), содержащееся в заданном сообщении, при условии, что значениями являются буквы кириллицы.

«Фамилия Имя Отчество» завершил ежегодный съезд эрудированных школьников, мечтающих глубоко проникнуть в тайны физических явлений и химических реакций

2. Построить таблицу распределения частот символов, характерные для заданного сообщения. Производится так называемая частотная селекция, текст сообщения анализируется как поток символов и высчитывается частота встречаемости каждого символа. Сравнить с имеющимися данными в табл 1.
3. На основании полученных данных определить среднее и полное количество информации, содержащееся в заданном сообщении
4. Оценить избыточность сообщения.

Контрольные вопросы:

5. Как должна возрастать мера информации?
6. Как выглядит формула Хартли для измерения количества информации?
7. Как выглядит формула Шенона для измерения количества информации?
8. Что называется количеством информации?
9. Что такое избыточность информации?

Лабораторная работа № 4. Использование закона аддитивности информации при решении задач на определение количества информации

Цель работы: научить решать задачи на количественное измерение информационного объема текстовой информации.

Методические указания.

В связи с разными подходами к определению информации выделяют два подхода к измерению информации.

Субъективный (содержательный) подход

При данном подходе информация – это сведения, знания, которые человек получает из различных источников. Таким образом, сообщение информативно (содержит ненулевую информацию), если оно пополняет знания человека.

При субъективном подходе информативность сообщения определяется наличием в нем **новых знаний** и **понятностью** для данного человека (определение 1). Разные люди, получившие одно и тоже сообщение, по-разному оценивают количество информации, содержащееся в нем. Это происходит оттого, что знания людей об этих событиях, явлениях до получения сообщения были различными. Сообщение информативно для человека, если оно содержит новые сведения, и неинформативно, если сведения старые, известные. Таким образом, количество информации в сообщении зависит от того, насколько ново это сообщение для получателя и определяется объемом знаний, который несет это сообщение получающему его человеку.

При содержательном подходе возможна качественная оценка информации: достоверность, актуальность, точность, своевременность, полезность, важность, вредность…

С точки зрения информации как новизны мы не можем оценить количество информации, содержащейся в новом открытии, музыкальном стиле, новой теории развития.

Субъективный подход основывается на том, что получение информации, ее увеличение, означает уменьшение **незнания** или **информационной неопределенности** (определение 2).

Единица измерения количества информации называется **бит (**bit – binary digit), что означает двоичный разряд.

Количество информации – это количество бит в сообщении.

Сообщение, уменьшающее информационную неопределенность (неопределенность знаний) в два раза, несет для него **1 бит** информации.

Что же такое «информационная неопределенность»?

Информационная неопределенность о некотором событии – это количество возможных результатов события.

Пример_1: Книга лежит на одной из двух полок – верхней или нижней. Сообщение о том, что книга лежит на верхней полке, уменьшает неопределенность ровно вдвое и несет 1 бит информации.

Сообщение о том, что произошло одно событие из двух равновероятных, несет **1 бит** информации.

Научный подход к оценке сообщений был предложен еще в 1928 году Р. Хартли.

Пусть в некотором сообщении содержатся сведения о том, что произошло одно из N равновероятных событий (равновероятность обозначает, что ни одно событие не имеет преимуществ перед другими). Тогда количество информации, заключенное в этом сообщении, - x бит и число N связаны формулой:

$2^x = N$

где x – количество информации или информативность события (в битах);

N – число равновероятных событий (число возможных выборов).

Данная формула является показательным уравнением относительно неизвестной x. Решая уравнение, получим формулу определения количества информации, содержащемся в сообщении о том, что произошло одно из N равновероятных событий, которая имеет вид:

$x = \log_2 N$

логарифм от N по основанию 2.

Если N равно целой степени двойки, то такое уравнение решается легко, иначе справиться с решением поможет таблица логарифмов.

Если N = 2 (выбор из двух возможностей), то x = 1 бит.

Пример_4: Какое количество информации несет сообщение о том, что встреча назначена на июль?

Решение: В году 12 месяцев, следовательно, число равновероятных событий или число возможных выборов N = 12. Тогда количество информации $x = \log_2 12$. Чтобы решить это уравнение воспользуемся таблицей логарифмов или калькулятором.

Ответ: x = 3,58496 бита.

Пример_5: При угадывании целого числа в диапазоне от1 до N было получено 8 бит информации. Чему равно N?

Решение: Для того, чтобы найти число, достаточно решить уравнение $N=2^x$, где x = 8. Поскольку $2^8 = 256$, то N = 256. Следовательно, при угадывании любого целого числа в диапазоне от 1 до 256 получаем 8 бит информации.

Ситуации, при которых точно известно значение N, редки. Попробуйте по такому принципу подсчитать количество информации, полученное при чтении страницы книги. Это сделать невозможно.

Объективный (алфавитный) подход к измерению информации

Теперь познакомимся с другим способом измерения информации. Этот способ не связывает количество информации с содержанием сообщения, и называется **объективный** или **алфавитный** подход.

При объективном подходе к измерению информации мы отказываемся от содержания информации, от человеческой важности для кого-то.

Информация рассматривается как последовательность символов, знаков (определение3).

Количество символов в сообщении называется **длиной сообщения**.

Основой любого языка является алфавит.

Алфавит – это набор знаков (символов), в котором определен их порядок.

Полное число символов алфавита принято называть мощностью алфавита. Обозначим эту величину буквой M.

Например, мощность алфавита из русских букв равна 33:

мощность алфавита из английских букв равна 26.

При алфавитном подходе к измерению информации количество информации от содержания не зависит. Количество информации зависит от объема текста (т.е. от числа знаков в тексте) и от мощности алфавита. Тогда информацию можно обрабатывать, передавать, хранить.

Каждый символ несет x бит информации. Количество информации x, которое несет один символ в тексте, зависит от мощности алфавита M, которые связаны формулой $2^x = M$. Следовательно $x = \log_2 M$ бит.

Количество информации в тексте, состоящем из K символов, равно K*x или

$K^* \log_2 M$, где x – информационный вес одного символа алфавита.

Удобнее измерять информацию, когда мощность алфавита M равна целой степени числа 2. Для вычислительной системы, работающей с двоичными числами, также более удобно представление чисел в виде степени двойки.

Пример_6, в 2-символьном алфавите каждый символ несет 1 бит информации ($2^x = 2$, откуда x = 1 бит).

Если M=16, то каждый символ несет 4 бита информации, т.к. $2^4 = 16$.

Если M=32, то один символ несет 5 бит информации.

При M=64, один символ «весит» 6 бит и т.д.

Задания

1. Измерьте информационный объем сообщения «Ура! Скоро Новый год!» в битах, байтах, килобайтах (Кб), мегабайтах (Мб).

2. Измерьте примерную информационную емкость одной страницы любого своего учебника, всего учебника.

3. Сколько таких учебников может поместиться на дискете 1,44 Мб, на винчестере в 1 Гб.

4. В детской игре «Угадай число» первый участник загадывает целое число от 1 до 32. Второй участник задает вопросы: «Загаданное число больше числа ___?». Какое количество вопросов при правильной стратегии гарантирует угадывание?

Указание: Вопрос задавайте таким образом, чтобы информационная неопределенность (число вариантов) уменьшалась в два раза.

5. Яд находится в одном из 16 бокалов. Сколько единиц информации будет содержать сообщение о бокале с ядом?

6. Сколько бит информации несет сообщение о том, что из колоды в 32 карты достали «даму пик»?

7. Проводят две лотереи: «4 из 32» и «5 из 64» Сообщение о результатах какой из лотерей несет больше информации?

8. Информационное сообщение объемом 1.5 Кбайта содержит 3072 символа. Сколько символов содержит алфавит, при помощи которого было записано это сообщение? (Объяснение решения задачи на доске).

9. Подсчитать в килобайтах количество информации в тексте, если текст состоит из 600 символов, а мощность используемого алфавита – 128 символов.

10. Скорость информационного потока – 20 бит/сек. Сколько времени потребуется для передачи информации объемом в 10 килобайт.

11. Сравните (поставьте знак отношения)

- 200 байт и 0,25 Кбайт.
- 3 байта и 24 бита.
- 1536 бит и 1,5 Кбайта.
- 1000 бит и 1 Кбайт.
- 8192 байта и 1 Кбайт.

12. В барабане для розыгрыша лотереи находится 32 шара. Сколько информации содержит сообщение о первом выпавшем номере (например, выпал номер 15)?

13. При игре в кости используется кубик с шестью гранями. Сколько бит информации получает игрок при каждом бросании кубика?

14. Книга, набранная с помощью компьютера, содержит 150 страниц; на каждой странице — 40 строк, в каждой строке — 60 символов. Каков объем информации в книге?

15. Подсчитайте объем информации, содержащейся в романе А. Дюма "Три мушкетера", и определите, сколько близких по объему произведений можно разместить на одном лазерном диске? (590 стр., 48 строк на одной странице, 53 символа в строке).

16. На диске объемом 100 Мбайт подготовлена к выдаче на экран дисплея информация: 24 строчки по 80 символов, эта информация заполняет экран целиком. Какую часть диска она занимает?

17. В школьной библиотеке 16 стеллажей с книгами. На каждом стеллаже 8 полок. Библиотекарь сообщил Пете, что нужная ему книга находится на пятом стеллаже на третьей сверху полке. Какое количество информации библиотекарь передал Пете?

18. В коробке лежат 7 цветных карандашей. Какое количество информации содержит сообщение, что из коробки достали красный карандаш?

19. Какое количество информации несет сообщение: “Встреча назначена на сентябрь”.

20. Сообщение занимает 3 страницы по 25 строк. В каждой строке записано по 60 символов. Сколько символов в использованном алфавите, если все сообщение содержит 1125 байтов?

Контрольные вопросы:

10. Какой подход называется субъективным?
11. Какой подход называется содержательным?
12. Что называется информационной неопределенностью?
13. Какой подход называется объективным (алфавитным)?
14. Что называется длиной информации?

Лабораторная работа № 5. Применение алфавитного подхода к измерению информации при решении задач на определение количества информации.

Цель работы: научиться применять алфавитный подход к измерению информации при решении задач на определение количества информации

Методические указания.

При содержательном подходе к измерению количества информации информационное сообщение рассматривают с точки зрения его содержания. Насколько полученная информация способствует расширению человеческих знаний (уменьшению степени нашего незнания или уменьшению неопределённости нашего знания о каком-либо объекте, событии, явлении, процессе и пр.).

Например:

При бросании монеты есть всего два варианта возможного результата – «орёл» или «решка». Значит, неопределенность знаний о результате бросания монеты равна двум.

Например:

После выполнения контрольной работы вы не знаете, какую оценку получили. Учитель объявляет результаты, и вы получаете одно из четырёх информационных сообщений «2», «3», «4», «5», которые приводят к уменьшению неопределённости ваших знаний в 4 раза.

Количественно измерить информацию как меру уменьшения неопределённости можно по формуле:

$N = 2^i$, где

N – количество возможных равновероятных событий;

i – количество информации, заключенное в одном событии.

Например:

- в примере 1 - N = 2 (орёл – решка), следовательно

$2 = 2^i$

i = 1 бит.

- в примере 2 - N = 4 («2», «3», «4», «5»), следовательно

$4 = 2^i$

i = 2 бита.

При алфавитном подходе к измерению количества информации информационное сообщение рассматривают как последовательность знаков определённого алфавита. Алфавитный подход является объективным, то есть не зависит от субъекта (человека), воспринимающего текст. Множество символов, используемых при записи текста, называется алфавитом.

Полное количество символов в алфавите называется мощностью (размером) алфавита. Если допустить, что все символы алфавита встречаются в тексте с одинаковой равновероятностью (N), то количество информации можно определить следующим образом:

$N = 2^i$,

где N – мощность алфавита

i – информационный вес одного символа (измеряется в битах).

Если весь текст состоит из К символов, то размер содержащейся в нём информации равен:

$I = K \cdot i$

Информационный вес символа компьютерного алфавита равен 8 бит (23) или 1 байту.

Для измерения больших объемов информации используют более крупные единицы:

1 килобайт (Кбайт или Kb) = 1024 байт (210)

1 мегабайт (Мбайт или Mb) = 1024 Кбайт (220)

1 гигабайт (Гбайт или Gb) = 1024 Мбайт (230)

1 терабайт (Тбайт или Tb) = 1024 Гбайтам (240)

Самая наибольшая единица измерения информации - йоттабайт. Она равна 1024 зеттабайтам, зеттабайт = 1024 эксабайтам, эксабайт = 1024 петабайтам, а петабайт = 1024 терабайтам.

Задание

1. Откройте текстовый редактор «Блокнот».
2. Сохраните файл под именем «Лабораторная работа 5» в своей папке.
3. Решите задачи.

№1. Некоторый алфавит содержит 128 символов. Сообщение содержит 10 символов. Определите объем сообщения.

№2. Считая, что один символ кодируется 8-ю битами, оцените информационный объем следующей поговорки в кодировке КОИ-8: Верный друг лучше сотни слуг.

№3. Один и тот же текст на русском языке записан в различных кодировках. Текст, записанный в 16-битной кодировке Unicode, на 120 бит больше текста, записанного в 8-битной кодировке КОИ-8. Сколько символов содержит текст?

№4. Сколько гигабайт содержит файл объемом 235 бит?

№5. Текстовый файл copia.txt имеет объем 40960 байт. Сколько таких файлов можно записать на носитель объемом 5 Мбайт?

№6. К текстовому сообщению объемом 46080 байт добавили рисунок объемом 2,5 Мбайт. Сколько кбайт информации содержит полученное сообщение?

№7. В алфавите некоторого языка два символа X и O. Слово состоит из четырех символов, например: OOXO, XOOX. Укажите максимально возможное количество слов в этом языке.

№8. Для записи текста использовался 64-символьный алфавит. Сколько символов в тексте, если его объем равен 8190 бита?

№9. Укажите наибольшее натуральное число, которое можно закодировать 8 битами (если все числа кодируется последовательно, начиная с единицы).

№10. Некоторый алфавит содержит 2 символа. Сообщение занимает 2 страницы, на каждой по 16 строк, и в каждой строке по 32 символа. Определите объем сообщения.

№11. Сколько бит информации содержится в сообщении объемом 1/4 килобайта?

№12. Найдите x из следующего соотношения: 8x бит = 16 Мбайт.

№13. Цветное растровое графическое изображение с палитрой 256 цветов имеет размер 64x128 пикселей. Какой информационный объем имеет изображение?

№14. Для хранения растрового изображения размером 64x128 пикселей отвели 4 Кбайта памяти. Каково максимально возможное количество цветов в палитре изображения?

4. Подготовьте отчёт о выполнении задания: напишите ваш ответ, рядом с номером задачи.
5. Сохраните внесённые в текстовый документ изменения.

Контрольные вопросы:

1. Как при алфавитном подходе подходят к измерению количества?
2. Как рассматривается информации информационное сообщение при алфавитном подходе?
3. Чему равен информационный вес символа компьютерного алфавита?
4. Что называется мощностью алфавита?
5. От чего не зависит алфавитный подход?

Лабораторная работа № 6. Применение теоремы Котельникова.

Цель работы: приметнеие теоремы Котельникова при кодирование информации

Методические указания.

При этом как следует из названия, символы некоторого первичного алфавита (например, русского) кодируются комбинациями символов двоичного алфавита (т.е. *0* и *1*), причем, длина кодов и, соответственно, длительность передачи отдельного кода, могут различаться. Длительности элементарных сигналов при этом одинаковы *($\tau_0=\tau_1=\tau$).* За счет чего можно оптимизировать кодирование в этом случае? Очевидно, суммарная длительность сообщения будет меньше, если применить следующий подход: тем буквам первичного алфавита, которые встречаются *чаще*, присвоить более *короткие* по длительности коды, а тем, относительная частота которых меньше – коды более длинные. Но длительность кода – величина дискретная, она *кратна* длительности сигнала τпередающего один символ двоичного алфавита. Следовательно, коды букв, вероятность появления которых в сообщении выше, следует строить из возможно меньшего числа элементарных сигналов. Построим кодовую таблицу для букв русского алфавита, Очевидно, возможны различные варианты двоичного кодирования, однако, не все они будут пригодны для практического использования – важно, чтобы закодированное сообщение могло быть *однозначно декодировано*, т.е. чтобы в последовательности *0* и *1*, которая представляет собой многобуквенное кодированное сообщение, всегда можно было бы различить обозначения отдельных букв. Проще всего этого достичь, если коды будут разграничены *разделителем* – некоторой постоянной комбинацией двоичных знаков. Условимся, что разделителем отдельных кодов букв будет последовательность *00* (признак конца знака), а разделителем слов – *000* (признак конца слова – пробел). Довольно очевидными оказываются следующие правила построения кодов:

- код признака конца знака может быть включен в код буквы, поскольку не существует отдельно (т.е. кода всех букв будут заканчиваться *00*);
- коды букв не должны содержать двух и более нулей подряд в середине (иначе они будут восприниматься как конец знака);
- код буквы (кроме пробела) всегда должен начинаться с *1*;
- разделителю слов (*000*) всегда предшествует признак конца знака; при этом реализуется последовательность *00000* (т.е. если в конце кода встречается комбинация *…000* или *…0000*, они не воспринимаются как разделитель слов); следовательно, коды букв могут оканчиваться на *0* или *00* (до признака конца знака).

Длительность передачи каждого отдельного кода t_i, очевидно, может быть найдена следующим образом: $t_i = k_i \bullet \tau$, где k_i – количество элементарных сигналов (бит) в коде символа *i*. В соответствии с приведенными выше правилами получаем следующую таблицу кодов:

Таблица 1.

Буква	**Код**	**$p_i \cdot 10^3$**	**k_i**	**Буква**	**Код**	**$p_i \cdot 10^3$**	**k_i**
пробел	***000***	**174**	**3**	**я**	***1011000***	**18**	**7**
о	***100***	**90**	**3**	**ы**	***1011100***	**16**	**7**
е	***1000***	**72**	**4**	**з**	***1101000***	**16**	**7**
а	***1100***	**62**	**4**	**ь,ъ**	***1101100***	**14**	**7**
и	***10000***	**62**	**5**	**б**	***1110000***	**14**	**7**
т	***10100***	**53**	**5**	**г**	***1110100***	**13**	**7**
н	***11000***	**53**	**5**	**ч**	***1111000***	**12**	**7**
с	***11100***	**45**	**5**	**й**	***1111100***	**10**	**7**
р	***101000***	**40**	**6**	**х**	***10101000***	**9**	**8**
в	***101100***	**38**	**6**	**ж**	***10101100***	**7**	**8**

л	***110000***	**35**	**6**	**ю**	***10110000***	**6**	**8**
к	***110100***	**28**	**6**	**ш**	***10110100***	**6**	**8**
м	***111000***	**26**	**6**	**ц**	***10111000***	**4**	**8**
д	***111100***	**25**	**6**	**щ**	***10111100***	**3**	**8**
п	***1010000***	**23**	**7**	**э**	***11010000***	**3**	**8**
у	***1010100***	**21**	**7**	**ф**	***11010100***	**2**	**8**

Теперь по формуле можно найти среднюю длину кода $K^{(2)}$ для данного способа кодирования:

$$K^{(2)} = \sum_{i=1}^{32} p_i \cdot k_i = 4,964$$

Поскольку для русского языка $I_1^{(r)} = 4,356$ *бит*, избыточность данного кода, составляет:

$Q^{(r)} = 1 - 4,356/4,964 \approx 0,122;$

это означает, что при данном способе кодирования будет передаваться приблизительно на 12% больше информации, чем содержит исходное сообщение. Аналогичные вычисления для английского языка дают значение $K^{(2)} = 4,716$, что при $I_1^{(e)} = 4,036$ *бит* приводят к избыточности кода $Q^{(e)} = 0,144$.

