Государственный комитет Российской Федерации по телекоммуникациям Сибирский государственный университет телекоммуникаций и информатики

Контрольная работа

По дисциплине «Теория информации»

Вариант №7

Работу выполнил:

студент 2 курса

группы ЗП-21

Лощаков Юрий Сергеевич

Работу проверил:

Работа защищена

«\_\_\_» \_\_\_\_\_\_\_\_\_\_2014г.

**Новосибирск 2014**

Оглавление

[Задача 1 2](#_Toc385909265)

[Теория 2](#_Toc385909266)

[Код программы 6](#_Toc385909267)

[Задача 2 12](#_Toc385909268)

[Теория 13](#_Toc385909269)

[Код программы 15](#_Toc385909270)

# Задача 1

Составить программу, оценивающую энтропию текстового файла. Оценки энтропии необходимо вычислить двумя способами, т.е. используя частоты отдельных символов и частоты пар символов.

Файл, содержит текст на естественном языке (русский или английский), строчные и заглавные символы не отличаются, знаки препинания объединены один символ. В программе должна быть предусмотрена возможность ввода имени файла.

Заполнить таблицу:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название текста | Оценка энтропии (одиночные символы) | Оценка избыточности (одиночные символы) | Оценка энтропии (частоты пар символов) | Оценка избыточности (частоты пар символов) |
| 1.txt(~250байт) | 4.521 | 0.104(10%) | 2.369 | 0.53(53%) |
| 2.txt(10байт) | 1.371 | 0.135(13.5%) | 0.913 | 0.424(42%) |
| 3.txt(~11Kбайт) | 4.508 | 0.205(20%) | 3.48 | 0.387(39%) |

Таблица 1

Примечание:

Все текстовые файлы и исходный код программы можно просмотреть по адресу - [*https://github.com/jurasikmaniac/TI\_2013/tree/master/Zadacha\_1*](https://github.com/jurasikmaniac/TI_2013/tree/master/Zadacha_1)

## Теория

Пусть имеется дискретный вероятностный источник (в нашем случае это текстовый файл), порождающий символы алфавита *А*={*a1,…,an*} с вероятностями *,* . Основной характеристикой источникаявляется *энтропия,* которая представляет собой среднее значение количества информации в сообщении источника и определяется выражением (для двоичного случая)



Формула 1

Энтропия характеризует меру неопределенности выбора для данного источника.

Максимальное значение энтропии будет, если символы алфавита порождают с одинаковой вероятностью. При этом

Применение Формулы 1 к алфавиту русского языка дает значение средней информации на знак: H**1(r) = 4,36** бит, а для английского языка H**1(e) = 4,04** бит, для французского H**1(l) = 3,96** бит, для немецкого H**1(d) = 4,10** бит, для испанского H**1(s) = 3,98** бит. Как мы видим, и для русского, и для английского языков учет вероятностей появления букв в сообщениях приводит к уменьшению среднего информационного содержания буквы. Несовпадение значений средней информации для английского, французского и немецкого языков, основанных на одном алфавите, связано с тем, что частоты появления одинаковых букв в них различаются.

    В рассматриваемом приближении по умолчанию предполагается, что вероятность появления любого знака в любом месте сообщения остается одинаковой и не зависит от того, какие знаки или их сочетания предшествуют данному. Такие сообщения называются ***шенноновскими*** (или сообщениями без памяти, *бернуллиевским*).

    Сообщения, в которых вероятность появления каждого отдельного знака не меняется со временем, называются ***шенноновскими***, а порождающий их отправитель – ***шенноновским источником***.

    Если сообщение является шенноновским, то набор знаков (алфавит) и вероятности их появления в сообщении могут считаться известными заранее. В этом случае, с одной стороны, можно предложить оптимальные способы кодирования, уменьшающие суммарную длину сообщения при передаче по каналу связи.