Рассмотрев один из вариантов двоичного неравномерного кодирования, попробуем найти ответы на следующие вопросы: возможно ли такое кодирование без использования разделителя знаков? Существует ли наиболее оптимальный способ неравномерного двоичного кодирования?

Суть первой проблемы состоит в нахождении такого варианта кодирования сообщения, при котором последующее выделение из него каждого отдельного знака (т.е. декодирование) оказывается однозначным без специальных указателей разделения знаков. Наиболее простыми и употребимыми кодами такого типа являются так называемые *префиксные коды*, которые удовлетворяют следующему условию (*условию Фано*):

Неравномерный код может быть однозначно декодирован, если никакой из кодов не совпадает с началом (префиксом [1]) какого-либо иного более длинного кода.

Например, если имеется код *110*, то уже не могут использоваться коды *1*, *11*, *1101*, *110101* и пр. Если условие Фано выполняется, то при прочтении (расшифровке) закодированного сообщения путем сопоставления со списком кодов всегда можно точно указать, где заканчивается один код и начинается другой.

Задание

Пусть имеется следующая таблица префиксных кодов:

а	**л**	**м**	**р**	**у**	**ы**
10	***010***	***00***	***11***	***0110***	***0111***

Требуется декодировать сообщение: *00100010000111010101110000110*

Декодирование производится циклическим повторением следующих действий:

1. отрезать от текущего сообщения крайний левый символ, присоединить к рабочему кодовому слову;
2. сравнить рабочее кодовое слово с кодовой таблицей; если совпадения нет, перейти к (1);
3. декодировать рабочее кодовое слово, очистить его;
4. проверить, имеются ли еще знаки в сообщении; если «да», перейти к (1).

Применение данного алгоритма дает:

Шаг	Рабочее слово	Текущее сообщение	Распознанный знак	Декодированное сообщение
0	пусто	**0**0100010000111010101110000110	–	–
1	0	**0**1000100001110101011100001 10	нет	–
2	**00**	**1**0001000011101010111000011 0	**м**	м
3	1	**0**0010000111010101110000110	нет	м
4	**10**	**0**0100001110101011100001 10	**а**	ма
5	0	**0**1000011101010111000011 0	нет	ма
6	**00**	**1**0000111010101110000110	**м**	мам
…				

Доведя процедуру до конца, получим сообщение: «*мама мыла раму*».

Таким образом, использование префиксного кодирования позволяет делать сообщение более коротким, поскольку нет необходимости передавать разделители знаков. Однако, условие Фано не устанавливает способа формирования префиксного кода и, в частности, наилучшего из возможных.

Контрольные вопросы:

15. Формулировка теоремы Котельникова
16. Что называется разделителем?
17. Какие коды называются префиксными кодами?
18. В чем заключается условие Фано?
19. Когда может быть декодированнеравномерный код?

Лабораторная работа № 7. Кодирование и декодирование информации

Цель работы: криптоанализ и программная реализация алгоритмов перестановок для шифрования и дешифрования исходного текста.

Методические указания.

Шифры перестановки

Шифр, преобразования из которого изменяют только порядок следования символов исходного текста, но не изменяют их самих, называется *шифром перестановки* (ШП).

Пусть имеем сообщение из *n* символов. Его можно представить с помощью таблицы:

$$\begin{pmatrix} 1 & 2 & - & - & n \\ i_1 & i_2 & - & - & i_n \end{pmatrix},$$

где i_1 - номер места зашифрованного текста, на которое попадает I-ая буква исходного сообщения при выбранном преобразовании, i_2 - номер места для II-й буквы и т.д. В верхней строке таблицы выписаны по порядку числа от 1 до *n*, а в нижней - те же числа, но в произвольном порядке. Такая таблица называется подстановкой степени *n*.

Зная подстановку, задающую преобразование, можно как зашифровать, так и расшифровать текст.

Например, если для преобразования используется подстановка:

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 5 & 2 & 3 & 1 & 4 & 6 \end{pmatrix},$$

и в соответствии с ней зашифровывается слово МОСКВА, то получится слово КОСМВА.

Итак, используя метод математической индукции, определим, что существует *n!* вариантов заполнения нижней строки таблицы. Т.е. число различных преобразований шифра перестановки, предназначенного для зашифрования сообщения длины *n*, меньше либо равно *n!*. При больших *n* для вычисления *n!* можно пользоваться формулой Стирлинга:

$$n! \approx \sqrt{2\pi n}\left(\frac{n}{e}\right)^n, \text{ ãääå } e \approx 2{,}71\ldots \quad .$$

Примером ШП, предназначенного для зашифрования сообщения длины *n*, является шифр, в котором в качестве множества ключей взято множество всех подстановок степени *n*. Число ключей такого шифра =*n!*.

Для использования на практике такой шифр не удобен, т.к. при больших значениях *n* приходится работать с длинными таблицами.

Примеры простейших шифров перестановок

Широкое распространение получили шифры перестановки, использующие некоторую геометрическую фигуру. Преобразования из этого шифра состоят в том, что в фигуру исходный текст вписывается по ходу одного «маршрута», а затем по ходу другого выписывается из нее. Такой шифр называют *маршрутной перестановкой*. Например, можно вписывать исходное сообщение в прямоугольную таблицу, выбрав такой маршрут по горизонтали, начиная с левого верхнего угла поочередно слева направо и справа налево. Используем прямоугольник размера 4×7

➔

П	Р	И	М	Е	Р	М	↓
Н	Т	У	Р	Ш	Р	А	←
О	Й	П	Е	Р	Е	С	↓
И	К	В	О	Н	А	Т	←

Выписывать будем по вертикали, начиная с верхнего правого угла и двигаясь поочередно сверху вниз и снизу вверх.

МАСТАЕРРЕШРНОЕРМИУПВКЙТРПНОИ

Теоретически, маршруты могут быть более изощренными.

Шифр «Сцитала» эквивалентен следующему шифру маршрутной перестановки: в таблицу, состоящую из *m* столбцов, построчно записывается сообщение, после чего выписывают буквы по столбцам. Число задействованных столбцов таблицы не может превосходить длины сообщения.

Из истории имеются еще чисто физические ограничения, накладываемые реализацией шифра Сцитала. Естественно предположить, что диаметр жезла не должен превосходить 10 сантиметров. При высоте строки в 1 см на одном витке такого жезла уместится не более 32 букв ($10\pi < 32$). Таким образом, число перестановок, реализуемых Сциталой, не больше 32.

Шифр вертикальной перестановки (ШВП). В нем снова используется прямоугольник, в который сообщение вписывается обычным способом (по строкам слева направо). Выписываются буквы по вертикали, а столбцы при этом берутся в порядке, определяемом ключом. Впишем сообщение в прямоугольник, столбцы которого пронумерованы в соответствии с ключом:

5	1	4	7	2	6	3
В	О	Т	П	Р	И	М
Е	Р	Ш	И	Ф	Р	А
В	Е	Р	Т	И	К	А
Л	Ь	Н	О	Й	П	Е
Р	Е	С	Т	А	Н	О
В	К	И	-	-	-	-

Выбирая столбцы в порядке, заданном ключом, выписываем последовательно буквы сверху вниз:

ОРЕЬЕКРФИЙА-МААЕО-ТШРНСИВЕВЛРВИРКПН-ПИТОТ-

Число ключей ШВП не более m!, где *m* - число столбцов таблицы. Пользуясь формулой Стирлинга, при больших *m* и *n* можно оценить, во сколько раз n! больше m!, если *n* кратно *m*.

В случае, когда ключ не рекомендуется записывать, его можно извлекать из какого-то легко запоминающегося слова или предложения.

Например, пусть ключевым словом будет ПЕРЕСТАНОВКА. Буква А получает номер 1. Если какая-то буква входит несколько раз, то ее появления нумеруются последовательно слева направо. Таким образом, второе вхождение А получает номер 2. Поскольку Б нет, то В получает номер 3 и т.д., пока все буквы не получат номера:

П	Е	Р	Е	С	Т	А	Н	О	В	К	А
9	4	1	5	1	1	1	7	8	3	6	2
		0		1	2						

Транспозиция с фиксированным периодом d. В этом случае сообщение делится на группы символов длины *d* и к каждой группе применяется одна и та же перестановка. Эта перестановка является ключом; она может быть задана некоторой перестановкой первых *d* целых чисел. Таким образом, для *d=5* в качестве перестановки можно взять: 23154. Это будет означать, что:

$m_1 m_2 m_3 m_4 m_5 m_6 m_7 m_8 m_9 m_{10} \ldots$

переходит в:

$$m_2 m_3 m_1 m_5 m_4 m_7 m_8 m_6 m_{10} m_9 \ldots .$$

Последовательное применение двух или более транспозиций будет называться *составной транспозицией*. Если периоды этих транспозиций $d_1,\ldots, d_s$, то, очевидно, в результате получится транспозиция периода *d*, где *d* - наименьшее общее кратное $d_1,\ldots, d_s$.

Задание

1. Используя один из криптографических алгоритмов перестановок, составить программу для шифрования и дешифрования текста.
2. Подсчитать количество возможных ключей выбранного шифра, оценить стойкость шифра перестановок, сравнить с шифрами замены, сделать выводы.

Контрольные вопросы

1. Что называют шифрами перестановок? Дать определение и привести общий алгоритм.
2. Какие алгоритмы шифров перестановок используются на практике?
3. К какому классу относится древний шифр Сцитала, разновидностью какого шифра он является и как реализуется?
4. Какой шифр называют шифром маршрутной перестановки?
5. В чем смысл шифра вертикальной перестановки?
6. Для чего применяется формула Стирлинга?
7. Что такое транспозиция?
8. Каков ключ составной транспозиции?

Лабораторная работа № 10. Применение формулы Шенона.

Цель работы: выработать навыки применения формулы Шенона

Методические указания.

В первую очередь для решения задач такого типа нам необходимо знать формулу расчета вероятности исхода. Она выглядит так:

p=M/N,

где M – это величина, показывающая сколько раз произошло интересующее нас событие, N – это общее число возможных исходов какого-то процесса.

Необходимо знать, что в сумме все вероятности дают единицу или в процентном выражении 100%.

Далее для решения задач на количество информации необходимо знать, каким событие является: равновероятным или с разными вероятностями.

Если событие равновероятно, можно применить для решения формулу Хартли:

$$I=\log_2 N,$$

где N – это количество равновероятных событий; I - количество бит в сообщении

Иногда формулу Хартли записывают так:

$$I = \log_2 N = \log_2 (1 / p) = - \log_2 p,$$

т. к. каждое из N событий имеет равновероятный исход p = 1 / N, то N = 1 / p.

Важно запомнить, что вероятность события и количество информации в сообщении имеют связь, которую можно выразить следующим образом: ***чем меньше вероятность некоторого события, тем больше информации содержит сообщение об этом событии.***

Также следует упомянуть - существует закономерность, ***что количество информации достигает максимального значения, если события равновероятны.***

Если в задаче дано событие с различными вероятностями, тогда следует использовать формулу Шеннона.

$$I = -\sum_{i=1}^{N} p_i * \log_2 p_i$$

где **I**-количество информации,
N-количество возможных событий,
$\mathbf{p_i}$ - вероятности отдельных событий

Задание 1.

В классе 30 человек. За контрольную работу по информатике получено 15 пятерок, 6 четверок, 8 троек и 1 двойка. Какое количество информации несет сообщение о том, что Андреев получил пятерку?

Задание 2.

В непрозрачном мешочке хранятся 10 белых, 20 красных, 30 синих и 40 зеленых шариков. Какое количество информации будет содержать зрительное сообщение о цвете вынутого шарика?

Задание 3.

Построить код Шеннона-Фано и вычислить его эффективность для источника с вероятностями букв 1/4; 1/4; 1/8; 1/8; 1/16; 1/16; 1/16; 1/16.

Задание 4.

Построить блочный код Шеннона-Фано с блоками длиной 3 и вычислить его эффективность для однородного марковского источника с матрицей переходных вероятностей

$$p_{ij} = p(u_j|u_i) = \begin{pmatrix} 1/3 & 2/3 \\ 3/4 & 1/4 \end{pmatrix}.$$

Задание 5.

Декодировать полученное сообщение11011101. При кодировании использовался (7, 4) код Хэмминга с проверкой четности.

Контрольные вопросы:

1. Код Шеннона-Фано.
2. Код Хаффмана.
3. Помехоустойчивое кодирование. Теорема Шеннона о кодировании для канала с шумом.
4. Код с проверкой четности. Код с тройными повторениями.
5. Код Хэмминга.

Лабораторная работа № 9. Алфавитное неравномерное двоичное кодирование

Цель работы: исследование простейших методов криптографической зашиты информации.

Методические указания.

Шифры простой замены

Система шифрования Цезаря - частный случай шифра простой замены. Метод основан на замене каждой буквы сообщения на другую букву того же алфавита, путем смещения от исходной буквы на K букв.

Известная фраза Юлия Цезаря

VENI VI D I VICI, где

A	B	C	D	E	F	G	H	I	G	K	L	M
N	O	P	Q	R	S	T	U	V	W	X	Y	Z

пришел, увидел, победил, зашифрованная с помощью данного метода, преобразуется в

SBKF SFAF SFZF

при смещении на 4 символа влево.

Греческим писателем Полибием за 100 лет до н.э. был изобретен так называемый **полибианский квадрат** размером 5*5, заполненный алфавитом в случайном порядке. Греческий алфавит имеет 24 буквы, а 25-м символом является пробел. Для шифрования на квадрате находили букву текста и записывали в зашифрованное сообщение букву, расположенную ниже ее в том же столбце. Если буква оказывалась в нижней строке таблицы, то брали верхнюю букву из того же столбца.

M	T	L	E	X
A	K	F	Q	Y
N	B	R	O	W
C	J	H	D	P
U	I	S	G	V

Схема шифрования Вижинера. Таблица Вижинера представляет собой квадратную матрицу с n^2 элементами, где n — число символов используемого алфавита. На рисунке показана верхняя часть таблицы Вижинера для кириллицы. Каждая строка получена циклическим сдвигом алфавита на символ. Для шифрования выбирается буквенный ключ, в соответствии с которым формируется рабочая матрица шифрования.

а	б	в	г	д	е	ё	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я
б	в	г	д	е	ё	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а
в	г	д	е	ё	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б
г	д	е	ё	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в
д	е	ё	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г
е	ё	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д
И т.д. до 33-ей строки..																																

Таблица Вижинера

Осуществляется это следующим образом. Из полной таблицы выбирается первая строка и те строки, первые буквы которых соответствуют буквам ключа. Первой размещается первая строка, а под нею — строки, соответствующие буквам ключа в порядке следования этих букв в ключе шифрования. Пример такой рабочей матрицы для ключа «книга» приведен на **Рис. 3.1.3**.

Процесс шифрования осуществляется следующим образом:

1. под каждой буквой шифруемого текста записываются буквы ключа. Ключ при этом повторяется необходимое число раз.

2. каждая буква шифруемого текста заменяется по подматрице буквами находящимися на пересечении линий, соединяющих буквы шифруемого текста в первой строке подматрицы и находящимися под ними букв ключа.

3. полученный текст может разбиваться на группы по несколько знаков.

Пусть, например, требуется зашифровать сообщение: *максимально допустимой ценой является пятьсот руб. за штуку*. В соответствии с первым правилом записываем под буквами шифруемого текста буквы ключа. Получаем:

```
максимально  допустимой  ценой  является пятьсот руб. за штуку
книгакнигак  нигакнигак   нигак  нигакниг акнигак ниг  ак нигак
```

Дальше осуществляется непосредственное шифрование в соответствии со вторым правилом, а именно: берем первую букву шифруемого текста (М) и соответствующую ей букву ключа (К); по букве шифруемого текста (М) входим в рабочую матрицу шифрования и выбираем под ней букву, расположенную в строке, соответствующей букве ключа (К),— в нашем примере такой буквой является Ч; выбранную таким образом букву помещаем в зашифрованный текст. Эта процедура циклически повторяется до зашифрования всего текста.

Эксперименты показали, что при использовании такого метода статистические характеристики исходного текста практически не проявляются в зашифрованном сообщении. Нетрудно видеть, что замена по таблице Вижинера эквивалентна простой замене с циклическим изменением алфавита, т.е. здесь мы имеем полиалфавитную подстановку, причем число используемых алфавитов определяется числом букв в слове ключа. Поэтому стойкость такой замены определяется произведением стойкости прямой замены на число используемых алфавитов, т.е. число букв в ключе.

Расшифровка текста производится в следующей последовательности:

1. над буквами зашифрованного текста последовательно надписываются буквы ключа, причем ключ повторяется необходимое число раз.
2. в строке подматрицы Вижинера, соответствующей букве ключа отыскивается буква, соответствующая знаку зашифрованного текста. Находящаяся под ней буква первой строки подматрицы и будет буквой исходного текста.
3. полученный текст группируется в слова по смыслу.

Нетрудно видеть, что процедуры как прямого, так и обратного преобразования являются строго формальными, что позволяет реализовать их алгоритмически. Более того, обе процедуры

а	б	в	г	д	е	ё	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я
к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ё	ж	з	и	й
н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ё	ж	з	и	й	к	л	м
и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в	г	д	е	ё	ж	з
г	д	е	ё	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я	а	б	в
а	б	в	г	д	е	ё	ж	з	и	й	к	л	м	н	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я

легко реализуются по одному и тому же алгоритму.

Одним из недостатков шифрования по таблице Вижинера является то, что при небольшой длине ключа надежность шифрования остается невысокой, а формирование длинных ключей сопряжено с трудностями.

Нецелесообразно выбирать ключи с повторяющимися буквами, так как при этом стойкость шифра не возрастает. В то же время ключ должен легко запоминаться, чтобы его можно было не записывать. Последовательность же букв не имеющих смысла, запомнить трудно.

С целью повышения стойкости шифрования можно использовать усовершенствованные варианты таблицы Вижинера. Приведем только некоторые из них:

- во всех (кроме первой) строках таблицы буквы располагаются в произвольном порядке.

• В качестве ключа используется случайность последовательных чисел. Из таблицы Вижинера выбираются десять произвольных строк, которые кодируются натуральными числами от 0 до 10. Эти строки используются в соответствии с чередованием цифр в выбранном ключе.

Известны также и многие другие модификации метода.

Алгоритм перестановки

Этот метод заключается в том, что символы шифруемого текста переставляются по определенным правилам внутри шифруемого блока символов. Рассмотрим некоторые разновидности этого метода, которые могут быть использованы в автоматизированных системах.

Самая простая перестановка — написать исходный текст задом наперед и одновременно разбить шифрограмму на пятерки букв. Например, из фразы

ПУСТЬ БУДЕТ ТАК, КАК МЫ ХОТЕЛИ.

получится такой шифротекст:

ИЛЕТО ХЫМКА ККАТТ ЕДУБЪ ТСУП

В последней группе (пятерке) не хватает одной буквы. Значит, прежде чем шифровать исходное выражение, следует его дополнить незначащей буквой

(например, О) до числа, кратного пяти:

ПУСТЬ-БУДЕТ-ТАККА-КМЫХО-ТЕЛИО.

Тогда шифрограмма, несмотря на столь незначительные изменения, будет выглядеть по-другому:

ОИЛЕТ ОХЫМК АККАТ ТЕДУБ ЬТСУП

Кажется, ничего сложного, но при расшифровке проявляются серьезные неудобства.