    Последующие (второе и далее) приближения при оценке значения информации, приходящейся на знак алфавита, строятся путем учета корреляций, т.е. связей между буквами в словах. Дело в том, что в словах буквы появляются не в любых сочетаниях; это понижает неопределенность угадывания следующей буквы после нескольких, например, в русском языке нет слов, в которых встречается сочетание щц или фъ. И напротив, после некоторых сочетаний можно с большей определенностью, чем чистый случай, судить о появлении следующей буквы, например, после распространенного сочетания пр- всегда следует гласная буква, а их в русском языке 10 и, следовательно, вероятность угадывания следующей буквы 1/10, а не 1/33. В связи с этим примем следующее определение:

***Сообщения (а также источники, их порождающие), в которых существуют статистические связи (корреляции) между знаками или их сочетаниями, называются сообщениями (источниками) с памятью или марковскими сообщениями (источниками).***

Расчёт **H2**

Обозначим как  вероятность того, что источник послал символ поле отправки символа .

Если символ  принимает произвольное значение из алфавита источника дискретных сообщений, то энтропию источника можно записать как:

Обозначим как  вероятность того, что источник послал символ поле отправки символа .

Если символ  принимает произвольное значение из алфавита источника дискретных сообщений, то энтропию источника можно записать как:



    Как указывается в книге Л.Бриллюэна, учет в английских словах двухбуквенных сочетаний понижает среднюю информацию на знак до значения **H2(e)=3,32** бит, учет трехбуквенных – до **H3(e)=3,10** бит. Шеннон сумел приблизительно оценить **H5(e) http://it.kgsu.ru/TI_3/images/ris5_2.gif2,1** бит, **H8(e) http://it.kgsu.ru/TI_3/images/ris5_2.gif1,9** бит. Аналогичные исследования для русского языка дают: **H2(r) = 3,52** бит; **H3(r)= 3,01**бит.

    Последовательность **H0, H1, H2...** является убывающей в любом языке. Экстраполируя ее на учет бесконечного числа корреляций, можно оценить предельную информацию на знак в данном языке , которая будет отражать минимальную неопределенность, связанную с выбором знака алфавита без учета семантических особенностей языка, в то время как**I0** является другим предельным случаем, поскольку характеризует наибольшую информацию, которая может содержаться в знаке данного алфавита. Шеннон ввел величину, которую назвал ***относительной избыточностью языка***:

    Избыточность является мерой бесполезно совершаемых альтернативных выборов при чтении текста. Эта величина показывает, какую долю лишней информации содержат тексты данного языка; лишней в том отношении, что она определяется структурой самого языка и, следовательно, может быть восстановлена без явного указания в буквенном виде.

    Исследования Шеннона для английского языка дали значение http://it.kgsu.ru/TI_3/images/ris5_5.gif1,4÷1,5 бит, что по отношению к **I0=4,755** бит создает избыточность около 0,68. Подобные оценки показывают, что и для других европейских языков, в том числе русского, избыточность составляет 60 – 70%. Это означает, что в принципе возможно почти трехкратное (!) сокращение текстов без ущерба для их содержательной стороны и выразительности.

В нашей задаче избыточность будет вычисляться для частот одиночных символов и для частот пар символов, тогда формулы расчета избыточности примут следующий вид соответственно:

Расчёты и определения избыточности языка-информации-текста в разных источниках может разниться. В этом задании была выбрано определение описанное выше. Однако все нужные величины для расчёта других интерпретаций избыточности также вычисляются в программе, что позволяет изменением одной формулы получать нужный вид результата.