Во время Гражданской войны в США в ходу был такой шифр: исходную фразу писали в несколько строк. Например, по пятнадцать букв в каждой (с заполнением последней строки незначащими буквами).

```
П У С Т Ь Б У Д Е Т Т А К К А
К М Ы Х О Т Е Л И К Л М Н О П
```

После этого вертикальные столбцы по порядку писали в строку с разбивкой на пятерки букв:

ПКУМС ЫТХЬО БТУЕД ЛЕИТК ТЛАМК НКОАП

Если строки укоротить, а количество строк увеличить, то получится прямоугольник-решетка, в который можно записывать исходный текст. Но тут уже потребуется предварительная договоренность между адресатом и отправителем посланий, поскольку сама решетка может быть различной длины-высоты, записывать к нее можно по строкам, по столбцам, по спирали туда или по спирали обратно, можно писать и по диагоналями, а для шифрования можно брать тоже различные направления.

Шифры сложной замены

Шифр Гронсфельда состоит в модификации шифра Цезаря числовым ключом. Для этого под буквами сообщения записывают цифры числового ключа. Если ключ короче сообщения, то его запись циклически повторяют. Зашифрованное сообщение получают примерно также, как в шифре Цезаря, но используют не одно жестко заданное смещение а фрагменты ключа.

Пусть в качестве ключа используется группа из трех цифр – 314, тогда сообщение

```
С О В Е Р Ш Е Н Н О С Е К Р Е Т Н О
3 1 4 3 1 4 3 1 4 3 1 4 3 1 4 3 1 4
---------------------------------------
Ф П Ё С Ь З О С С А Х З Л Ф З У С С
```

В **шифрах многоалфавитной замены** для шифрования каждого символа исходного сообщения применяется свой шифр простой замены (свой алфавит).

	АБВГДЕЁЖЗИКЛМНОПРСТУФХЧШЩЪЫЬЭЮЯ
А	АБВГДЕЁЖЗИКЛМНОПРСТУФХЧШЩЪЫЬЭЮЯ_
Б	_АБВГДЕЁЖЗИКЛМНОПРСТУФХЧШЩЪЫЬЭЮЯ

В	Я_АБВГДЕЁЖЗИКЛМНОПРСТУФХЧШЩЪЫЬЭЮ
Г	ЮЯ_АБВГДЕЁЖЗИКЛМНОПРСТУФХЧШЩЪЫЬЭ
.	……
Я	ВГДЕЁЖЗИКЛМНОПРСТУФХЧШЩЪЫЬЭЮЯ_АБ
_	БВГДЕЁЖЗИКЛМНОПРСТУФХЧШЩЪЫЬЭЮЯ_А

Каждая строка в этой таблице соответствует одному шифру замены аналогично шифру Цезаря для алфавита, дополненного пробелом. При шифровании сообщения его выписывают в строку, а под ним ключ. Если ключ оказался короче сообщения, то его циклически повторяют. Зашифрованное сообщение получают, находя символ в колонке таблицы по букве текста и строке, соответствующей букве ключа. Например, используя ключ АГАВА, из сообщения ПРИЕЗЖАЮ ШЕСТОГО получаем следующую шифровку:

ПРИЕЗЖАЮ_ШЕСТОГО
АГАВААГАВААГАВАА
ПОИГЗЖЮЮЮШЕПТНГО

Такая операция соответствует сложению кодов ASCII символов сообщения и ключа по модулю 256.

Задание

Придумайте 3 фразы, каждая минимум из 7 слов. Реализуйте шифрование этой фразы всеми перечисленными видами шифрования.

Контрольные вопросы:

20. В чем заключается система шифрования Цезаря?
21. Как используется схема Вижинера?
22. Объясните сущность алгоритма перестановки
23. Из чего состоит Шифр Гронсфельда?
24. Как производится расшифровка текста?

Лабораторная работа № 10. Решение задач с использованием оптимального кодирования информации

Цель: Познакомиться с различными кодировками символов, используя текстовые редакторы, выполнить задания в различных текстовых приложениях.

Методические указания

Правило цифрового представления символов следующее: каждому символу ставится в соответствие некоторое целое число, то есть каждый символ нумеруется.

Пример:

Рассмотрим последовательность строчных букв русского алфавита: а, б, в, г, д, е, ё, ж, з, и, й. к, л, м. н. о, п, р, с, т, у, ф, х, ц, ч, ш, щ, ъ, ы, в, э, ю, я. Присвоив каждой букве номер от 0 до 33. получим простейший способ представления символов. Последнее число - 32 в двоичной форме имеет вид 100000, то есть для хранения символа в памяти понадобится 6 бит.Так как с помощью шести бит можно представить число 2^6 - 1 = 63, то шести бит будет достаточно для представления 64 букв.

Имеются разные стандарты для представления, символов, которые отличаются лишь порядком нумерации символов. Наиболее распространён американский стандартный код для информационного обмена - ASCII [American Standard-Code for Information Interchange] введён в США в 1963г. В 1977 году в несколько модифицированном виде он был принят в качестве всемирного стандарта Международной организации стандартов [International Standards Organization -. ISO] под названием ISO-646. Согласно этому стандарту каждому символу поставлено в соответствие число от 0 до 255. Символы от 0 до 127 - латинские буквы, цифры и знаки препинания - составляют постоянную часть таблицы. Остальные символы используются для представления национальных алфавитов. Конкретный состав этих символов определяется кодовой страницей. В русской версии ОС Windows95 используется кодовая, страница 866. В ОС Linux для представления русских букв более употребительна кодировка КОИ-8. Недостатки такого способа кодировки национального, алфавита очевидны. Во-первых, невозможно одновременное представление русских и ,например, французских букв. Во-вторых, такая кодировка совершенно непригодна для представления, китайских иероглифов. В 1991 году была создана некоммерческая организация Unicode, в которую входят представители ряда фирм (Borland. IBM, Noyell, Sun и др) и которая занимается развитием и внедрением нового стандарта. Кодировка Unicode использует 16 разрядов ,и может содержать 65536 символов. Это символы большинства народов мира, элементы иероглифов, спецсимволы, 5000 – мест для частного использования, резерв из 30000 мест.

Пример:

ASCII-код символа А= 65_{10} =41_{16}= 01000111_2;

Unicode-код символа С= 67_{10}=0000000001100111_2

Задания

1. Закодируйте свое имя, фамилию и отчество с помощью одной из таблиц (win-1251, KOI-8)
2. Раскодируйте ФИО соседа
3. Закодируйте следующие слова, используя таблицы ASCII-кодов: ИНФОРМАТИЗАЦИЯ, МИКРОПРОЦЕССОР, МОДЕЛИРОВАНИЕ
4. Раскодируйте следующие слова, используя таблицы ASCII-кодов:

88 AD E4 AE E0 AC A0 E2 A8 AA A0

50 72 6F 67 72 61 6D

43 6F 6D 70 75 74 65 72 20 49 42 4D 20 50 43

5. Текстовый редактор Блокнот

Открыть блокнот.

а) Используя клавишу Alt и малую цифровую клавиатуру раскодировать фразу: 145 170 174 224 174 255 170 160 173 168 170 227 171 235;

Технология выполнения задания: При удерживаемой клавише Alt, набрать на малой цифровой клавиатуре указанные цифры. Отпустить клавишу Alt, после чего в тексте появится буква, закодированная набранным кодом.

б) Используя ключ к кодированию, закодировать слово – зима;

Технология выполнения задания: Из предыдущего задания выяснить, каким кодом записана буква а. Учитывая, что буквы кодируются в алфавитном порядке, выяснить коды остальных букв.

Что вы заметили при выполнении этого задания во время раскодировки? Запишите свои наблюдения.

6. Текстовый процессор MS Word.

Технология выполнения задания: рассмотрим на примере: представить в различных кодировках слово Кодировка

Решение:

- Создать новый текстовый документ в Word;
- Выбрать – Команда – Вставка – Символ.

В открывшемся окне «Символ» установить из: Юникод (шестн.),

- В наборе символов находим букву **К** и щелкнем на ней левой кнопкой мыши (ЩЛКМ).
- В строке код знака появится код выбранной буквы 041А (незначащие нули тоже записываем).
- У буквы **о** код – 043Е и так далее: д – 0434, и – 0438, р – 0440, о – 043Е, в – 0432, к – 043А, а – 0430.
- Установить Кириллица (дес.)
- К – 0202, о – 0238, д – 0228, и – 0232, р – 0240, о – 0238, в –0226, к – 0202, а –0224.

7. Открыть Word.

Используя окно «Вставка символа» выполнить задания: Закодировать слово **Forest**

а) Выбрать шрифт Courier New, кодировку ASCII(дес.) Ответ: **70 111 114 101 115 116**
б) Выбрать шрифт Courier New, кодировку Юникод(шест.) Ответ: **0046 006F 0072 0065 0073 0074**

в) Выбрать шрифт Times New Roman, кодировку Кирилица(дес.) Ответ: **70 111 114 101 115 116**

г) Выбрать шрифт Times New Roman, кодировку ASCII(дес.) Ответ: **70 111 114 101 115 116**

Вывод: _____

Выполнение лабораторной работы оформить в виде таблицы.

8. Буква Z имеет десятичный код 90, а z – 122. Записать слово «sport» в десятичном коде.

9. С помощью десятичных кодов зашифровано слово «info» 105 110 102 111. Записать последовательность десятичных кодов для этого же слова, но записанного заглавными буквами.

10. Буква Z имеет десятичный код 90, а z – 122. Записать слово «forma» в десятичном коде.

11. С помощью десятичных кодов зашифровано слово «port» 112 111 114 116. Записать последовательность десятичных кодов для этого же слова, но записанного заглавными буквами. Ответ: **80 79 82 84**

Контрольные вопросы:

25. Правило цифрового представления символов
26. Чем отличаются стандарты для представления символов?
27. Какие кодировки символов вы знаете?
28. Назовите самый распространённый код для информационного обмена?
29. Назовите недостатки стандарта ISO-646?

Лабораторная работа № 11. Сжатие информации.

Цель работы: научиться сжимать информацию с помощью метода Хаффмана и метода RLE.

Методические указания:

Код Хаффмана

Определение 1: Пусть $A=\{a_1,a_2,...,a_n\}$ - алфавит из n различных символов, $W=\{w_1,w_2,...,w_n\}$ - соответствующий ему набор положительных целых весов. Тогда набор бинарных кодов $C=\{c_1,c_2,...,c_n\}$, такой что:

1) при i!=j c_i не является префиксом для c_j,
2) $\sum_{i=1}^{n} w_i|c_i|$ минимальна ($|c_i|$ длина кода c_i)

называется *минимально-избыточным префиксным кодом* или иначе *кодом Хаффмана*.

Замечания:

1. Свойство (1) называется *свойством префиксности*. Оно позволяет однозначно декодировать коды переменной длины.
2. Сумму в свойстве (2) можно трактовать как размер закодированных данных в битах. На практике это очень удобно, т.к. позволяет оценить степень сжатия не прибегая непосредственно к кодированию.
3. В дальнейшем, чтобы избежать недоразумений, под кодом будем понимать битовую строку определенной длины, а под минимально-избыточным кодом или кодом Хаффмана - множество кодов (битовых строк), соответствующих определенным символам и обладающих определенными свойствами.

Известно, что любому бинарному префиксному коду соответствует определенное бинарное дерево.

Определение 2: Бинарное дерево, соответствующее коду Хаффмана, будем называть *деревом Хаффмана*.

Задача построения кода Хаффмана равносильна задаче построения соответствующего ему дерева. Приведем общую схему построения дерева Хаффмана:

1. Составим список кодируемых символов (при этом будем рассматривать каждый символ как одноэлементное бинарное дерево, вес которого равен весу символа).
2. Из списка выберем 2 узла с наименьшим весом.
3. Сформируем новый узел и присоединим к нему, в качестве дочерних, два узла выбранных из списка. При этом вес сформированного узла положим равным сумме весов дочерних узлов.
4. Добавим сформированный узел к списку.
5. Если в списке больше одного узла, то повторить 2-5.

Приведем пример: построим дерево Хаффмана для сообщения S="A H F B H C E H E H C E A H D C E E H H H C H H H D E G H G G E H C H H".

Для начала введем несколько обозначений:

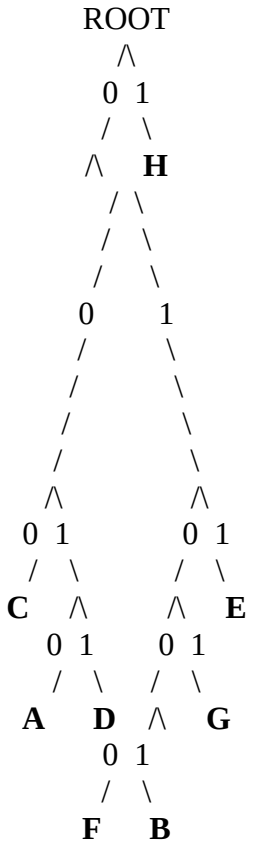
1. Символы кодируемого алфавита будем выделять жирным шрифтом: **A**, **B**, **C**.
2. Веса узлов будем обозначать нижними индексами: $\mathbf{A}_5$, $\mathbf{B}_3$, $\mathbf{C}_7$.
3. Составные узлы будем заключать в скобки: $((\mathbf{A}_5+\mathbf{B}_3)_8+\mathbf{C}_7)_{15}$.

Итак, в нашем случае A={**A**, **B**, **C**, **D**, **E**, **F**, **G**, **H**}, W={2, 1, 5, 2, 7, 1, 3, 15}.

1. $\mathbf{A}_2\ \mathbf{B}_1\ \mathbf{C}_5\ \mathbf{D}_2\ \mathbf{E}_7\ \mathbf{F}_1\ \mathbf{G}_3\ \mathbf{H}_{15}$
2. $\mathbf{A}_2\ \mathbf{C}_5\ \mathbf{D}_2\ \mathbf{E}_7\ \mathbf{G}_3\ \mathbf{H}_{15}\ (\mathbf{F}_1+\mathbf{B}_1)_2$
3. $\mathbf{C}_5\ \mathbf{E}_7\ \mathbf{G}_3\ \mathbf{H}_{15}\ (\mathbf{F}_1+\mathbf{B}_1)_2\ (\mathbf{A}_2+\mathbf{D}_2)_4$
4. $\mathbf{C}_5\ \mathbf{E}_7\ \mathbf{H}_{15}\ (\mathbf{A}_2+\mathbf{D}_2)_4\ ((\mathbf{F}_1+\mathbf{B}_1)_2+\mathbf{G}_3)_5$
5. $\mathbf{E}_7\ \mathbf{H}_{15}\ ((\mathbf{F}_1+\mathbf{B}_1)_2+\mathbf{G}_3)_5\ (\mathbf{C}_5+(\mathbf{A}_2+\mathbf{D}_2)_4)_9$

6. $\mathbf{H}_{15}$ $(\mathbf{C}_5+(\mathbf{A}_2+\mathbf{D}_2)_4)_9$ $(((\mathbf{F}_1+\mathbf{B}_1)_2+\mathbf{G}_3)_5+\mathbf{E}_7)_{12}$
7. $\mathbf{H}_{15}$ $((\mathbf{C}_5+(\mathbf{A}_2+\mathbf{D}_2)_4)_9+(((\mathbf{F}_1+\mathbf{B}_1)_2+\mathbf{G}_3)_5+\mathbf{E}_7)_{12})_{21}$
8. $(((\mathbf{C}_5+(\mathbf{A}_2+\mathbf{D}_2)_4)_9+(((\mathbf{F}_1+\mathbf{B}_1)_2+\mathbf{G}_3)_5+\mathbf{E}_7)_{12})_{21}+\mathbf{H}_{15})_{36}$

В списке, как и требовалось, остался всего один узел. Дерево Хаффмана построено. Теперь запишем его в более привычном для нас виде.



Листовые узлы дерева Хаффмана соответствуют символам кодируемого алфавита. Глубина листовых узлов равна длине кода соответствующих символов.

Путь от корня дерева к листовому узлу можно представить в виде битовой строки, в которой "0" соответствует выбору левого поддерева, а "1" - правого. Используя этот механизм, мы без труда можем присвоить коды всем символам кодируемого алфавита. Выпишем, к примеру, коды для всех символов в нашем примере:

$\mathbf{A}=0010_{bin}$	$\mathbf{C}=000_{bin}$	$\mathbf{E}=011_{bin}$	$\mathbf{G}=0101_{bin}$
$\mathbf{B}=01001_{bin}$	$\mathbf{D}=0011_{bin}$	$\mathbf{F}=01000_{bin}$	$\mathbf{H}=1_{bin}$

Теперь у нас есть все необходимое для того чтобы закодировать сообщение S. Достаточно просто заменить каждый символ соответствующим ему кодом:

$S^{/}$="0010 1 01000 01001 1 000 011 1 011 1 000 011 0010 1 0011 000 011 011 1 1 1 000 1 1 1 0011 011 0101 1 0101 0101 011 1 000 1 1".

Оценим теперь степень сжатия. В исходном сообщении S было 36 символов, на каждый из которых отводилось по $[\log_2|A|]=3$ бита (здесь и далее будем понимать квадратные скобки [] как целую часть, округленную в положительную сторону, т.е. [3,018]=4). Таким образом, размер S равен 36*3=108 бит

Размер закодированного сообщения $S^{/}$ можно получить воспользовавшись замечанием 2 к определению 1, или непосредственно, подсчитав количество бит в $S^{/}$. И в том и другом случае мы получим 89 бит.

Итак, нам удалось сжать 108 в 89 бит.

Теперь декодируем сообщение $S^{/}$. Начиная с корня дерева будем двигаться вниз, выбирая левое поддерево, если очередной бит в потоке равен "0", и правое - если "1". Дойдя до листового узла мы декодируем соответствующий ему символ.

Ясно, что следуя этому алгоритму мы в точности получим исходное сообщение S.

Метод RLE.

Наиболее известный простой подход и алгоритм сжатия информации обратимым путем - это кодирование серий последовательностей (Run Length Encoding - RLE). Суть методов данного подхода состоит в замене цепочек или серий повторяющихся байтов или их последовательностей на один кодирующий байт и счетчик числа их повторений. Проблема всех аналогичных методов заключается лишь в определении способа, при помощи которого распаковывающий алгоритм мог бы отличить в результирующем потоке байтов кодированную серию от других - некодированных последовательностей байтов. Решение проблемы достигается обычно простановкой меток в начале кодированных цепочек. Такими метками могут быть, например, характерные значения битов в первом байте кодированной серии, значения первого байта кодированной серии и т.п. Данные методы, как правило, достаточно эффективны для сжатия растровых графических изображений (BMP, PCX, TIF, GIF), т.к. последние содержат достаточно много длинных серий повторяющихся последовательностей байтов. Недостатком метода RLE является достаточно низкая степень сжатия или стоимость кодирования файлов с малым числом серий и, что еще хуже - с малым числом повторяющихся байтов в сериях.

Задание

1. Сжатие методом Хаффмана

«КАКАЯ ЗИМА ЗОЛОТАЯ!
КАК БУДТО ИЗ ДЕТСКИХ ВРЕМЕН...
НЕ НАДО НИ СОЛНЦА, НИ МАЯ –
ПУСТЬ ДЛИТСЯ ТОРЖЕСТВЕНИЫЙ СОН.

ПУСТЬ Я В ЭТОМ СНЕ ПОЗАБУДУ
КОГДА-ТО МАНИВШИЙ ОГОНЬ,
И ЛЕТО ПРЕДАМ, КАК ИУДА,
ЗА ТРИДЦАТЬ СНЕЖИНОК В ЛАДОНЬ.

ЗАТЕМ, ЧТО И Я ХОЛОДЕЮ,
ТЕПЛО УЖЕ СТРАШНО ПРИНЯТЬ:
Я СЛИШКОМ ДАВНО НЕ УМЕЮ
НИ ТЛЕТЬ, НИ ГОРЕТЬ, НИ СЖИГАТЬ…

ВСЕ ЧАЩЕ, ВСЕ ДОЛЬШЕ НЕМЕЮ:
К ЗИМЕ УЖЕ ДЕЛО, К ЗИМЕ...
И ТОЛЬКО ТОГО ОТОГРЕЮ,
КОМУ ХОЛОДНЕЕ, ЧЕМ МНЕ»

2. С помощью сжатия по методу RLE.