## Код программы

#include "stdafx.h"

#include <cmath>

#include <cstdio>

#include <ctype.h>

#include <clocale>

#include <Windows.h>

#pragma warning (once : 4996)

#define AlfabetN 256

//B Visual studio 2012 нет поддержки log2 Но в 2013 есть

//inline float log2(float n) {

// if (n==0)

// {

// return 0;

// }

// return log(n) / log(2.0);

//}

//Предварительные настройки приложения

inline void SetApp(void)

{

//setlocale( LC\_ALL, "ru\_RU.cp1251" );

SetConsoleCP(1251); // Русская локаль

SetConsoleOutputCP(1251);

}

struct result // Сруктура хранения информации

{

float freq[AlfabetN];

float H,Hmax;

}res;

//Оценка энтропии, одиночные символы

float entropy\_1 (char \*fn)

{

FILE \*fi;

float freq[AlfabetN] = {0};

float H=0,Hmax=0;

int K=0;

int ch=EOF;

long total=0;

puts("Файл:");

puts(fn);

fi = fopen(fn, "r");

if(fi == NULL)

{

puts("Такой файл не найден!");

system("Pause");

return -1;

}

while((ch = fgetc(fi)) != EOF)

{

//Если англ или рус алфавит

if(isalpha(ch)||strchr("ЁЙЦУКЕНГШЩЗХЪФЫВАПРОЛДЖЭЯЧСМИТЬБЮёйцукенгшщзхъфывапролджэячсмитьбю", ch))

{

if (strchr("ЁЙЦУКЕНГШЩЗХЪФЫВАПРОЛДЖЭЯЧСМИТЬБЮ", ch))//Если Заглавные рус

{

if ( ch == 0xA8) //Отдельно для Ё

{

ch=0xB8;

}else

{

ch=ch-0xC0+0xE0; //А и а рус

}

freq[ch]++;

total++;

}else{

ch = tolower(ch);

freq[ch]++;

total++;

}

}

else

{

if (ch == ' ') //пробел

{

freq[ch]++;

total++;

}else

{

if (strchr("-.,:!?;", ch)) //знаки препинания

{

ch = '.';

freq[ch]++;

total++;

}

}

}

}

for (int i = 0; i < AlfabetN; i++)

{

if (freq[i]!=0)

{

//printf("freq =%3.0f ",freq[i]);

freq[i] = freq[i]/total;

printf("P[%c] = %1.3f\n",i,freq[i]);

H = H + freq[i]\*log2(freq[i]);

K++;

}

}

for (int i = 0; i < AlfabetN; i++)

{

res.freq[i]=0;

res.freq[i]=freq[i];

}

res.H = H;

Hmax=log2(K);

if (Hmax==0)Hmax=1;

res.Hmax = Hmax;

printf("Всего символов в файле: %d\n",total);

printf("Всего символов в алфавите: %d\n",K);

printf("Энтропия файла: %3.3f\n",(-1)\*H);

printf("Максимальная Энтропия файла: %3.3f\n",Hmax);

printf("Избыточность текста составила: %3.3f\n",1 + H/Hmax);

fclose(fi);

system("PAUSE");

return 0;

}

//Оценка энтропии, пары символов

float entropy\_2 (char \*fn)

{

FILE \*fi;

float freq2[AlfabetN][AlfabetN] = {0};

float H2=0;

int ch=EOF,lastch=EOF;;

long sum=0;

entropy\_1(fn);

fi = fopen(fn, "r");

if(fi == NULL)

{

puts("Такой файл не найден!");

system("Pause");

return -1;

}

while((ch = fgetc(fi)) != EOF)

{

sum++;

}

if (sum<2||sum==0)

{

puts("в файле нет пар символов");

system("Pause");

return -2;

}

rewind(fi);

lastch= fgetc(fi);

//Если англ или рус алфавит

if(isalpha(lastch)||strchr("ЁЙЦУКЕНГШЩЗХЪФЫВАПРОЛДЖЭЯЧСМИТЬБЮёйцукенгшщзхъфывапролджэячсмитьбю", lastch))

{

if (strchr("ЁЙЦУКЕНГШЩЗХЪФЫВАПРОЛДЖЭЯЧСМИТЬБЮ", lastch)) //Если Заглавные рус

{

if ( lastch == 0xA8) //Отдельно для Ё

{

lastch=0xB8;

}else

{

lastch=lastch-0xC0+0xE0; //А и а рус

}

}else{

lastch = tolower(lastch);