1 последовательность:

SSSSOOOEEERROOOAAYYYYYDDDDOEUUUUUWWWWJJJORRUUUUUUUUUUXXXKHHHHHHMMMMMMGGGLLLLLLLJJJJ

2 последовательность:

FFFFFFFFKKKKKSSSSUURERRRRRRRRRPPPPPPPPDDDDKKKKKKGLDDDDDDDDDKKKKKKKKGGGGMGMMMM

3. Создайте презентацию по теме «Алгоритмы сжатия изображений». Используйте ресурсы Интернет.

Контрольные вопросы:

30. Что такое код Хаффмана?
31. Что называется деревом Хаффмана?
32. Как происходит сжатие методом Хаффмана?
33. Как происходит сжатие по методу RLE?
34. Назовите расширения растровых графических изображений?

Компьютерное представление видеоинформации.

Цель: научиться кодировать растровые графические файлы; научиться измерять информационный объем графических файлов.

Методические указания.

Графическая информация на экране дисплея представляется в виде изображения, которое формируется из точек (пикселей). Всмотритесь в газетную фотографию, и вы увидите, что она тоже состоит из мельчайших точек. Если это только чёрные и белые точки, то каждую из них можно закодировать 1 битом. Но если на фотографии оттенки, то два бита позволяет закодировать 4 оттенка точек: 00 - белый цвет, 01 - светло-серый, 10 - тёмно-серый, 11 - чёрный. Три бита позволяют закодировать 8 оттенков и т.д.

Количество бит, необходимое для кодирования одного оттенка цвета, называется глубиной цвета.

$K=2^G$, где K – количество оттенков, G – глубина цвета в битах.

В современных компьютерах разрешающая способность (количество точек на экране), а также количество цветов зависит от видеоадаптера и может изменяться программно.

Цветные изображения могут иметь различные режимы: 16 цветов, 256 цветов, 65536 цветов (high color), 16777216 цветов (true color). На одну точку для режима high color необходимо 16 бит или 2 байта.

Наиболее распространённой разрешающей способностью экрана является разрешение 800 на 600 точек, т.е. 480000 точек. Рассчитаем необходимый для режима high color объём видеопамяти: 2 байт *480000=960000 байт.

Для измерения объёма информации используются и более крупные единицы:

1 Кбайт (один килобайт) = 2^{10} байт=1024 байт
1 Мбайт (один мегабайт) = 2^{20} байт=1048576 байт
1 Гбайт (один гигабайт) = 2^{30} байт=около1млрд. байт

Следовательно, 960000 байт приблизительно равно 937,5 Кбайт. Если человек говорит по восемь часов в день без перерыва, то за 70 лет жизни он наговорит около 10 гигабайт информации (это 5 миллионов страниц - стопка бумаги высотой 500 метров).

Скорость передачи информации - это количество битов, передаваемых в 1 секунду. Скорость передачи 1 бит в 1 секунду называется 1 бод.

1 Кбод = 1024 бит/сек; 1 Мбод = 1024 Кбод; 1 Гбод = 1024 Мбод

В видеопамяти компьютера хранится битовая карта, являющаяся двоичным кодом изображения, откуда она считывается процессором (не реже 50 раз в секунду) и отображается на экран.

Задания

1. Известно, что видеопамять компьютера имеет объем 512 Кбайт. Разрешающая способность экрана 640 на 200. Сколько страниц экрана одновременно разместится в видеопамяти при палитре: а) из 8 цветов, б) 16 цветов; в) 256 цветов?

2. Сколько бит требуется, чтобы закодировать информацию о 130 оттенках?

3. Подумайте, как уплотнить информацию о рисунке при его записи в файл, если известно, что: а) в рисунке одновременно содержится только 16 цветовых оттенков из 138 возможных; б) в рисунке присутствуют все 130 оттенков одновременно, но количество точек, закрашенных разными оттенками, сильно различаются.

4. Найдите в сети Интернет информацию на тему «Цветовые модели HSB, RGB, CMYK» и создайте на эту тему презентацию. В ней отобразите положительные и отрицательные стороны каждой цветовой модели, принцип ее функционирования и применение.

5. В приложении «Точечный рисунок» создайте файл размером (по вариантам):

А) 200*300, (№ по списку 1, 8, 15, 22, 29)

Б) 590*350, (№ по списку 2, 9, 16, 23, 30)

В) 478*472, (№ по списку 3, 10, 17, 24, 31)

Г)190*367, (№ по списку 4, 11, 18, 25, 32)

Д) 288*577; (№ по списку 5, 12, 19, 26, 33)

Е) 100*466, (№ по списку 5, 13, 20, 27, 34)

Ж) 390*277. (№ по списку 6, 14, 21, 28)

Сохраните его под следующими расширениями:

- монохромный рисунок,
- 16-цветный рисунок,
- 256-цветный рисунок,
- 24-битный рисунок,
- формат JPG.

Используя информацию о размере каждого из полученных файлов, вычислите количество используемых цветов в каждом из файлов, проверьте с полученным на практике. Объясните, почему формула расчета количества цветов не подходит для формата JPG. Для этого воспользуйтесь информацией из сети Интернет.

6. На бумаге в клетку (или в приложении Excel) нарисуйте произвольный рисунок 10*10 клеток. Закодируйте его двоичным кодом (закрашена клетка – 1, не закрашена - 0). Полученный код отдайте одногруппнику для раскодирования и получения изображения.

Контрольные вопросы:

35. В виде чего представлена графическая информация на экране?
36. Что называется глубиной цвета?
37. Что называется разрешающей способностью?
38. Что называется скорость передачи информации?
39. Что считается процессором?

Лабораторная работа № 12 Разработка системы передачи информации на базе Packet Tracer Cisco Systems

Цель работы. Изучить основные функциональные возможности программного сетевого эмулятора Packet Tracer Cisco Systems.

Введение

Packet Tracer является интегрированной средой моделирования, визуализации, совместной работы и оценки состояния окружающей среды. Packet Tracer помогает студенту и преподавателю создавать сетевые модели, осуществлять визуализацию и анимацию передачи информации в сети. Как и любое моделирование, Packet Tracer опирается на упрощенные модели сетевых устройств и протоколов. Реальные компьютерные сети остаются эталоном для понимания поведения сети и развития навыков для их построения. Packet Tracer был создан, чтобы помочь решить проблему обеспечения доступа к сетевому оборудованию для студентов и преподавателей.

Наименование	*Описание*
Протоколы	**LAN**: Ethernet (including CSMA/CD*), 802.11 a/b/g/n wireless*, PPPOE **Switching**: VLANs, 802.1q, trunking, VTP, DTP, STP*, RSTP*, multilayer switching*, Etherchannel, LACP, PAgP **TCP/IP**: HTTP, HTTPS, DHCP, DHCPv6, Telnet, SSH, TFTP, DNS, TCP*, UDP, IPv4*, IPv6*, ICMP, ICMPv6, ARP, IPv6 ND, FTP, SMTP, POP3, VOIP(H.323) **Routing**: static, default, RIPv1, RIPv2, EIGRP, single-area OSPF, multi-area OSPF, BGP, inter-VLAN routing, redistribution **Other**: ACLs (standard, extended, and named), CDP, NAT (static, dynamic, inside/outside, and overload), NATv6 **WAN**: HDLC, SLARP, PPP*, and Frame Relay* **Security**: IPsec, GRE, ISAKMP, NTP, AAA, RADIUS, TACACS, SNMP, SSH, SYSLOG, CBAC, Zone-based policy firewall, IPS **QoS**: Layer 2 QoS, Layer 3 Diffserv QoS, FIFO Hardware queues, Priority Queuing, Custom Queuing, Weighted Fair Queuing, MQC, NBAR* *** обозначает наложенные функциональные ограничения**
Логическое пространство	Создание сетевой топологии Устройства: маршрутизаторы, коммутаторы, хранилища(Server, Desktop and Laptop), хабы, мосты, беспроводные точки доступа, беспроводные маршрутизаторы и DSL/cable модемы Соединение устройств осуществляется с использованием медных, оптоволоконных, коаксиальных кабелей.
Физическое пространство	Поддерживает следующие виды: иерархия устройств, коммутационные шкафы, здания, города Также поддерживается отображение допустимой длины кабелей в сети

	Ethernet, масштабирование созданных пользователем графиков.
	Обмен данными происходит в режиме реального времени.
Режим реального времени	Настраиваемая конфигурация: DHCP, DNS, HTTP, TFTP, Syslog, AAA, and NTP servers
	Анимация передачи пакетов
	Лист событий
Режим симуляции	Имеется широкий выбор протоколов модели OSI.
	Пользователь имеет возможность создания сценария передачи пакетов.

Логическое пространство

Для того, чтобы расположить устройство, необходимо выбрать его из меню и перетащить на главную панель.

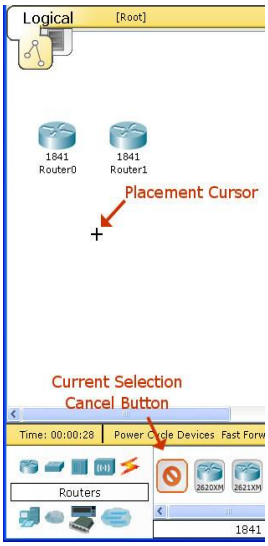


Рис.1. Выбор устройства

Большинство из устройств в Packet Tracer имеют модули расширения, необходимые для подключения дополнительных портов. Добавление модулей осуществляется в панели настройки устройства. При подключении нового модуля устройство должно быть отключено от электросети.

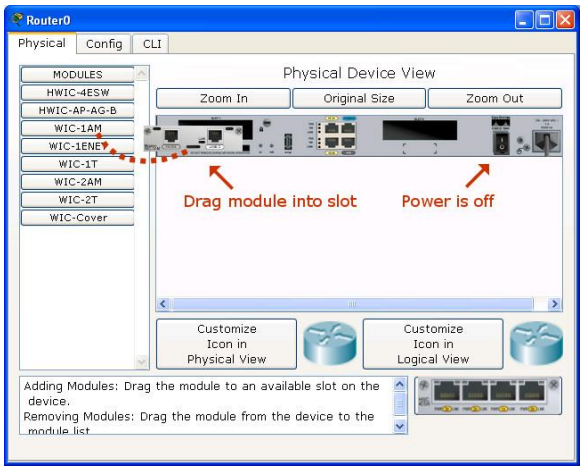


Рис.2. Добавление модулей расширения

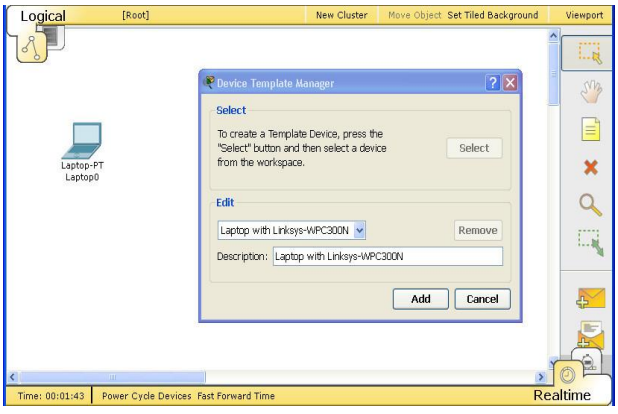


Рис.3. Создание шаблона устройства.

Packet Tracer предоставляет возможность создания шаблонов устройств. Для создания шаблона необходимо выбрать устройство, добавить необходимые модули расширения, затем перейти в **Custom Devices Dialog.** Затем добавить описание выбранного устройства, нажав на **Select**. Добавить новое созданное пользователем устройство можно через **Custom Made Devices.**

Для соединения устройств между собой необходимо выбрать подходящие кабели, расположенных на панели **Connections**. Затем нужно щелкнуть правой кнопкой мыши по одному из устройств и выбрать порт подключения. Аналогичные дествия выполнить для второго устройства.

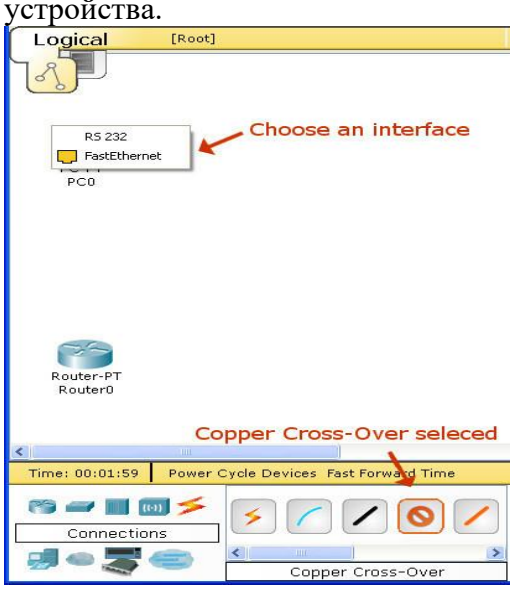


Рис. 4. Создание соединений.

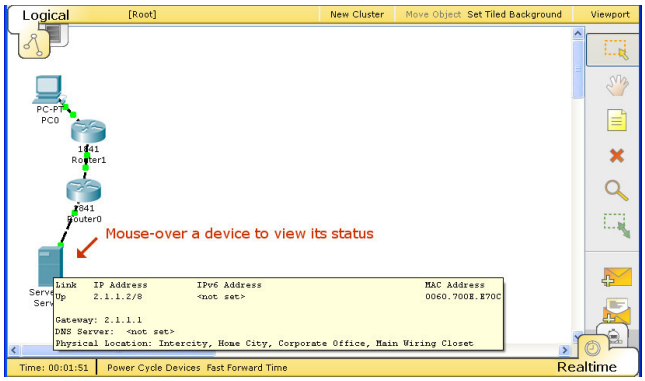


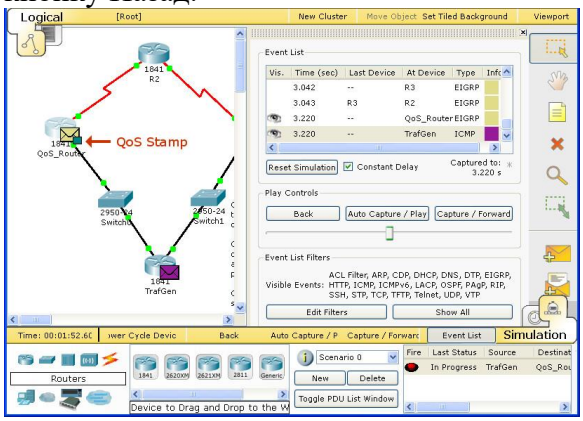
Рис. 5. Функционирование сети в режиме реального времени.

Режим реального времени

В режиме реального времени, сеть всегда работает независимо от действий пользователя. Конфигурирование сети осуществляется в реальном времени. При просмотре статистики сети, они отображаются в режиме реального времени, как показано на панели инструментов. В дополнение к использованию Cisco IOS для настройки и диагностики сети, вы можете использовать Add Simple PDU и User Created PDU List для наглядной отправки пакета.

Режим симуляции

В режиме симуляции, вы можете смотреть свои сети работать в более медленном темпе, исследуя пути, по которым пересылаются пакеты. При переключении в режим моделирования, появится специальная панель. Вы можете графически просматривать распространение пакетов по сети, нажав на кнопку Add Simple PDU. Имеется возможность контроля скорости моделирования с использованием кнопки Speed Slider. Также можно просматривать предыдущие события, нажав на кнопку Назад.



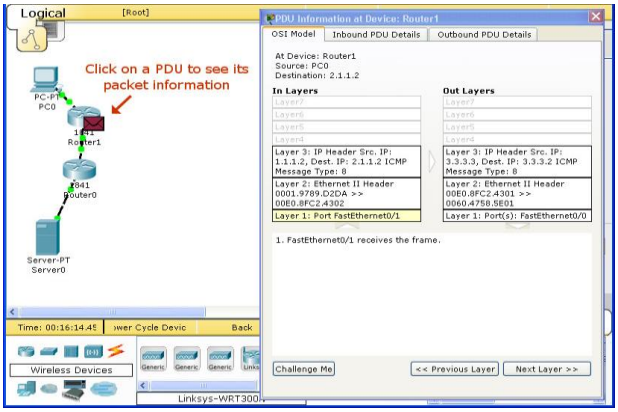


Рис.6. Режим моделирования

Рис.7. Информация о пакете

Во время моделирования можно кликнуть на пересылаемом пакете и получить о нем подробную информацию.

Физическое пространство

Целью физической рабочей области является обзор физических аспектов логической топологии сети. Это дает ощущение масштаба и размещения (как ваша сеть может выглядеть в реальной среде).

Физические рабочая область разделена на четыре слоя, чтобы отразить физический масштаб четыре среды: междугородная, город, дом, и коммутационного узла. Междугородный является крупнейшим окружающей среды. Он может содержать много городов. Каждый город может содержать множество зданий. Наконец, каждое здание может содержать множество шкафов проводки. Распределительный шкаф обеспечивает вид, который отличается от трех других видов. Здесь вы можете увидеть устройства, которые были созданы в логических Workspace; позиционируется в области сетевых технологий стойки и на столах. Три других слоя обеспечивают просмотр миниатюр их макетов. Это по умолчанию расположение в физической рабочей области, но устройства в шкафу могут быть перемещены в любой из слоев.

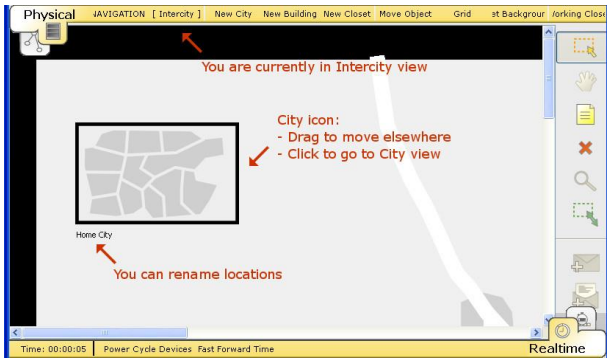


Рис.8. Междугородний масштаб

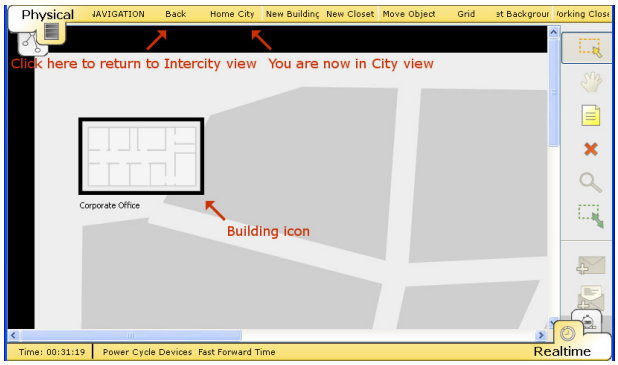


Рис.9. Масштаб здания

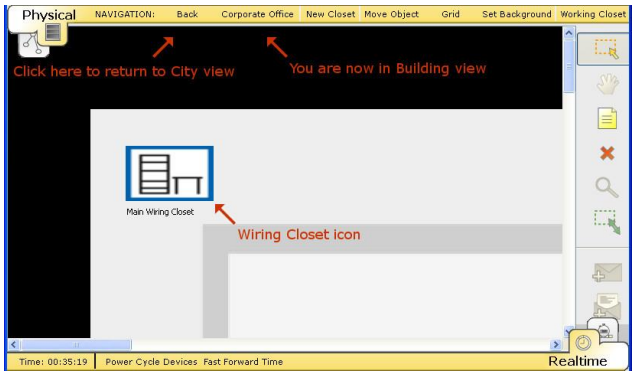


Рис.10. Коммутационный шкаф

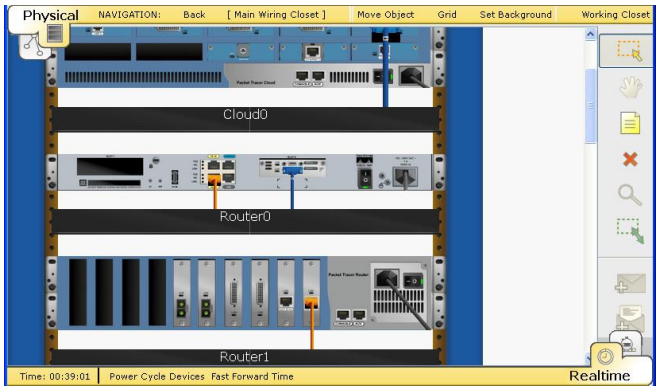


Рис.11. Устройства, размещенные в коммутационном шкафу

Лабораторная работа № 13 Топология и построение сети в Packet Tracer

Цель работы. Ознакомится с архитектурой стека протоколов TCP/IP с использованием программного сетевого эмулятора Packet Tracer Cisco Systems.