}

}

else

{

if (strchr("-.,:!?;", lastch)) //знаки припинания

{

lastch = '.';

}

}

//Считаем количество повторений всех пар символов

while((ch = fgetc(fi)) != EOF)

{

//Если англ или рус алфавит

if(isalpha(ch)||strchr("ЁЙЦУКЕНГШЩЗХЪФЫВАПРОЛДЖЭЯЧСМИТЬБЮёйцукенгшщзхъфывапролджэячсмитьбю", ch))

{

if (strchr("ЁЙЦУКЕНГШЩЗХЪФЫВАПРОЛДЖЭЯЧСМИТЬБЮ", ch))//Если Заглавные рус

{

if ( ch == 0xA8) //Отдельно для Ё

{

ch=0xB8;

}else

{

ch=ch-0xC0+0xE0; //А и а рус

}

freq2[lastch][ch]++;

lastch=ch;

}else{

ch = tolower(ch);

freq2[lastch][ch]++;

lastch=ch;

}

}

else

{

if (ch == ' ') //пробел

{

freq2[lastch][ch]++;

lastch=ch;

}else

{

if (strchr("-.,:!?;", ch)) //знаки припинания

{

ch = '.';

freq2[lastch][ch]++;

lastch=ch;

}

}

}

}

//Вычистляем частоту(вероятность) пар символов

for (int i = 0; i < AlfabetN; i++)

{

sum=0;

//вычисляем число повторений пар с фиксированной i буквой

for (int j = 0; j < AlfabetN; j++)

{

sum=sum+freq2[i][j];

}

//вычисляем вероятность появление j символа после i символа

for (int j = 0; j < AlfabetN; j++)

{

freq2[i][j]=freq2[i][j]/sum;

//if(freq2[i][j]>0) printf("P2[%c][%c]= %f\n",i,j,freq2[i][j]);

}

}

H2=0;

//Считаем энтропию для пар символов

for (int i = 0; i < AlfabetN; i++)

{

for (int j = 0; j < AlfabetN; j++)

{

if(freq2[i][j]>0 )

{

H2=H2+res.freq[i]\*freq2[i][j]\*log2(freq2[i][j]);

}

}

}

printf("Энтропия файла для пар символов: %3.3f\n",(-1)\*H2);

printf("Максимальная Энтропия файла: %3.3f\n",res.Hmax);

if (res.Hmax==0)res.Hmax=1;

printf("Избыточность текста для пар символов составила: %3.3f\n",1 + H2/res.Hmax);

fclose(fi);

system("PAUSE");

return 0;

}

//Меню программы, type - задает нужные пункты меню.

inline wchar\_t Menu(unsigned char type)

{

wchar\_t c;

switch (type)

{

case 1: system("CLS");

puts("########################");

puts("1. Ввести имя файла");

puts("2. Оценка энтропии, одиночные символы");

puts("3. Оценка энтропии, пары символов");

puts("4. Выход из программы");

puts("\n########################");

puts("Выбрать пункт меню №: ");

c=getchar();

fflush(stdin);

return c;

default:puts("Err, нет такого меню");

system("PAUSE");

break;

}

return 0;

}

// Количество символов в алфавите

int \_tmain(int argc, \_TCHAR\* argv[])

{

float freq2[AlfabetN][AlfabetN] = {0};

char FILENAME[200]={NULL};

SetApp ();

while (1)

{

switch (Menu(1))

{

case '1': fflush(stdin); system("DIR /a:-d"); puts("Введите имя файла\n"); gets(FILENAME);

break;

case '2': entropy\_1( FILENAME);break;

case '3': entropy\_2( FILENAME);break;

case '4': return 0;

default:break;

}

}

/\*Вывод кодов ASCII, для проверки правильности их определения

for (int i = 0; i < AlfabetN; i++)

{

puts("P[%c-%3d] = %4d\n",i,i,P1[i]);

}\*/

system("PAUSE");

return 0;

}

# Задача 2

Запрограммировать процедуру кодирования текстового файла, методом Гилберта-Мура.

Заполнить таблицу:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название текста | Средняя длина кодового слова | Оценка избыточности кодирования | Оценка энтропии выходной последовательности (частоты одиночных символов) | Оценка энтропии выходной последовательности (частоты пар символов) |
| 1.txt(~250байт) | 6 | 1,526 | 0,999 | 0,9909 |
| 2.txt(10байт) | 2,8 | 1,429 | 0,9403 | 0.8862 |
| 3.txt(~11Kбайт) | 6,053 | 1,545 | 0,9982 | 0,9967 |

Таблица 1

Примечание:

Все текстовые файлы и исходный код программы можно просмотреть по адресу - [*https://github.com/jurasikmaniac/TI\_2013/tree/master/Zadacha\_2*](https://github.com/jurasikmaniac/TI_2013/tree/master/Zadacha_2)

## Теория

При кодировании сообщений считается, что символы сообщения порождаются некоторым *источником информации*. Источник считается заданным полностью, если дано вероятностное описание процесса появления сообщений на выходе источника. Это означает, что в любой момент времени определена вероятность порождения источником любой последовательности символов *Р(x*1*x*2*x*3*...xL*), *L*≥1. Такой источник называется *дискретным вероятностным источником.*

Если вероятностный источник с алфавитом *А=*{*a*1*, a*2*, ..., an*}порождает символы алфавита независимо друг от друга, т.е. знание предшествующих символов не влияет на вероятность последующих, то такой источник называется *бернуллиевским*. Тогда для любой последовательности *x1x2...xL*, *L*≥1, порождаемой источником, выполняется равенство:

*P(x*1*x*2*...xL ) = P(x*1*)\*P(x*2*)\*...\*P(xL),*

где *P*(*x*) – вероятность появления символа *х*, *Р*(*x*1*x*2*x*3*...xL*) – вероятность появления последовательности *x*1*x*2*x*3*...xL.*

Для другого класса источников (марковских) существует статистическая взаимосвязь между порождаемыми символами. В дальнейшем мы будем рассматривать кодирование стационарных (с неизменным распределением вероятностей) бернуллиевских дискретных источников без памяти.

Пусть имеется дискретный вероятностный источник без памяти, порождающий символы алфавита *А*={*a1,…,an*} с вероятностями *,* . Основной характеристикой источникаявляется *энтропия,* которая представляет собой среднее значение количества информации в сообщении источника и определяется выражением (для двоичного случая)

.

Энтропия характеризует меру неопределенности выбора для данного источника.

**Пример**. Если *А*={*a*1*,a*2}, *p*1=0, *p*2 =1, т.е. источник может породить только символ a2, то неопределенности нет, энтропия *H(p*1*,p*2*)*=0.

Источник с равновероятными символами А={*a*1*,a*2}, *p*1 =1/2, *p*2 =1/2, будет иметь максимальную энтропию *H*(*p*1*,p*2)=1.

Величина  называется *энтропией на символ*  последовательности длины *L*, где *AL* **−** множество всех последовательностей длины *L*в алфавите*A, x=*(*x*1*,x*2*,...,xL*) **−** последовательность *L*букв дискретного cтационарного источника. Обозначим через предел энтропии *HL* при *L→ ∞* ****. Эту величину называют *предельной энтропией источника***.** Показано, что для стационарного бернуллиевского источника

****.

Для практических применений важно, чтобы коды сообщений имели по возможности наименьшую длину. *Основной характеристикой неравномерного кода* является количество символов, затрачиваемых на кодирование одного сообщения. Пусть имеется разделимый побуквенный код для источника, порождающего символы алфавита А={*a*1*,…,an*} с вероятностями *pi**=P*(*ai*), состоящий из *n* кодовых слов с длинами *L*1*,…,Ln*в алфавите {0,1}. *Средней длиной кодового слова* называется величина , которая показывает среднее число кодовых букв на одну букву источника.