Задание. Построить сеть с использованием маршрутизаторов, коммутаторов и оконечного оборудования.

Краткие теоретические сведения

Открытая система(OSI) — это стандартизированный набор протоколов и спецификаций, который гарантирует возможность взаимодействия оборудования различных производителей. Она реализуется набором модулей, каждый из которых решает простую задачу внутри элемента сети. Каждый из модулей связан с одним или несколькими другими модулями. Решение сложной задачи подразумевает определенный порядок следования решения простых задач, при котором образуется многоуровневая иерархическая структура на рис. 1. Это позволяет любым двум различным системам связываться независимо от их основной архитектуры.

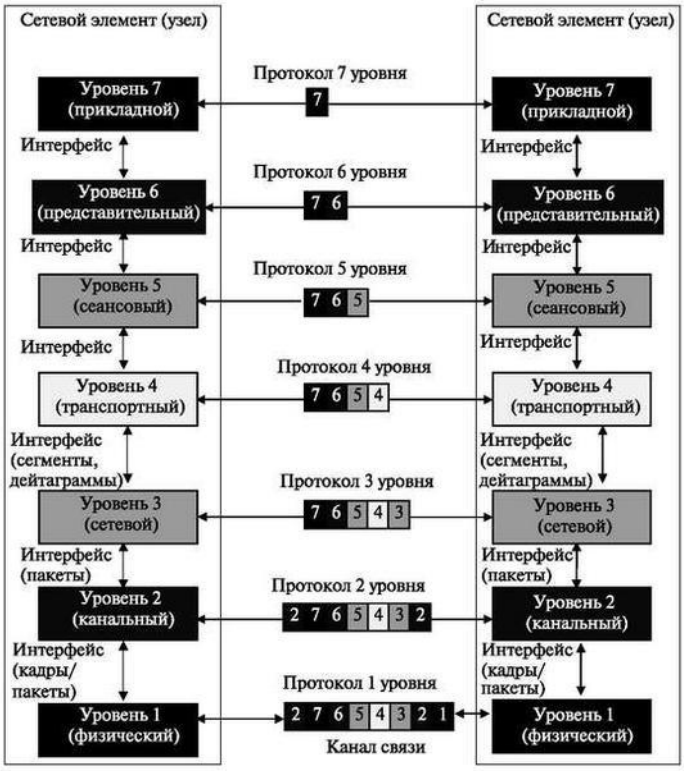


Рис. 1. Модель взаимодействия открытых систем OSI

Модель OSI составлена из семи упорядоченных уровней: физического (уровень 1), звена передачи данных (уровень 2), сетевого (уровень 3), транспортного (уровень 4), сеансового (уровень 5), представления (уровень 6) и прикладного (уровень 7).

Обмен информацией между модулями происходит на основе определенных соглашений, которые называются интерфейсом. При передаче сообщения модуль верхнего уровня решает свою часть задачи, а результат, понятный только ему, оформляет в виде дополнительного поля к исходному сообщению (заголовка) и передает измененное сообщение на дообслуживание в нижележащий уровень. Этот процесс называется инкапсуляцией.

Стек протоколов Интернета

Стек протоколов сети Интернет был разработан до модели OSI. Поэтому уровни в стеке протоколов Интернета не соответствуют аналогичным уровням в модели OSI. Стек протоколов Интернета состоит из пяти уровней: физического, звена передачи данных, сети, транспортного и прикладного. Первые четыре уровня обеспечивают физические стандарты, сетевой интерфейс, межсетевое взаимодействие и транспортные функции, которые соответствуют первым четырем уровням модели OSI. Три самых верхних уровня в модели OSI представлены в стеке протоколов Интернета единственным уровнем, называемым прикладным уровнем рис. 2.

Рис. 2. Стек протоколов Интернета по сравнению с OSI

Стек базовых протоколов Интернета — иерархический, составленный из диалоговых модулей, каждый из которых обеспечивает заданные функциональные возможности; но эти модули не обязательно взаимозависимые. В отличие от модели OSI, где определяется строго, какие функции принадлежат каждому из ее уровней, уровни набора протокола TCP/IP содержат относительно независимые протоколы, которые могут быть смешаны и согласованы в зависимости от потребностей системы. Термин иерархический означает, что каждый верхний протокол уровня поддерживается соответственно одним или более протоколами нижнего уровня.

На транспортном уровне стек определяет два протокола: протокол управления передачей (TCP) и протокол пользовательских дейтаграмм (UDP). На сетевом уровне — главный протокол межсетевого взаимодействия (IP), хотя на этом уровне используются некоторые другие протоколы, о которых будет сказано ниже.

Адресация узлов в IP-сетях

В сетях TCP/IP принято различать адреса сетевых узлов трех уровней

- физический (или локальный) адрес узла (MAC-адрес сетевого адаптера или порта маршрутизатора); эти адреса назначаются производителями сетевого оборудования;
- IP-адрес узла (например, 192.168.0.1), данные адреса назначаются сетевыми администраторами или Интернет-провайдерами;
- символьное имя (например, www.microsoft.com); эти имена также назначаются сетевыми администраторами компаний или Интернет-провайдерами.

Рассмотрим подробнее IP-адресацию.

Компьютеры или другие сложные сетевые устройства, подсоединенные к нескольким физическим сетям, имеют несколько IP-адресов — по одному на каждый сетевой интерфейс. Схема адресации

позволяет проводить единичную, широковещательную и групповую адресацию. Таким образом, выделяют 3 типа IP-адресов.

1. Unicast-адрес (единичная адресация конкретному узлу) — используется в коммуникациях "один-к-одному".
2. Broadcast-адрес (широковещательный адрес, относящийся ко всем адресам подсети) — используется в коммуникациях "один-ко-всем". В этих адресах поле идентификатора устройства заполнено единицами. IP-адресация допускает широковещательную передачу, но не гарантирует ее — эта возможность зависит от конкретной физической сети. Например, в сетях Ethernet широковещательная передача выполняется с той же эффективностью, что и обычная передача данных, но есть сети, которые вообще не поддерживают такой тип передачи или поддерживают весьма ограничено.
3. Multicast-адрес (групповой адрес для многоадресной отправки пакетов) — используется в коммуникациях "один-ко-многим". Поддержка групповой адресации используется во многих приложениях, например, приложениях интерактивных конференций. Для групповой передачи рабочие станции и маршрутизаторы используют протокол IGMP, который предоставляет информацию о принадлежности устройств определенным группам.

Класс сети	Наименьший идентификатор сети	Наибольший идентификатор сети	Количество сетей
Класс A	1.0.0.0	126.0.0.0	126
Класс B	128.0.0.0	191.255.0.0	16384
Класс C	192.0.0.0	223.255.255.0	2097152

Сетевое оборудование

Мост (bridge), как и репитер, может соединять сегменты или локальные сети рабочих групп. Однако, в отличие от репитера, мост также служит для разбиения сети, что помогает изолировать трафик или отдельные проблемы. Например, если трафик одного-двух компьютеров или одного отдела “затопляет” сеть пакетами, уменьшая ее производительность в целом, мост изолирует эти компьютеры или этот отдел. Мосты обычно решают следующие задачи. Увеличивают размер сети. Увеличивают максимальное количество компьютеров в сети.

В среде, объединяющей несколько сетевых сегментов с различными протоколами и архитектурами, мосты не всегда гарантируют быструю связь между всеми сегментами. Для такой сложной сети необходимо устройство, которое не только знает адрес каждого сегмента, но и определяет наилучший маршрут для передачи данных и фильтрует широковещательные сообщения. Такое устройство называется **маршрутизатором**.

Маршрутизаторы (routers) работают на Сетевом уровне модели OSI. Это значит, что они могут переадресовывать и маршрутизировать пакеты через множество сетей, обмениваясь информацией (которая зависит от протокола) между раздельными сетями. Маршрутизаторы считывают в пакете адресную информацию сложной сети и, поскольку они функционируют на более высоком по сравнению с мостами уровне модели OSI, имеют доступ к дополнительным данным. Маршрутизаторы могут выполнять следующие функции мостов: фильтровать и изолировать трафик; соединять сегменты сети.

Устройства **Switch** - коммутатор. Это оборудование относится к активному сетевому оборудованию, и служит для обработки пакетов в сети. Свитч (то же самое, что переключатель, мост, switch, bridge) - устройство, служащее для разделения сети на отдельные сегменты, которые могут содержать хабы и сетевые карты. Свитчи являются устройствами 2-го уровня, т.е. содержат в себе порты - устройства 1-го уровня для работы с сигналами, но помимо того, работают с содержимым сетевых пакетов - читают поле физического адреса назначения (MAC) пакета,

пришедшего на один из портов, и в зависимости от его значения и таблицы MAC-адрес - порт "ретранслируют" пакет на другой порт (или не ретранслируют).

ВЫПОЛНЕНИЕ

Ниже на рис.3 приведена спроектированная сеть, которая включает в себя следующее оборудование:

- Маршрутизаторы;
- Коммутаторы;
- ПК;
- IP-телефоны;
- Сервер.

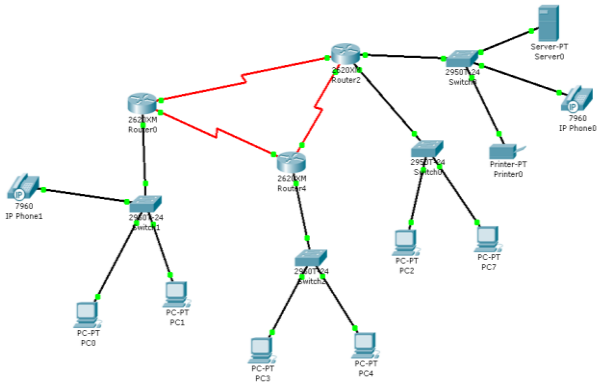


Рис. 3. Результат построения сети

Маршрутизаторы соединяются между собой при помощи DCE – кабеля. В данной сети маршрутизаторы используют RIP – протокол для осуществления передачи данных между различными подсетями(рис. 4).

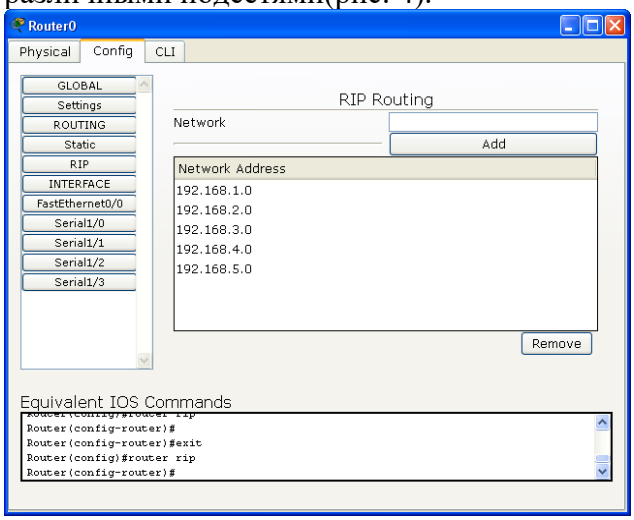


Рис. 4. Список подсетей

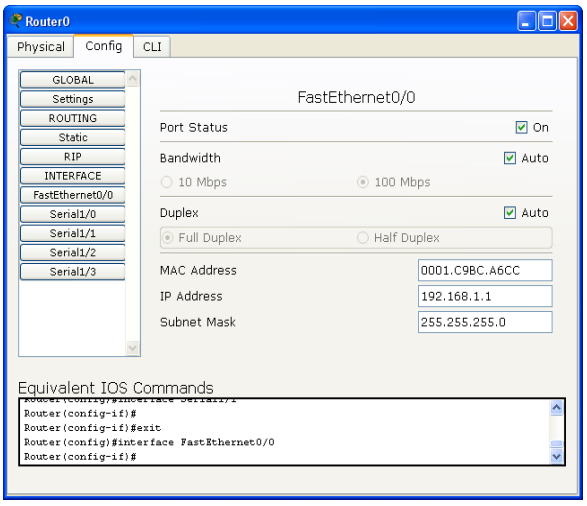
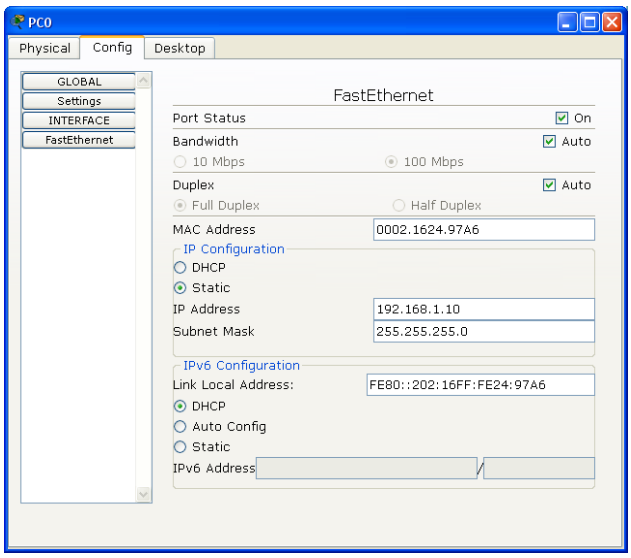


Рис. 5. Назначение IP-адреса маршрутизатору



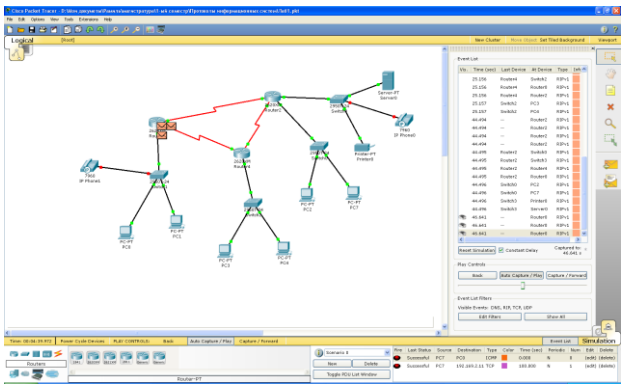


Рис.6. Назначение IP-адреса ПК

При этом, IP-адреса назначаются статически для маршрутизаторов и оконечного оборудования(рис. 5, 6).

Лабораторная работа №14 «Анализ протоколов уровня приложения и транспорта»

Цель работы. Провести анализ работы протоколов уровня приложений и транспорта с использованием программного сетевого эмулятора Packet Tracer Cisco Systems.

Краткие теоретические сведения

Модель TCP/IP

Семейство протоколов TCP/IP основано на четырехуровневой эталонной модели. Все протоколы, входящие в семейство протоколов TCP/IP, расположены на трех верхних уровнях этой модели. Каждый уровень модели TCP/IP соответствует одному или нескольким уровням семиуровневой эталонной модели OSI (Open Systems Interconnection — взаимодействие открытых систем), предложенной ISO — международной организацией по стандартам (International Standards Organization).

Типы служб и протоколов, используемых на каждом уровне модели TCP/IP, более подробно описаны в следующей таблице.

Уровень	Описание	Протоколы
Приложение	Определяет прикладные протоколы TCP/IP и интерфейс программ со службами транспортного уровня, необходимый для использования сети.	HTTP, Telnet, FTP, TFTP, SNMP, DNS, SMTP, X Windows, другие прикладные протоколы
Транспортный	Обеспечивает управление сеансами связи между компьютерами. Определяет уровень служб и состояние подключения, используемые при транспортировке данных.	TCP, UDP
Интернет	Упаковывает данные в IP-датаграммы, содержащие информацию об адресах источника и приемника, которая используется для перенаправления датаграмм от узла к узлу и по сетям. Выполняет маршрутизацию IP-датаграмм.	IP, ICMP, ARP, RARP
Сетевого интерфейса	Определяет средства и принципы физической передачи данных по сети, включая преобразование битов данных в электрические или другие сигналы аппаратными устройствами, непосредственно подключенными к среде передачи, такой как коаксиальный кабель, оптоволокно или витая пара.	Ethernet, Token Ring, FDDI, X.25, Frame Relay, RS-232, v.35

Протоколы уровня приложений

SMTP (Simple Mail Transfer Protocol) – простой почтовый протокол. Он поддерживает передачу почтовых электронных сообщений по сети Интернет. Протокол называется простым, потому что обеспечивает передачу информации пользователям, готовым к немедленной доставке. Передача осуществляется в режиме 7-битовых слов. Он требует наличия программ перехода от принятого в большинстве программ формата с 8-разрядными словами к формату с 7-разрядными словами. Система поддерживает:

- посылку одиночных сообщений одному или более получателям;
- посылку сообщений, включающих в себя текст, голосовые сообщения, видео или графические материалы.

Протокол передачи файлов (FTP — File Transfer Protocol) используется для передачи файлов от одного компьютера к другому. Обеспечивает просмотр каталогов удаленного компьютера,

копирование, удаление и пересылку файлов. FTP отличается от других протоколов тем, что устанавливает два соединения между хостами. Одно используется для передачи информации, а другое — для управления передачей.

DNS (Domain Name System) – служба доменных имен. Она осуществляет присвоение уникальных имен всем пользователям и узлам сети Интернет и устанавливает логическую связь с их сетевыми адресами. Доменное имя представляется иерархической структурой, имеющей несколько уровней. Типовые имена доменов верхнего уровня закреплены следующим образом:

.com – коммерческие организации;

.gov – правительственные учреждения;

.org – некоммерческие организации;

.net — центры поддержки сети;

.int – международные организации;

.mil – военные структуры.

SNMP (Simple Network Management Protocol) — простой протокол управления сетью. Он обеспечивает набор фундаментальных действий по наблюдению и обслуживанию Интернета. Протокол разработан так, чтобы он мог контролировать устройства, созданные различными изготовителями и установленные на различных физических сетях. Другими словами, SNMP освобождает задачи управления от учета физических характеристик управляемых устройств и от основной технологии организации сети.

Сетевая файловая система (NFS — Network File System). Это один из многих протоколов (например, на рисунке показан еще один протокол RPC –Remote Procedure Call – вызов удаленной процедуры), который позволяет использование файлов, содержащих процедуры управления и периферии в другом компьютере.

Тривиальный (простейший) протокол передачи файлов TFTP (Trivial File Transfer Protocol). Используется в простых случаях при начальной загрузке рабочих станций или загрузке маршрутизаторов, не имеющих внешней памяти.

Протокол передачи гипертекста (HTTP — Hyper Text Transfer Protocol) — транспортный протокол, который применяется в Интернете при обмене документами, представленными на языке описания гипертекстовых документов.

Язык разметки гипертекста (HTML — Hyper Text Markup Language). Является одним из главных языков, используемых в сети WWW.