## Код программы

#include "stdafx.h"

#include <cmath>

#include <cstdio>

#include <ctype.h>

#include <clocale>

#include <Windows.h>

//Русский алфавит заглавные

#define rusUp "ЙЦУКЕНГШЩЗХЪФЫВАПРОЛДЖЭЯЧСМИТЬБЮ"

//Русский алфавит прописные

#define rusDown "йцукенгшщзхъфывапролджэячсмитьбю"

//Английский алфавит заглавные

#define engUp "QWERTYUIOPASDFGHJKLZXCVBNM"

//Английский алфавит прописные

#define engDown "qwertyuiopasdfghjklzxcvbnm"

//Количество символов в алфавите

#define n\_Alfa 256

//Длина последовательности кода

#define length\_code 26

//Указатели на открываемые файлы

FILE \*fi;

FILE \*fcode;

//Количество символов в тексте

UINT32 lengthText=0;

//Энтропия файла

float H = (float)0;

//Максимальная энтропия файла

float Hmax = (float)0;

//Средняя длина кодового слова

float Lmean = (float)0;

void readFilename();

UINT8 PaintMainMenu();

void Gilbert\_Mur();

void FreqOne();

void WriteCodeFile();

void entropyBinOne();

//Структура для хранения данных о букве алфавита

struct arrayAlfabeth

{

//ASCII код символа

UINT8 simbol;

//Частота символа в анализируемом тексте

float p;

//Длина кодового слова

UINT8 length;

//Кодовая последовательность Гилберта-Мура

UINT8 code[length\_code];

}ArrAlfa[n\_Alfa];

//Инициализация алфавитного массива

inline void ArrayInit(arrayAlfabeth \*a)

{

for (size\_t i = 0; i < n\_Alfa; i++)

{

a[i].simbol = i;

a[i].p = (float)0;

a[i].length = 0;

memset(a[i].code, 0, length\_code);

}

}

//Установка русской локали

inline void SetLoc(UINT loc)

{

//setlocale( LC\_ALL, "ru\_RU.cp1251" );

//Русская локаль

SetConsoleCP(loc);

SetConsoleOutputCP(loc);

}

//Рисуем пользовательское меню

UINT8 PaintMainMenu()

{

UINT8 index;

system("CLS");

puts("########################");

puts("1. Гилберт-Мур");

puts("2. Записать кодовую последовательность в файл(code.txt)");

puts("3. Оценка энтропии выходной последовательности");

puts("4. Выход");

puts("########################");

puts("Выбрать пункт меню №: ");

index = getchar();

fflush(stdin);

return index;

}

//Считаем энтропию текста, формируем кодовые слова для символом алфавита файла

void FreqOne()

{

UINT8 ch = 0;

UINT8 K = 0;

lengthText = 0;

H = 0;

Lmean = 0;

readFilename();

ArrayInit(ArrAlfa);

while (!feof(fi))

{

fscanf\_s(fi, "%c", &ch);

if (feof(fi))

{

break;

}

if (strchr(rusUp, ch))

{

ch = ch - 0xC0 + 0xE0;

ArrAlfa[ch].p += 1.0;

lengthText++;

}

else

if (strchr(rusDown, ch))

{

ArrAlfa[ch].p += 1.0;

lengthText++;

}

else

if (strchr(engUp, ch))

{

ch = ch - 'A' + 'a';

ArrAlfa[ch].p += 1.0;

lengthText++;

}

else

if (strchr(engDown, ch))

{

ArrAlfa[ch].p += 1.0;

lengthText++;

}

else

if (strchr("-.,:!?;", ch))

{

ch = '.';

ArrAlfa[ch].p += 1.0;

lengthText++;

}

else

if (ch == ' ')

{

ArrAlfa[ch].p += 1.0;

lengthText++;

}

else

if (ch == 0xA8 || ch == 0xB8)

{

ch = 0xE5;

ArrAlfa[ch].p += 1.0;

lengthText++;

}

}

fclose(fi);

printf\_s("The file was closed\n");

for (int i = 0; i < n\_Alfa; i++)

{

if (ArrAlfa[i].p != (float)0)

{

ArrAlfa[i].p = ArrAlfa[i].p / lengthText;

printf\_s("P[%c] = %1.4f\n", i, ArrAlfa[i].p);

H = H + ArrAlfa[i].p \* log2(ArrAlfa[i].p);

K++;

}

}

Hmax = log2((float)K);

if (Hmax == 0)Hmax = 1;

printf\_s("Всего символов в файле: %d\n", lengthText);

printf("Всего символов в алфавите: %d\n", K);

printf("Энтропия файла: %3.3f\n", (-1)\*H);

printf("Максимальная Энтропия файла: %3.3f\n", Hmax);

printf("Избыточность текста составила: %3.3f\n", (float)1 + H / Hmax);

Gilbert\_Mur();

system("pause");

for (int i = 0; i < n\_Alfa; i++)

{

if (ArrAlfa[i].p != (float)0)

{

printf\_s("P[%c] = %1.4f kod = ", i, ArrAlfa[i].p);

for (int j = 0; j<ArrAlfa[i].length; j++) printf("%c", ArrAlfa[i].code[j]);

printf\_s("\n");

}

}

printf("Средняя длина кодового слова: %3.3f\n", Lmean);

printf("Избыточность кодирования составила: %3.3f\n", Lmean - (-1.0)\*H);

system("pause");

}

//Сам Гилберт-Мур

void Gilbert\_Mur()

{

float q[n\_Alfa], pr = 0.0;

q[0] = ArrAlfa[0].p / 2;

ArrAlfa[0].length = (UINT8)(-log2(ArrAlfa[0].p) + 2);

pr = ArrAlfa[0].p;

for (int i = 1; i<n\_Alfa; i++)

{

if (ArrAlfa[i].p!=(float)0)

{

q[i] = pr + ArrAlfa[i].p / 2;

ArrAlfa[i].length = (UINT8)(-log2(ArrAlfa[i].p) + 2);

pr += ArrAlfa[i].p;

}

}

for (int i = 0; i < n\_Alfa; i++)

{

if (ArrAlfa[i].p != (float)0)

{

for (int j = 0; j<ArrAlfa[i].length; j++)

{

q[i] \*= 2.0;

ArrAlfa[i].code[j] = (int)(q[i])+0x30;//превратим 0 и 1 в символы ASCII, для удобства читаемости выходной последовательности

if (q[i]>=1.0) q[i]--; //Если условие оставить условие > то при q=0.75 и p = 0.009 происходит кодировние 11222222!!!!

} //В методичке указано >

}

}

for (int i = 0; i<n\_Alfa; i++)

{

Lmean += ArrAlfa[i].length\*ArrAlfa[i].p;

}

}

void WriteCodeFile()

{

int numclosed;

errno\_t err;

UINT8 ch = 0;

readFilename();

err = fopen\_s(&fcode, "code.txt", "wb");

if (err == 0)

{

printf\_s("The file CODE.txt was opened\n");

}

else

{

printf\_s("The file CODE.txt was not opened\n");

numclosed = \_fcloseall();

}

while (!feof(fi))

{

fscanf\_s(fi, "%c", &ch);

if (feof(fi))

{

break;

}

if (strchr(rusUp, ch))

{

ch = ch - 0xC0 + 0xE0;

fwrite(ArrAlfa[ch].code, ArrAlfa[ch].length, 1, fcode);

}

else

if (strchr(rusDown, ch))

{

fwrite(ArrAlfa[ch].code, ArrAlfa[ch].length, 1, fcode);

}

else

if (strchr(engUp, ch))

{

ch = ch - 'A' + 'a';

fwrite(ArrAlfa[ch].code, ArrAlfa[ch].length, 1, fcode);

}

else

if (strchr(engDown, ch))

{

fwrite(ArrAlfa[ch].code, ArrAlfa[ch].length, 1, fcode);

}

else

if (strchr("-.,:!?;", ch))