Протоколы уровня транспорта

Протокол управления передачей TCP (Transmission Control Protocol) является обязательным стандартом TCP/IP, который описан в документе RFC 793 «Transmission Control Protocol (TCP)» и предоставляет надежную службу доставки пакетов, ориентированную на установление соединения. Протокол TCP:

- гарантирует доставку IP-датаграмм;
- выполняет разбиение на сегменты и сборку больших блоков данных, отправляемых программами;
- обеспечивает доставку сегментов данных в нужном порядке;
- выполняет проверку целостности переданных данных с помощью контрольной суммы;
- посылает положительные подтверждения, если данные получены успешно. Используя избирательные подтверждения, можно также посылать отрицательные подтверждения для данных, которые не были получены;
- предлагает предпочтительный транспорт для программ, которым требуется надежная передача данных с установлением сеанса связи, например для баз данных «клиент-сервер» и программ электронной почты.

Как работает TCP

TCP основан на связи «точка-точка» между двумя узлами сети. TCP получает данные от программ и обрабатывает их как поток байтов. Байты группируются в сегменты, которым TCP присваивает последовательные номера, необходимые для правильной сборки сегментов на узле-приемнике. Чтобы два узла TCP могли обмениваться данными, им нужно сначала установить сеанс связи друг с другом. Сеанс TCP инициализируется с помощью процесса, называемого трехэтапным установлением связи. В этом процессе синхронизируются номера последовательности и передается управляющая информация, необходимая для установления виртуального соединения между узлами.
По завершении процесса трехэтапного установления связи начинается пересылка и подтверждение пакетов в последовательном порядке между этими узлами. Аналогичный процесс используется TCP перед прекращением соединения для того, чтобы убедиться, что оба узла закончили передачу и прием данных.

Протокол UDP

Протокол датаграмм пользователя UDP (User Datagram Protocol) является стандартом TCP/IP, описанным в документе RFC 768 «User Datagram Protocol (UDP)». UDP используется некоторыми программами вместо TCP для быстрой, простой, но ненадежной передачи данных между узлами TCP/IP.
UDP обеспечивает службу датаграмм, не ориентированную на установление соединения, что означает, что UDP не гарантирует ни доставку, ни правильность порядка доставки датаграмм. Узел-источник, которому требуется надежная связь, должен использовать либо протокол TCP, либо программу, которая сама обеспечивает подтверждения и следит за правильностью порядка датаграмм.

UDP	TCP
Служба, не ориентированная на установление соединения; сеанс связи между узлами не устанавливается.	Служба, ориентированная на установление соединения; между узлами устанавливается сеанс связи.
UDP не гарантирует и не подтверждает доставку данных, а также не гарантирует порядок их доставки.	TCP гарантирует доставку при помощи подтверждений и контроля порядка принимаемых данных.
Программы, использующие UDP, ответственны за обеспечение надежности передачи данных.	Программам, использующим TCP, гарантируется надежность передачи данных.
UDP — быстрый протокол с небольшими накладными расходами, поддерживающий связь «точка-точка» и «точка-многие точки».	TCP медленнее, требует больших накладных расходов и поддерживает только связь «точка-точка».

ВЫПОЛНЕНИЕ

Ниже на рис.1 приведена спроектированная сеть, которая включает в себя следующее оборудование:

- Маршрутизаторы;
- ПК;
- Сервер.

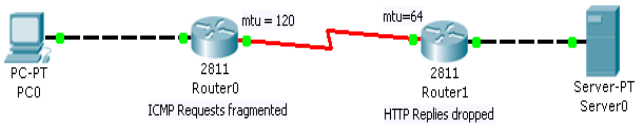


Рис. 1. Результат построения сети

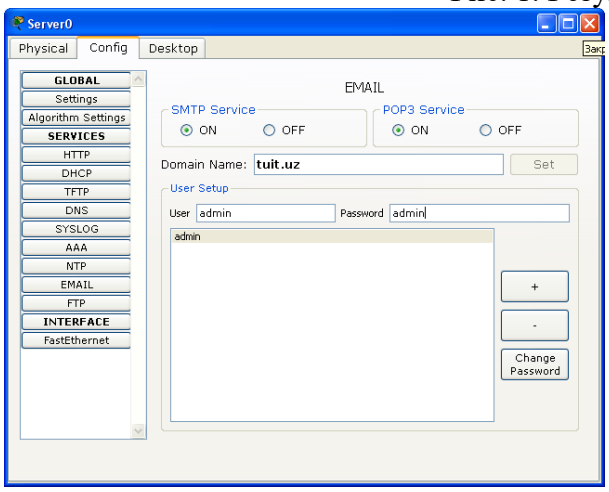


Рис.2. Настройка e-mail

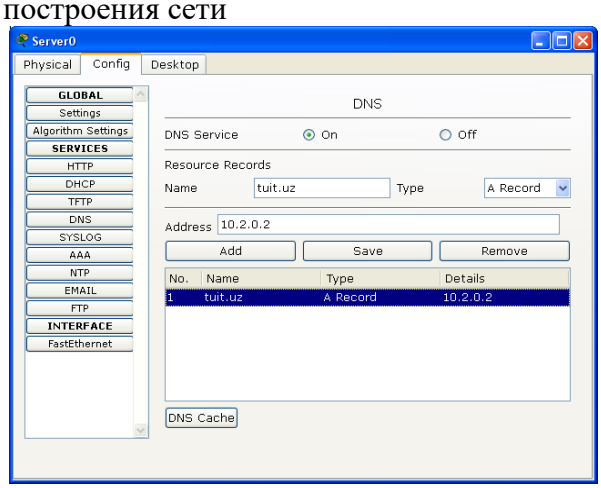


Рис.3. Настройка DNS-сервиса

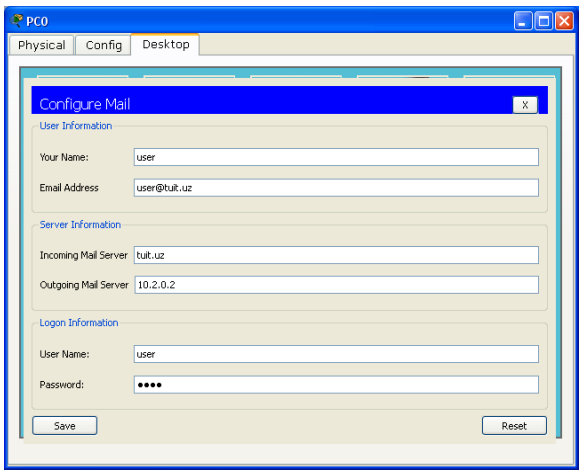


Рис.4. Настройка электронной почты

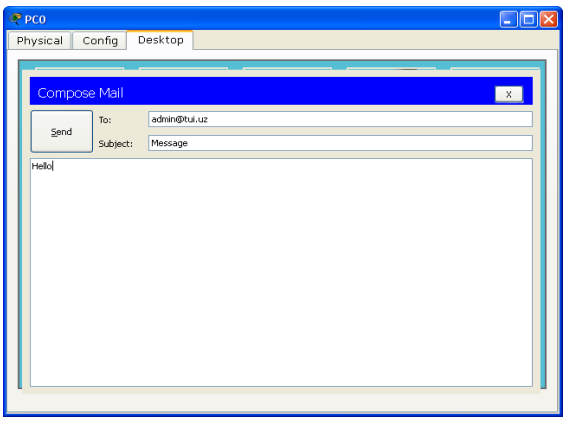


Рис.6. Передача сообщения

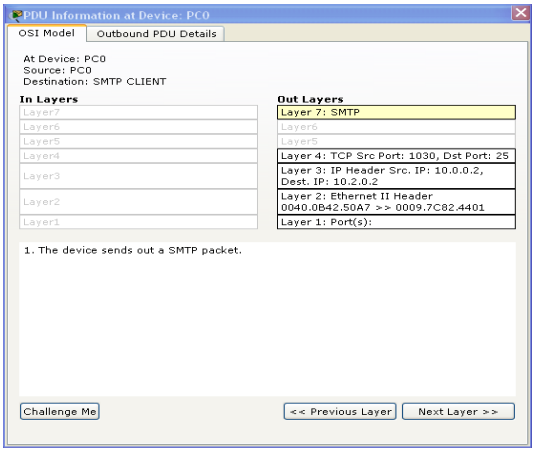


Рис.7. Передача пакетов SMTP

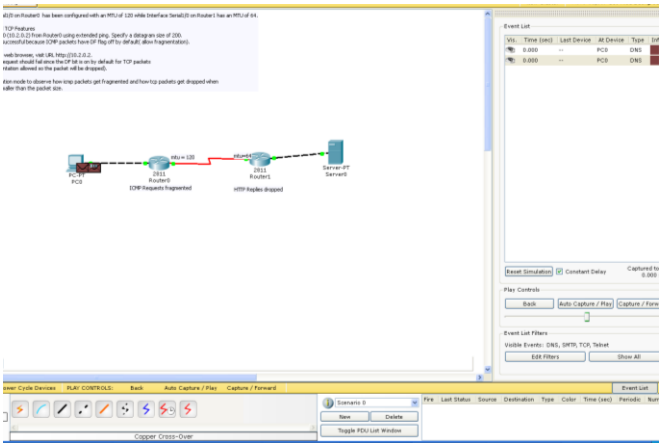


Рис.8. Работа DNS-сервиса

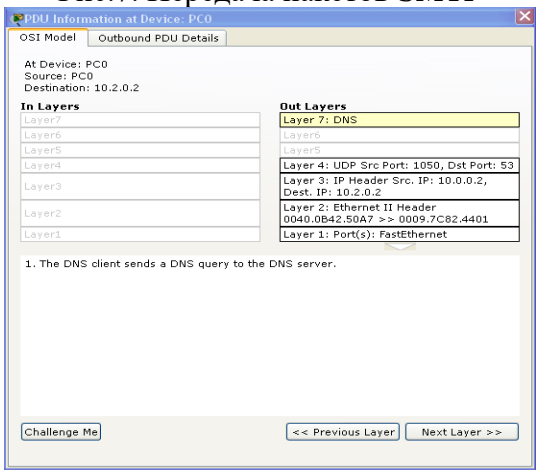


Рис.9. Выполнение DNS-запроса

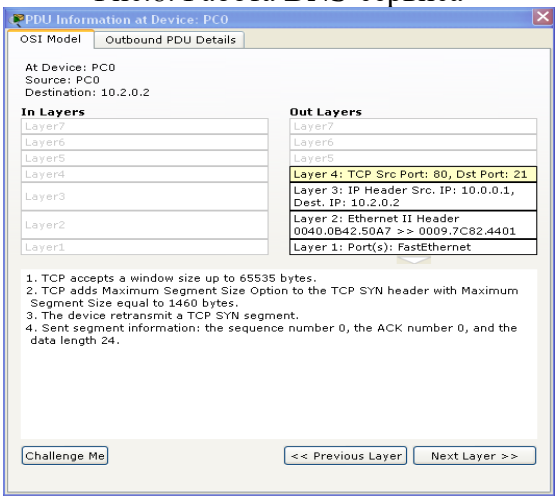


Рис.10. Передача TCP-пакетов

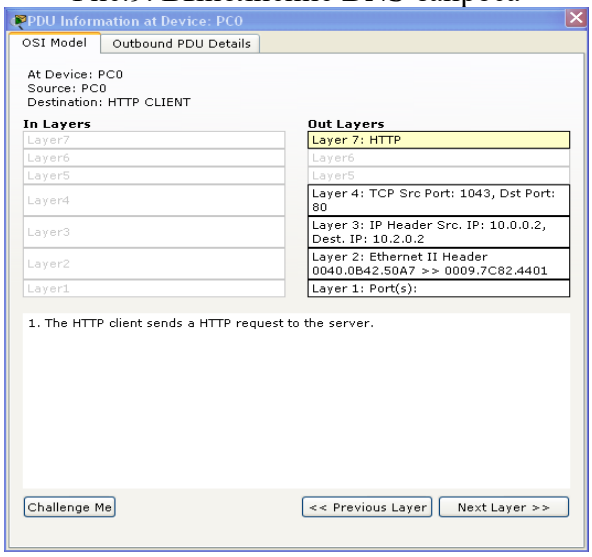


Рис.11. Выполнение HTTP-запроса

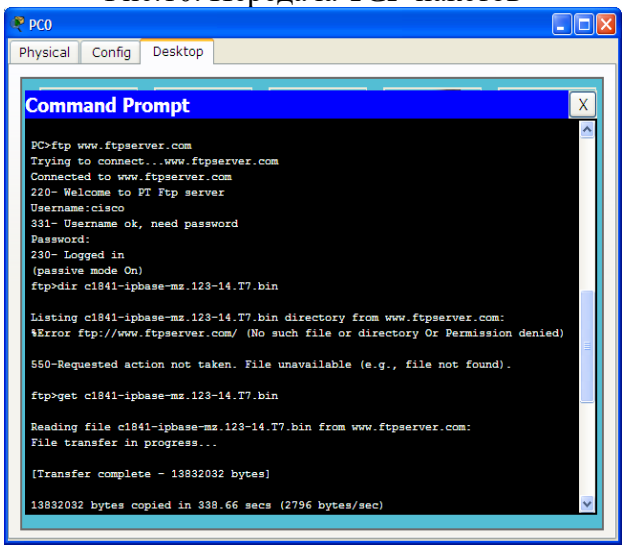


Рис.12. Работа с FTP-сервером

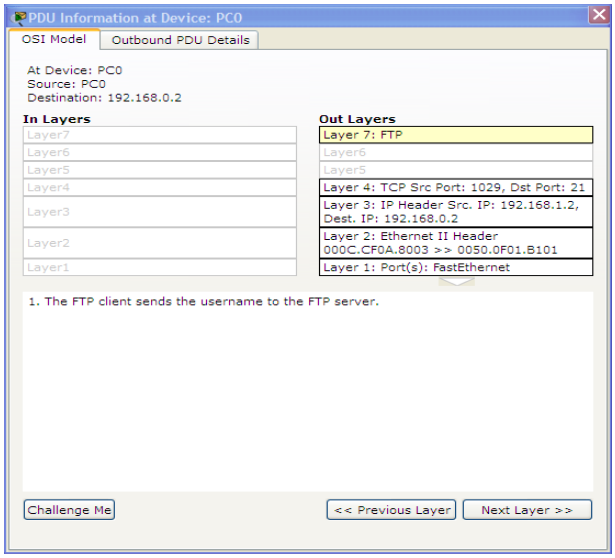


Рис.13. Авторизация пользователя на FTP-сервера

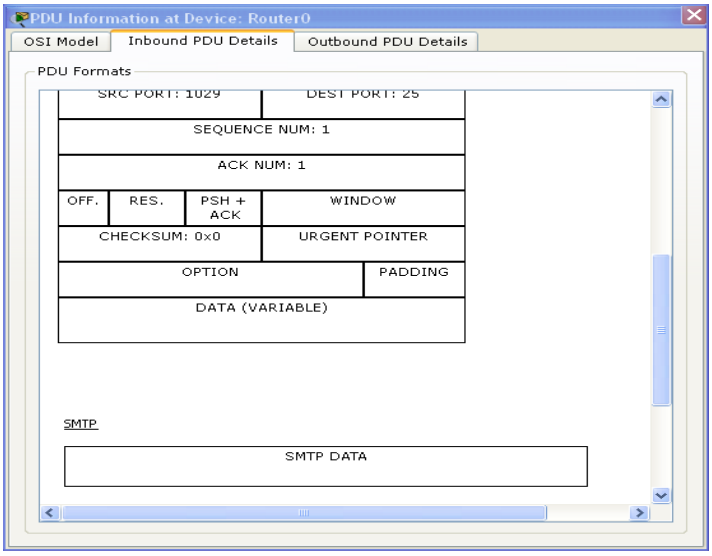


Рис.14. Структура пакетов

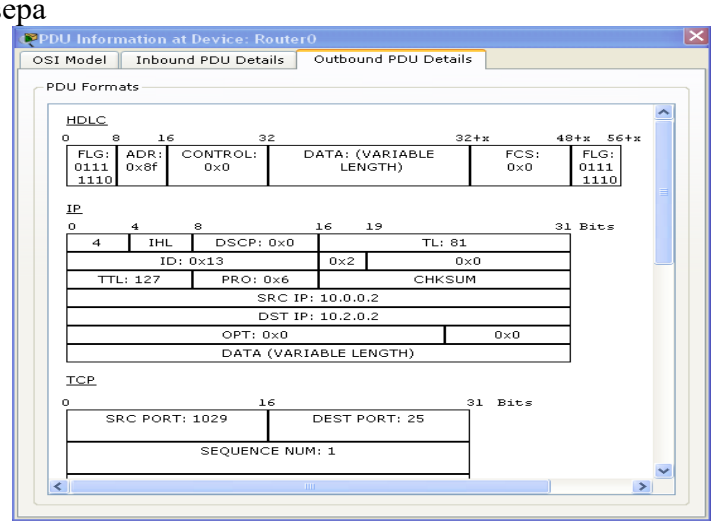


Рис.15. Структура пакетов

Содержание отчета

1. Краткое теоретические сведение
2. Анализ протоколов уровня приложения и транспорта.
3. Сравнение протоколов TCP и UDP.
4. Смоделированная сеть передачи данных с указанием параметров настройки.

Лабораторная работа №15 «E-mail услуги и протоколы»

Цель работы. Провести анализ работы E-mail протоколов и предоставляемые услуги с использованием Wireshark.

Краткие теоретические сведения

Главной целью протокола SMTP (Simple Mail Transfer Protocol, RFC-821, -822) служит надежная и эффективная доставка электронных почтовых сообщений. SMTP является довольно независимой субсистемой и требует только надежного канала связи. Средой для SMTP может служить отдельная локальная сеть, система сетей или весь Интернет.

SMTP базируется на следующей модели коммуникаций: в ответ на запрос пользователя почтовая программа-отправитель устанавливает двухстороннюю связь с программой-приемником (TCP, порт 25). Получателем может быть оконечный или промежуточный адресат. SMTP-команды генерируются отправителем и посылаются получателю. На каждую команду должен быть отправлен и получен отклик.

Когда канал организован, отправитель посылает команду MAIL, идентифицируя себя. Если получатель готов к приему сообщения, он посылает положительное подтверждение. Далее отправитель посылает команду RCPT, идентифицируя получателя почтового сообщения (таких команд можно выдать несколько, если число получателей более одного). Если получатель может принять сообщение для оконечного адресата, он выдает снова положительное подтверждение. В противном случае он отвергает получение сообщения для данного адресата, но не вообще почтовой посылки.

SMTP-отправитель и SMTP-получатель могут вести диалог с несколькими оконечными пользователями (Рис. 1). Любое почтовое сообщение завершается специальной последовательностью символов. Если получатель успешно завершил прием и обработку почтового сообщения, он посылает положительное подтверждение.

SMTP обеспечивает передачу почтового сообщения непосредственно конечному получателю, когда они соединены непосредственно. В противном случае пересылка может выполняться через одну или более промежуточных "почтовых станций".

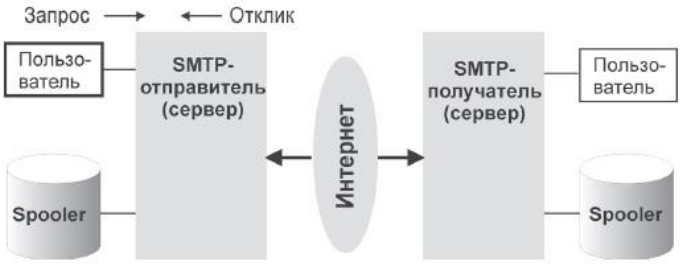


Рис. 1. Схема взаимодействия различных частей почтовой системы

Для решения поставленной задачи SMTP-сервер должен знать имя конечного получателя и название почтового ящика места назначения. Аргументом команды MAIL является адрес отправителя (обратный адрес). Аргументом команды RCPT служит адрес конечного получателя. Обратный адрес используется для посылки сообщения в случае ошибки.

Все отклики имеют цифровые коды. Команды, отклики и имена ЭВМ не чувствительны к тому, строчные или прописные символы использованы при их написании, но это не всегда справедливо при написании имен и адресов получателя.

Почтовый протокол SMTP работает только с ASCII-символами. Если транспортный канал работает с октетами, 7-битные коды будут дополнены нулевым восьмым битом. Именно здесь коренилась проблема пересылки почтовых сообщений на русском языке (русский алфавит требует

8-битового представления). Проблема усугубляется тем, что для русского алфавита принято 4 кодовых представления, здесь мы впереди планеты всей…
Как уже было сказано, процедура отправки почтового сообщения начинается с посылки команды MAIL, которая имеет формат:
MAIL <SP> FROM:<reverse-path> <CRLF>,
где <SP> — пробел, <CRLF> — комбинация кодов возврата каретки и перехода на новую строку, а <reverse-path> — обратный путь (имя почтового ящика отправителя). Именно этот адрес используется, если получатель сообщения воспользуется командой reply.
Эта команда сообщает SMTP-получателю, что стартует новая процедура и следует сбросить в исходное состояние все статусные таблицы, буферы и т.д. Если команда прошла, получатель реагирует откликом: 250 OK.
Аргумент может содержать не только адрес почтового ящика — в общем случае он является списком адресов ЭВМ-серверов, через которые пришло данное сообщение, включая, разумеется, и адрес почтового ящика отправителя. Первым в списке <reverse-path> стоит адрес ЭВМ-отправителя. После прохождения команды MAIL посылается команда RCPT:
`RCPT <SP> TO:<forward-path> <CRLF>`

Эта команда указывает адрес конечного получателя (<forward-path>). При благополучном прохождении команды получатель посылает кодотклик 250 OK, и запоминает полученный адрес. Если получатель неизвестен, SMTP-сервер пошлет отклик 550 Failure reply. Команда RCPT может повторяться сколько угодно раз, если адресат не один.
Аргумент может содержать не только адрес почтового ящика, но и маршрутный список ЭВМ по дороге к нему. Первым в этом списке должно стоять имя ЭВМ, получившей данную команду. По завершении этого этапа посылается собственно сообщение:
`DATA <CRLF>`

При правильном приеме этого сообщения SMTP-сервер реагирует посылкой отклика 354 Intermediate reply (промежуточный отклик), и рассматривает все последующие строки в качестве почтового текста. При получении кода конца текста отправляется отклик: 250 OK.
Признаком конца почтового сообщения является точка в самом начале строки, за которой следует <CRLF>.
В некоторых случаях адрес места назначения может содержать ошибку, но получатель знает правильный адрес. Тогда возможны два варианта отклика:
`1. 251 User not local; will forward to <forward-path>`

Это означает, что получатель берет на себя ответственность за доставку сообщения. Такое случается, когда адресат, например, мигрировал в другую субсеть в пределах зоны действия данного почтового сервера.
`2. 551 User not local; please try <forward-path>`

Получатель знает правильный адрес и предлагает отправителю переадресовать сообщение по адресу <forward-path>.
SMTP имеет команды для проверки корректности имени адресата (VRFY) и расширения списка адресов (EXPN). Обе команды в качестве аргументов используют строки символов (в некоторых реализациях эти две команды по своей функции идентичны). Для команды VRFY параметром является имя пользователя, а отклик может содержать его полное имя и адрес его почтового ящика.
Реакция на команду VRFY зависит от аргумента. Так если среди клиентов почтового сервера имеется два пользователя с именем Ivanov, откликом на команду "VRFY Ivanov" будет "553 User ambiguous". В общем случае команда VRFY Ivanov может получить в качестве откликов:
`250 Vasja Ivanov Ivanov@cl.itep.ru`
или:

```
251 User not local; will forward to Ivanov@cl.itep.ru
или:
550 String does not match anything (данная строка ничему не
соответству-
ет).
или:
551 User not local; please try Vasja@ns.itep.ru
или:
VRFY Chtozachertovchina
553 User ambiguous (несуществующее имя)
```

В случае распечатки списка адресов отклик занимает несколько строк, например:

```
EXPN Example-People
250-Juri Semenov Semenov@ns.itep.ru
250-Alexey Sher Sher@suncom.itep.ru
250-Andrey Bobyshev Bobyshev@ns.itep.ru
250-Igor Gursky Gursky@ns.itep.ru
```

В некоторых системах аргументом команды EXPN может быть имя файла, содержащего список почтовых адресов.
Команда Send And MaiL (SAML) предполагает доставку сообщение на экран терминала адресата и занесение в его почтовый ящик. Для открытия и закрытия коммуникационного канала используются команды:

```
HELO <SP> <domain> <CRLF>, где <domain> — имя запрашивающего домена.
QUIT <CRLF>
```

Выражение <forward-path> может быть маршрутом, имеющим вид "@ONE,@TWO:VANJA@THREE", где ONE, TWO и THREE — имена ЭВМ. Это подчеркивает различие между адресом и маршрутом. Концептуально элементы из <forward-path> переносятся в <reverse-path> при пересылке сообщений от одного SMTP-сервера к другому.
Если SMTP-сервер обнаружит, что доставка сообщения по адресу невозможна, тогда он формирует сообщение о "недоставленном письме", используя <reverse-path>. Следует также помнить, что и прямой, и обратный адреса-маршруты, вообще говоря, могут не иметь ничего общего с текстом заголовка почтового сообщения.
Если вы или ваша программа не указали обратного адреса, не следует думать, что это помешает работе почтовой программы и она не будет знать, куда посылать отклики. Практически все почтовые программы позволяют произвольно модифицировать поле <reverse-path>. Это может быть удобно, если вы собираетесь в командировку, но эта возможность широко используется и спамерами.
Поле <reverse-path> применяется почтовой программой, когда вы отвечаете на полученное сообщение с помощью утилиты Reply. Таким образом, ваш возмущенный ответ спамеру может прийти, например, к вам самому.
Следует помнить, что обратный IP-адрес (адрес отправителя) указан в каждом пакете, посылаемом адресату!

Протокол POP3 (Post Office Protocol)

Протокол обмена почтовой информацией **POP3 (RFC-1939)**предназначен для разбора почты из почтовых ящиков пользователей на их рабочие места при помощи программ-клиентов. Если по протоколу SMTP пользователи отправляют корреспонденцию через Интернет, то по протоколу

POP3 пользователи получают корреспонденцию из своих почтовых ящиков на почтовом сервере в локальные файлы.
Итак, POP3 (Post Office Protocol version 3). Номер его TCP-порта - 110. Основное отличие POP3 от других Интернет-протоколов верхнего уровня заключается в том, что в нем отсутствует широкий спектр кодов ошибок: в ответ на любую команду он посылает строки, начинающиеся с "+OK" или "-ERR", сигнализирующие соответственно об успешном или неудачном выполнении команды.
Набор основных команд протокола также достаточно прост:

USER name.

Это первое, что посылает клиент после того, как он считал строку-приветствие **+OK POP3 served ready**. Аргумент name указывает имя пользователя на данном почтовом сервере, для которого требуется получить доступ к почтовому ящику. Пример:

```
USER paaa
```

PASS password.

После того как пользователь указал свое имя, он должен указать пароль к своему почтовому ящику. Пример:

```
PASS doom
```

STAT.

Если пользователь существует и правильно ввел свой пароль, он может посмотреть почту. Команда **STAT** сообщает текущее состояние ящика. Формат ответа - **"+OK n m"**, где n - количество сообщений, m - количество байт. Пример (в почтовом ящике находятся 11 писем общим объемом 1594 байт):

```
C: STAT
S: +OK 11 1594
```

LIST n.

Для оценки размера конкретного письма существует команда **LIST.** Формат вывода такой же, как у **STAT**. Пример (третье письмо имеет размер 512 байт):

```
C: LIST 3
S: +OK 3 512
```

RETR n.

Для получения письма с сервера используется команда **RETR n**, где n - номер письма. Пример (получаем четвертое письмо):

```
C: RETR 4
S: +OK 124 octets
S: Здесь
S: идет
S: текст
S: письма
S:.
```

DELE n.

После того как письма успешно получены и сохранены локально, их можно удалить с почтового сервера. Для этого служит команда **DELE n.**. Пример (удаляем первое письмо):

```
C: DELE 1
S: +OK message 1 deleted
```

NOOP

(не использует каких-либо аргументов). При реализации этой команды сервер не делает ничего, лишь посылает положительный отклик.

RSET

(не использует каких-либо аргументов)Если какие-либо сообщения помечены как удаленные, сервер POP3 удаляет эту пометку и возвращает положительный отклик. Например:

```
K: RSET
C: +OK maildrop has 2 messages (320 octets)
```

TOP msg n,

где msg — номер сообщения, а n — число строк (применяется только в режиме TRANSACTION).

При положительном отклике на команду TOP сервер посылает заголовки сообщений и вслед за ними n строк их текста. Если n больше числа строк в сообщении, посылается все сообщение.

UIDL [msg],

где msg — номер сообщения является опционным (Unique-ID Listing).

APOP name digest,

где name — идентификатор почтового ящика, a digest — дайджест сообщения — MD5 (RFC-1828). Команда используется только на стадии авторизации.

Обычно любая сессия начинается с обмена USER/PASS. Но так как в некоторых случаях подключения к серверу POP3 может осуществляться достаточно часто, возрастает риск перехвата пароля. Альтернативным методом авторизации является использование команды APOP. Сервер, который поддерживает применение команды APOP, добавляет временную метку в свое стартовое уведомление. Синтаксис временной метки соответствует формату идентификаторов сообщений, описанному в [RFC822], и должен быть уникальным для всех заголовков уведомлений.

`QUIT.`

Для завершения сеанса используется команда **`QUIT.`**.

Протокол IMAP (Internet Message Access Protocol)

IMAP предоставляет пользователю обширные возможности для работы с почтовыми ящиками, находящимися на центральном сервере. Почтовая программа, использующая этот протокол, получает доступ к хранилищу корреспонденции на сервере так, как будто эта корреспонденция расположена на компьютере получателя. Электронными письмами можно манипулировать с компьютера пользователя (клиента) без постоянной пересылки с сервера и обратно файлов с полным содержанием писем.

Любая процедура начинается с команды клиента. Любая команда клиента начинается с префикса-идентификатора (обычно короткая буквенно-цифровая строка, например A0001, A0002 и т.д.), называемого меткой (tag). Для каждой команды клиент генерирует свою метку. Имеется два случая, когда строка, посланная клиентом, не представляет собой законченную команду. В первом — аргумент команды снабжается кодом, определяющим число октетов в строке (см. описание литеральных строк в разделе "Форматы данных"). Во втором — аргументы команды требуют отклика со стороны сервера (см. описание команды authenticate). В обоих вариантах сервер посылает запрос продолжения команды, если он готов. Такой отклик сервера начинается с символа "+".

Данные, передаваемые сервером клиенту, а также статусные отклики, которые не указывают на завершение выполнения команды, имеют префикс "*" и называются непомеченными откликами. Имеется три вида отклика завершения сервера: ok (указывает на успешное выполнение), no (отмечает неуспех) или bad (указывает на протокольную ошибку, например, не узнана команда или зафиксирована синтаксическая ошибка).

Команды клиента

Ниже описаны команды IMAP 4.1. Команды рассматриваются с учетом состояния, в котором они допустимы.

Следующие команды могут использоваться в любом состоянии: CAPABILITY, NOOP и LOGOUT.

Команда CAPABILITY

Аргументы:	отсутствуют	
Отклики:	необходим немаркированный отклик: CAPABILITY.	
Результат:	OK	успешное завершение команды;
	BAD	команда неизвестна или неверный аргумент.

Команда CAPABILITY запрашивает перечень возможностей, поддерживаемых сервером.

Команда NOOP

Аргументы:	отсутствуют.	
Отклики:	никакого специального отклика на эту команду не требуется.	
Результат:	OK	команда успешно завершена;
	BAD	команда неизвестна или неверен аргумент.

Команда NOOP ничего не делает и всегда успешно завершается.

Команда LOGOUT

Аргументы:	отсутствуют.	
Отклики:	необходим немаркированный отклик BYE.	
Результат:	OK	прерывание сессии завершено;
	BAD	неизвестная команда или неверный аргумент.

Команда LOGOUT информирует сервер о том, что клиент прерывает соединение. Сервер должен послать немаркированный отклик BYE, прежде чем отсылать маркированный отклик OK, после чего завершить разрыв соединения.

Команды клиента в состоянии без аутентификации

В состоянии без аутентификации команды AUTHENTICATE или LOGIN организуют аутентификацию и переводят систему в состояние с аутентификацией. Об аутентификации в IMAP можно прочесть в документе RFC-1731. Команда AUTHENTICATE предоставляет общий механизм для целого ряда методов аутентификации, среди которых команда LOGIN используется для традиционного ввода имени и пароля в текстовом виде.

Команда AUTHENTICATE

Аргументы:	имя механизма аутентификации.	
Отклики:	может быть запрошена дополнительная информация.	
Результат:	OK	Аутентификация завершена, осуществлен переход в состояние аутентификация выполнена;
	NO	Ошибка аутентификации: неподдерживаемый механизм аутентификации, параметры аутентификации отвергнуты;
	BAD	Неизвестная команда или неверный аргумент, механизм аутентификации прерван.

Команда AUTHENTICATE указывает серверу на механизм аутентификации, как это описано в [IMAP-AUTH]. Если сервер поддерживает запрошенный механизм аутентификации, он выполняет обмен согласно аутентификационному протоколу и идентифицирует клиента. Он может также согласовать опционный механизм защиты для последующих протоколов взаимодействия. Если запрошенный механизм аутентификации не поддерживается, сервер должен отвергнуть команду AUTHENTICATE путем посылки маркированного отклика NO.

Команда LOGIN

Аргументы:	имя пользователя, пароль.	
Отклики:	команда не требует какого-либо специального отклика.	
Результат:	OK	login завершено, система в состоянии с аутентификацией;

	NO	login не прошла: имя пользователя или пароль отвергнуты;
	BAD	команда неизвестна или неверный аргумент.

Команда LOGIN идентифицирует клиента серверу и передает пароль пользователя открытым текстом.

Команды клиента в состоянии "аутентификация осуществлена"

В состоянии "аутентификация осуществлена" разрешены команды манипуляции почтовыми ящиками как объектами-атомами. Команды SELECT и EXAMINE реализуют выбор почтового ящика и переход в состояние "выбрано".

В добавление к стандартным командам (CAPABILITY, NOOP и LOGOUT), в состоянии "аутентификация осуществлена" допустимы следующие команды: SELECT, EXAMINE, CREATE, DELETE, RENAME, SUBSCRIBE, UNSUBSCRIBE, LIST, LSUB, STATUS и APPEND.

Команда SELECT

Аргументы:	имя почтового ящика.	
Отклики:	Необходимы немаркированные отклики: FLAGS, EXISTS, RECENT; опционны немаркированные отклики OK: UNSEEN, PERMANENTFLAGS.	
Результат:	OK	процедура выбора закончена, система находится в состоянии выбрано;
	NO	выбор неудачен: нет такого ящика, доступ к почтовому ящику невозможен;
	BAD	команда неизвестна или неверен аргумент.

Команда SELECT осуществляет выбор почтового ящика, так, чтобы обеспечить доступ к сообщениям, находящимся там. Прежде чем присылать клиенту OK, сервер должен послать клиенту следующие немаркированные данные:

- FLAGS — флаги, определенные для почтового ящика
- <n> EXISTS — число сообщений в почтовом ящике
- <n> RECENT — число сообщений с набором флагов \Recent
- OK [UIDVALIDITY <n>] — уникальный идентификатор корректности

Команда EXAMINE

Аргументы:	имя почтового ящика.	
Отклики:	Необходимы немаркированные отклики: FLAGS, EXISTS, RECENT;	
	опционны немаркированные отклики OK: UNSEEN, PERMANENTFLAGS.	
Результат:	OK	Просмотр закончен, система в состоянии выбор сделан;
	NO	Просмотр не прошел, система в состоянии аутентификация выполнена; нет такого почтового ящика; доступ к почтовому ящику невозможен;
	BAD	Команда неизвестна или неверен аргумент.

Команда EXAMINE идентична команде SELECT и дает тот же результат, однако, выбранный почтовый ящик идентифицируется "только для чтения". Никакие изменения постоянного состояния почтового ящика в этом случае не разрешены. Текст маркированного отклика OK на команду EXAMINE должен начинаться с кода отклика [READONLY].

Команда CREATE

Аргументы:	имя почтового ящика.	
Отклики:	На эту команду не посылается каких-либо специфических откликов.	
Результат:	OK	Команда выполнена;
	NO	команда не выполнена: почтовый ящик с таким именем не может быть создан;
	BAD	команда неизвестна или неверен аргумент.

Команда CREATE создает почтовый ящик с заданным именем. Отклик OK присылается в случае, когда новый почтовый ящик с указанным именем создан. Попытка создания INBOX или

почтового ящика с именем существующего почтового ящика является ошибкой. Любая ошибка при попытке создания почтового ящика вызовет маркированный отклик NO.

Команда DELETE

Аргументы:	имя почтового ящика.	
Отклики:	Команда не требует каких-либо откликов.	
Результат:	OK	команда завершена;
	NO	ошибка при выполнении команды: не удается стереть ящик с этим именем;
	BAD	команда неизвестна или неверен аргумент.

Команда DELETE навечно удаляет почтовый ящик с указанным именем. При этом присылается маркированный отклик OK только в том случае, когда ящик уничтожен. Ошибкой считается попытка стереть INBOX или ящик с несуществующим именем.

Команда UNSUBSCRIBE

Аргументы:	имя почтового ящика.	
Отклики:	Эта команда не требует каких-либо специфических откликов.	
Результат:	OK	ликвидация подписки прошла успешно;
	NO	ликвидация подписки не прошла: это невозможно для данного имени;
	BAD	команда неизвестна или неверен аргумент.

Команда UNSUBSCRIBE удаляет специфицированный почтовый ящик из списка активных или подписных почтовых ящиков данного сервера, как это определяется командой LSUB. Эта команда возвращает маркированный отклик OK только в случае, если ликвидация подписки прошла успешно.

Команда LIST

Аргументы:	имя, имя почтового ящика может содержать символы подмены (wildcard).	
Отклики:	немаркированные отклики LIST.	
Результат:	OK	команда LIST выполнена;
	NO	команда не прошла: не возможно выполнение LIST для данного образца или имени;
	BAD	команда неизвестна или неверен аргумент.

Команда LIST возвращает субнабор имен из полного набора, доступного клиенту. Присылается нуль или более немаркированных откликов LIST, содержащих атрибуты имен, иерархические разделители и имена.