{

ch = '.';

fwrite(ArrAlfa[ch].code, ArrAlfa[ch].length, 1, fcode);

}

else

if (ch == ' ')

{

fwrite(ArrAlfa[ch].code, ArrAlfa[ch].length, 1, fcode);

}

else

if (ch == 0xA8 || ch == 0xB8)

{

ch = 0xE5;

fwrite(ArrAlfa[ch].code, ArrAlfa[ch].length, 1, fcode);

}

}

numclosed = \_fcloseall();

system("pause");

}

//Безопасное открытие файла на чтение

void readFilename()

{

errno\_t err;

char FileName[200];

fflush(stdin);

system("DIR /a:-d");

puts("Введите имя файла\n");

while (true)

{

gets\_s(FileName,200);

err = fopen\_s(&fi, FileName, "r");

if (err == 0)

{

printf\_s("The file ""%s"" was opened\n",FileName);

break;

}

else

{

printf\_s("The file ""%s"" was not opened, try again\n",FileName);

}

}

}

//Подсчет энтропии выходной последовательности в файле code.txt Алфавит={0,1}

void entropyBinOne()

{

errno\_t err;

float p1[2] = { 0 };

float p2[2][2] = { 0 };

long total = 0;

long sum = 0;

float entr = 0;

float entr2 = 0;

UINT8 ch = 0;

UINT8 lastch = 0;

err = fopen\_s(&fi, "code.txt", "r");

if (err == 0)

{

printf\_s("The file code.txt was opened\n");

}

else

{

printf\_s("The file code.txt was not opened, try again\n");

}

while (!feof(fi))

{

fscanf\_s(fi, "%c", &ch);

if (feof(fi))

{

break;

}

ch = ch - 0x30;

p1[ch]++;

total++;

}

p1[0] = p1[0] / total;

printf\_s("p[0]=%1.4f\n",p1[0]);

p1[1] = p1[1] / total;

printf\_s("p[1]=%1.4f\n", p1[1]);

entr = p1[0] \* log2f(p1[0]) + p1[1] \* log2f(p1[1]);

entr = entr\*(float)(-1);

ch = 0;

rewind(fi);

fscanf\_s(fi, "%c", &lastch);

lastch = lastch - 0x30;

while (!feof(fi))

{

fscanf\_s(fi, "%c", &ch);

if (feof(fi))

{

break;

}

ch = ch - 0x30;

p2[lastch][ch]++;

lastch = ch;

}

for (int i = 0; i < 2; i++)

{

sum = 0;

for (int j = 0; j < 2; j++)

{

sum = sum + p2[i][j];

}

for (int j = 0; j < 2; j++)

{

p2[i][j] = p2[i][j] / sum;

}

}

for (int i = 0; i < 2; i++)

{

for (int j = 0; j < 2; j++)

{

if (p2[i][j]>0)

{

entr2 = entr2 + p1[i] \* p2[i][j] \* log2(p2[i][j]);

printf("P2[%c][%c]= %f\n", i+0x30, j+0x30, p2[i][j]);

}

}

}

entr2 = entr2\*(float)(-1);

printf\_s("Энтропия кодовой последовательности(частоты одиночных символов) равна = %1.4f\n", entr);

printf\_s("Энтропия кодовой последовательности(частоты пар символов) равна = %1.4f\n", entr2);

system("pause");

fclose(fi);

}

//Тело основной функции программы

int \_tmain(int argc, \_TCHAR\* argv[])

{

int numclosed;

SetLoc(1251);

while (true)

{

switch (PaintMainMenu())

{

case '1':FreqOne(); break;

case '2':WriteCodeFile(); break;

case '3':entropyBinOne(); break;

case '4':numclosed = \_fcloseall();return 0;

default: break;

}

}

numclosed = \_fcloseall();

return 0;

}

# Задача 3

Построить коды Хафмана, Шеннона, Фано, Гилберта-Мура для букв ФИО(Лощаков Юрий Сергеевич)