ВЫПОЛНЕНИЕ

Доменное имя сервера: mail.ru

Ответ сервера на запрос HELO:

250-smtp8.mail.ru Hello sisko24e37e878 [83.69.130.8]

E-mail client	E-mail server
Mail from: ramil_kh@mail.ru	250 Ok
Rcpt to: mescase@rambler.ru	250 Accepted
Data	354 Enter message, ending with “.” On a line by itself

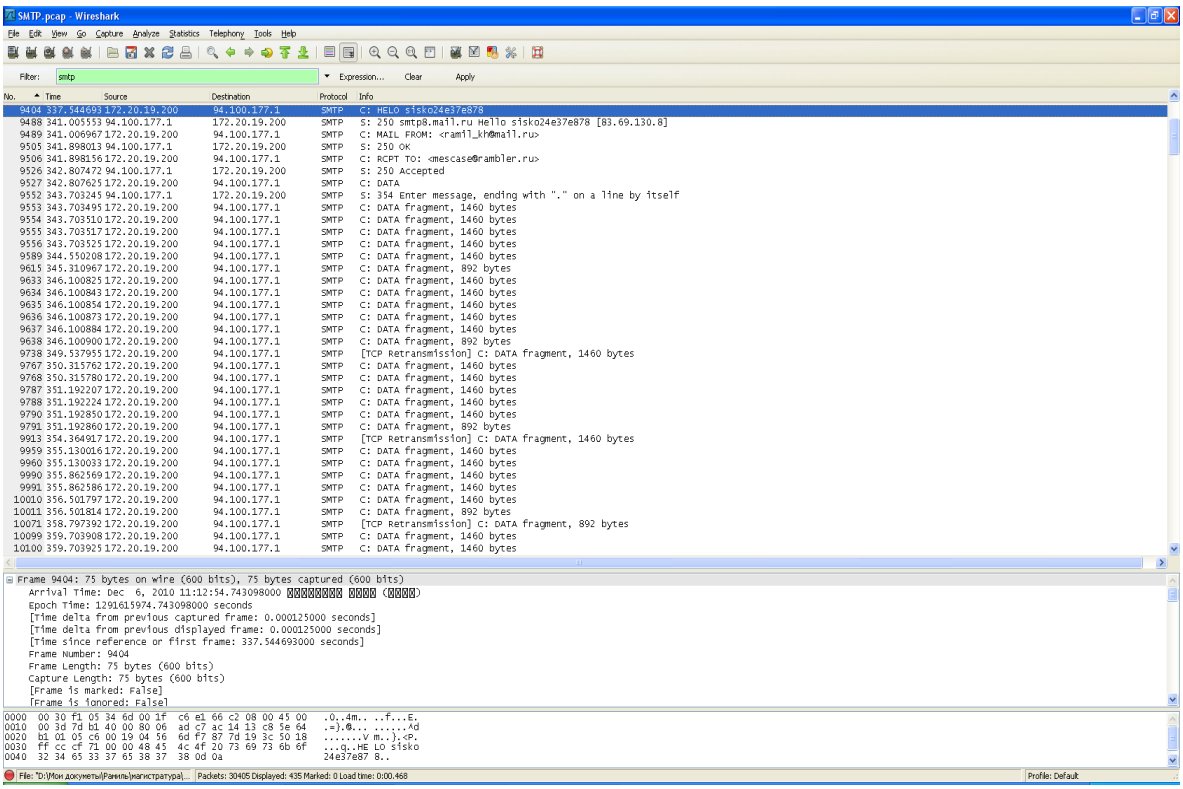


Рис.1. Начало отправки

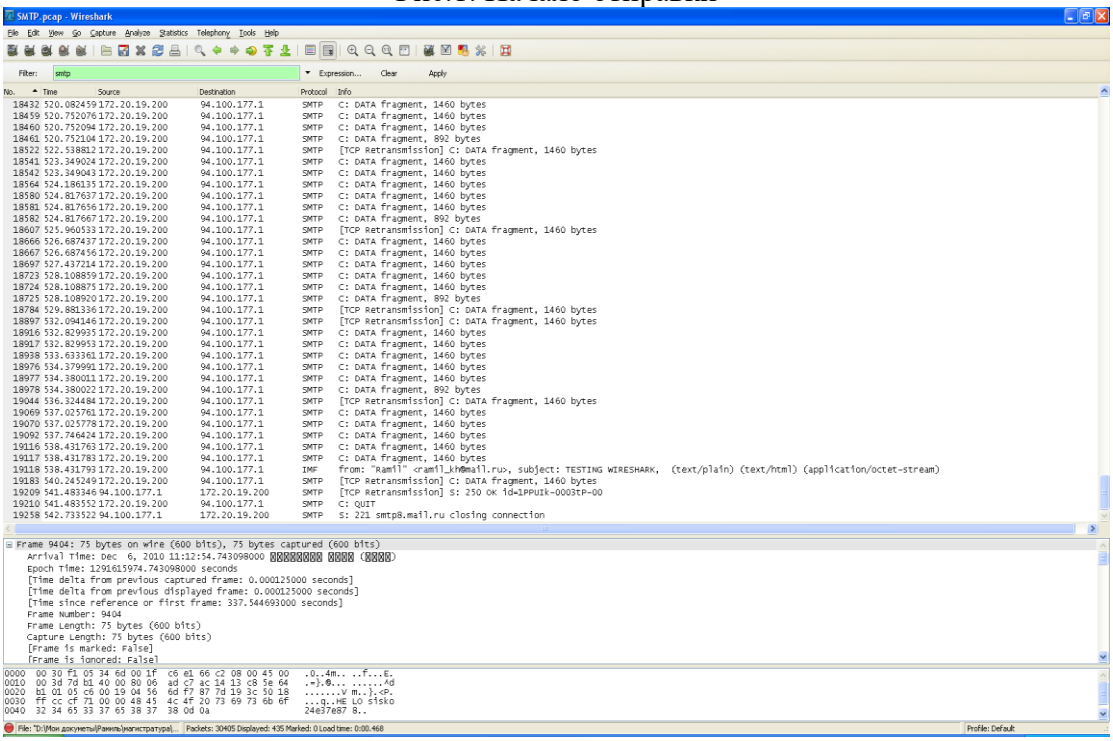


Рис.2. Окончание отправки

В результате отправки сообщения реальный размер файла составил 354 КВ. Размер файла, полученный адресатом сообщения, оказался равным 488 КВ. Таким образом, реальный размер файла составил 73% от размера доставленного сообщения, оставшаяся информация представляет собой дополнительные данные, полученные вследствие выполнения кодирования для надежной доставки данных. На сегодняшний день, большинство из почтовых служб позволяют передавать файлы размером до 20-25 МБ.

Содержание отчета

1. Краткое теоретические сведение
2. Анализ протоколов e-mail и услуг.
3. Протоколы SMTP, POP3, IMAP 4.1.
4. Результаты отправки SMTP – пакетов

Лабораторная работа №16 Протоколы транспортного уровня TCP/IP, TCP и UDP

Цель работы. Провести анализ работы протоколов транспортного уровня TCP/IP(TCP, UDP) с использованием Netstat.

Краткие теоретические сведения

Netstat – отображает активные подключения TCP, портов, прослушиваемых компьютером, статистики Ethernet, таблицы маршрутизации IP, статистики IPv4 (для протоколов IP, ICMP, TCP и UDP) и IPv6 (для протоколов IPv6, ICMPv6, TCP через IPv6 и UDP через IPv6). Запущенная без параметров, команда **nbtstat** отображает подключения TCP.

Синтаксис

```
netstat [-Aan] [-f семейство_адресов] [-I интерфейс] [-p
имя_протокола] [система] [core]
netstat [-n] [-s] [-i | -r] [-f семейство_адресов] [-I интерфейс] [-p
имя_протокола] [система] [core]
netstat [-n] [-I интерфейс] интервал [система] [core]
```

Первая форма команды показывает список активных сокетов (sockets) для каждого протокола. Вторая форма выбирает одну из нескольких других сетевых структур данных. Третья форма показывает динамическую статистику пересылки пакетов по сконфигурированным сетевым интерфейсам; аргумент интервал задает, сколько секунд собирается информация между последовательными показами.

Опции

-a	Показывать состояние всех сокетов; обычно сокеты, используемые серверными процессами, не показываются.
-A	Показывать адреса любых управляющих блоков протокола, связанных с сокетами; используется для отладки.
-i	Показывать состояние автоматически сконфигурированных (auto-configured) интерфейсов. Интерфейсы, статически сконфигурированные в системе, но не найденные во время загрузки, не показываются.
-n	Показывать сетевые адреса как числа. netstat обычно показывает адреса как символы. Эту опцию можно использовать с любым форматом показа.
-r	Показать таблицы маршрутизации. При использовании с опцией -s, показывает статистику маршрутизации.
-s	Показать статистическую информацию по протоколам. При использовании с опцией -r, показывает статистику маршрутизации.
-f семейство_адресов	Ограничить показ статистики или адресов управляющих блоков только указанным семейством_адресов, в качестве которого можно указывать: **inet** Для семейства адресов **AF_INET**, или **unix** Для семейства адресов **AF_UNIX**.
-I интерфейс	Выделить информацию об указанном интерфейсе в отдельный столбец; по умолчанию (для третьей формы команды) используется интерфейс с наибольшим объёмом переданной информации с момента последней перезагрузки системы. В качестве интерфейса можно указывать любой из интерфейсов, перечисленных в файле конфигурации системы, например, emd1 или lo0.

-p имя_протокола	Ограничить показ статистики или адресов управляющих блоков только протоколом с указанным именем_протокола, например, tcp.

Активные сокеты

Для каждого активного сокета показывается протокол, размер очередей приема и получения (в байтах), локальный и удаленный адрес, а также внутреннее состояние протокола. Символьный формат, обычно используемый для показа адресов сокетов, — это либо:
`имя_хоста.порт`
если имя хоста указано, либо:
`сеть.порт`
если адрес сокета задает сеть, но не конкретный хост. Имена хостов и сетей берутся из соответствующих записей в файле /etc/hosts или /etc/networks.
Если имя сети или хоста для адреса неизвестно (или если указана опция -n), адрес показывается числами. Не указанные или «обобщенные» адреса и порты показываются как «*».

Сокеты TCP

Для сокетов TCP допустимы следующие значения состояния:

CLOSED	Закрыт. Сокет не используется.
LISTEN	Ожидает входящих соединений.
SYN_SENT	Активно пытается установить соединение.
SYN_RECEIVED	Идет начальная синхронизация соединения.
ESTABLISHED	Соединение установлено.
CLOSE_WAIT	Удаленная сторона отключилась; ожидание закрытия сокета.
FIN_WAIT_1	Сокет закрыт; отключение соединения.
CLOSING	Сокет закрыт, затем удаленная сторона отключилась; ожидание подтверждения.
LAST_ACK	Удаленная сторона отключилась, затем сокет закрыт; ожидание подтверждения.
FIN_WAIT_2	Сокет закрыт; ожидание отключения удаленной стороны.
TIME_WAIT	Сокет закрыт, но ожидает пакеты, ещё находящиеся в сети для обработки

Показ таблицы маршрутизации

Таблица маршрутизации показывает все имеющиеся маршруты (routes) и статус каждого из них. Каждый маршрут состоит из целевого хоста или сети и шлюза (gateway), который используется для пересылки пакетов. Столбец flags (флаги) показывает статус маршрута (U, если он включен), ведет ли маршрут на шлюз (G), был ли маршрут создан динамически при помощи перенаправления (D) и используется ли адрес индивидуального хоста (H) вместо адреса сети. Например, интерфейс закольцовывания (loopback transport provider), lo0, всегда имеет флаг H. Прямые маршруты создаются для каждого интерфейса, подключенного к локальному хосту; поле gateway (шлюз) для таких записей показывает адрес выходного интерфейса.
Столбец refcnt показывает текущее количество активных использований для маршрута. Протоколы, ориентированные на соединение, обычно используют в ходе соединения один

маршрут, тогда как протоколы без соединения получают маршрут для каждой посылки одному и тому же адресату.
Столбец use показывает количество пакетов, посланных по маршруту.
Столбец interface показывает сетевой интерфейс, используемый маршрутом.

Суммарная статистика передачи данных

Когда задан аргумент интервал, netstat показывает таблицу суммарной статистической информации о переданных пакетах, ошибках и коллизиях. Первая показываемая строка данных, а также каждая последующая 24-я строка содержит суммарную статистическую информацию с момента последней перезагрузки системы. Каждая последующая строка показывает данные, накопленные за очередной указанный в командной строке интервал с момента последнего показа.

ВЫПОЛЕНИЕ

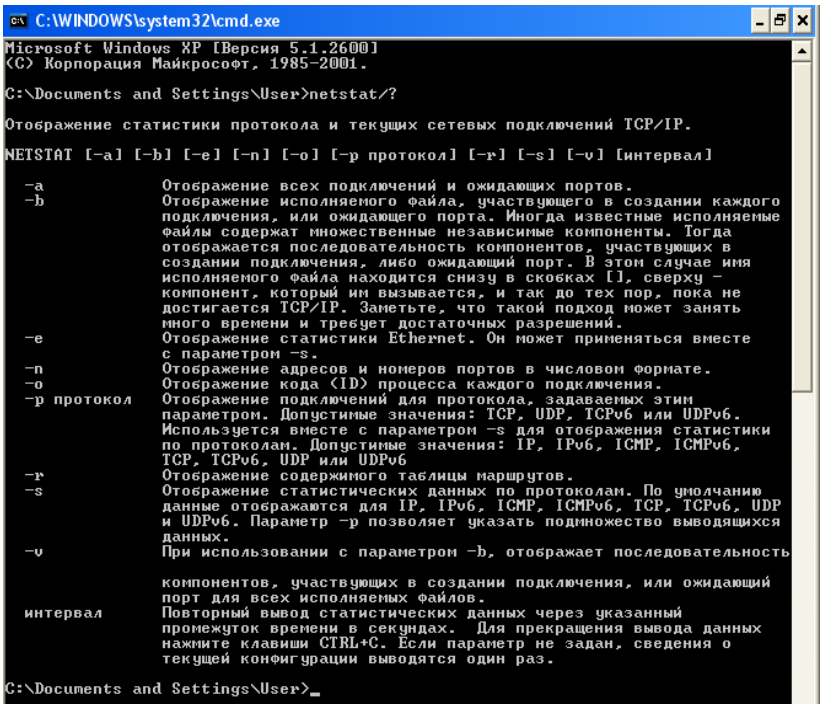


Рис.1. Вывод справки (netstat/?)

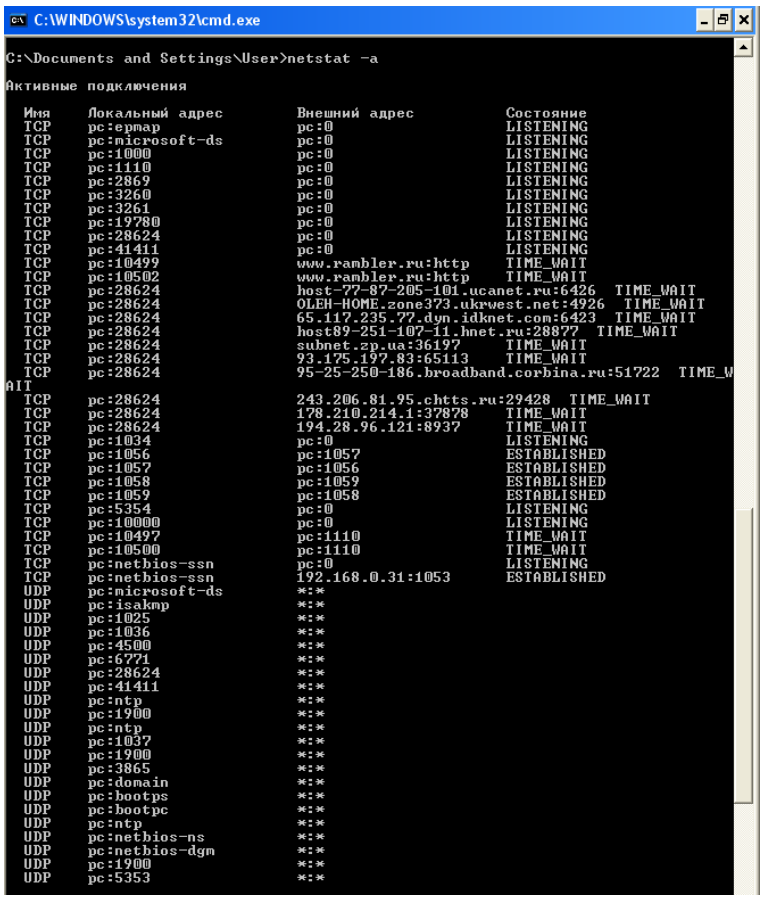


Рис. 2. Использование Netstat для просмотра имеющихся соединений(netstat -a):

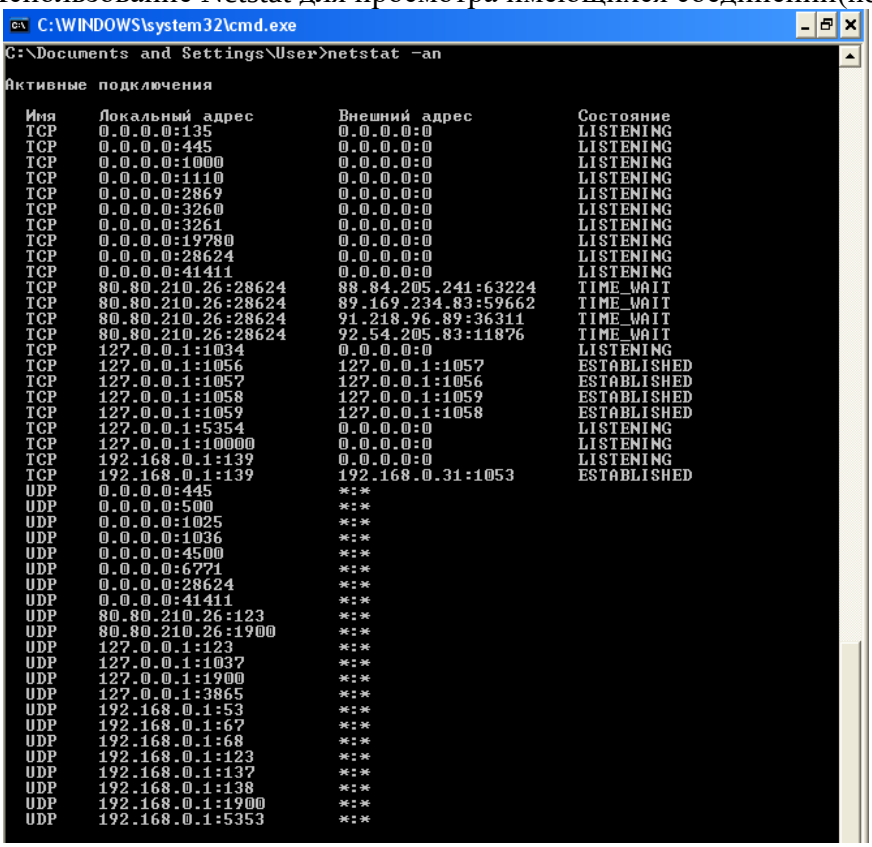


Рис.3. Вывод соединений в числовом формате(netstat -an)

В процессе выполнения команд Netstat компьютер клиента имел выход в Интернет. При этом выполнялся запрос на сайт www.rambler.ru через браузер Mozilla Firefox. Через определенные интервалы времени компьютер находился в режиме ожидания для установления соединения

(состояние TIME_WAIT). После успешного выполнения соединения на экран выводилось состояние ESTABLISHED.

Соединение	Имя	Локальный адрес	Внешний адрес	Состояние
1.	TCP	pc: 10499	www.rambler.ru: http	TIME_WAIT
	TCP	80.80.210.26:28624	88.84.205.241: 63224	TIME_WAIT
2.	TCP	pc: 10502	www.rambler.ru: http	TIME_WAIT
	TCP	80.80.210.26:28624	89.169.234.83: 59662	TIME_WAIT
3.	TCP	pc: 28624	Host-77-81-205-101.ucanet.ru: 6426	TIME_WAIT
	TCP	80.80.210.26:28624	91.218.96.89: 36311	TIME_WAIT
4.	UDP	pc: ntp	*:*	
	UDP	80.80.210.26:123	*:*	
5.	UDP	pc:1037	*:*	
	UDP	80.80.210.26:1900	*:*	
6.	UDP	pc: 1900	*:*	
	UDP	127.0.0.1:123	*:*	

Содержание отчета

1. Краткое теоретические сведение
2. Основные команды утилиты Netstat.
3. Основные состояния, выводимые утилитой Netstat.
4. Результаты анализа выполнения запросов к сети Интернет.