1. Министерство образования и науки Российской Федерации
2. Санкт-Петербургский политехнический университет Петра Великого
3. —
4. Институт компьютерных наук и технологий
5. **Кафедра «Информационная безопасность компьютерных систем»**

**ОТЧЕТ**

**по лабораторной работе № 8**

По дисциплине «Основы информационной безопасности»

1. Выполнил
2. Студент гр. 13508/13 А.Э. Палёный

\_\_\_\_\_\_\_\_\_\_

1. Проверил
2. Преподаватель

\_\_\_\_\_\_\_\_\_\_ М.О.Калинин

1. Санкт-Петербург
2. 2016

Цель работы:

Приобретение навыков криптоанализа, ознакомление с дешифрованием криптограмм на примере частотного метода.

1. Решаемые задачи:
2. 1) Создать программу, реализующую функции помощника криптоаналитика.
3. 2) ОрРасшифровать одну из криптограмм, выданных преподавателем.
4. Содержание:  
   1) Ход работы (4)
5. 2) Листинг программы (4-8)
6. 3) Контрольные вопросы (9-10)
7. 4) Вывод (10)

Ход работы:  
Исходная криптограмма

САУУЕЛЩА?ЬЦШЕ СЩАВЛЧМАМЛЕУШ (ЛДСНЕ МАУУЕЛЩАЬЦЧЕ ЭАПЩЧЙДЦАЕ, МАУУЕЛЩАЬЦШЕ ЭАПЩШ) - МВЧМЧИ ЭАПЩЧЙДЦАХ, Й СЧЛЧЩЧУ ОТХ

(ГД)ЭАПЩЧЙДЦАХ А ЩДМЭАПЩЧЙДЦАХ ВЩАУЕЦХЕЛМХ ЧОАЦ А ЛЧЛ НЕ СЩАВЛЧКЩДПАЬЕМСАР СТЯЬ. ДЧ АГЧИЩЕЛЕЦАХ МЫЕУШ ДМАУУЕЛЩАЬЦЧКЧ ЭАПЩЧЙДЦАХ ЕОАЦМЛЙЕЦЦШУ

МЪЗЕМЛЙЧЙДЙЭАУ МВЧМЧИЧУ ХЙТХТЧМЖ МАУУЕЛЩАЬЦЧЕ ЭАПЩЧЙДЦАЕ. КТЯЬ ДТКЧЩАЛУД ОЧТНЕЦ МЧЫЩДЦХЛЖМХ Й МЕСЩЕЛЕ ЧИЕАУА МЛЧЩЧЦДУА. КТЯЬ ДТКЧЩАЛУД ЙШИАЩДЕЛМХ

МЛЧЩЧЦДУА ОЧ ЦДЬДТД ЧИУЕЦД МЧЧИЗЕЦАХУА.

В ЦДМЛЧХЗЕЕ ЙЩЕУХ МАУУЕЛЩАЬЦШЕ ЭАПЩШ ЩДГОЕТХЯЛМХ ЦД 2 СТДММД:

1. БТЧЬЦШЕ ЭАПЩШ. ОИЩДИДЛШЙДЯЛ АЦПЧЩУДБАЯ ИТЧСДУА ЧВЩЕОЕТёЦЦЧР ОТАЦШ (ЧИШЬЦЧ 64, 128 ИАЛ), ВЩАУЕЦХХ С ИТЧСЪ СТЯЬ Й ЪМЛДЦЧЙТЕЦЦЧУ

ВЧЩХОСЕ, СДС ВЩДЙАТЧ, ЦЕМСЧТЖСАУА БАСТДУА ВЕЩЕУЕЭАЙДЦАХ А ВЧОМЛДЦЧЙСА, ЦДГШЙДЕУШУА ЩДЪЦОДУА. РЕГЪТЖЛДЛЧУ ВЧЙЛЧЩЕЦАХ ЩДЪЦОЧЙ ХЙТХЕЛМХ

ТДЙАЦЦШР ЮППЕСЛ — ЦДЩДМЛДЯЗДХ ВЧЛЕЩХ МЧЧЛЙЕЛМЛЙАХ ИАЛЧЙ УЕНОЪ ИТЧСДУА ЧЛСЩШЛШЫ А ГДЭАПЩЧЙДЦЦШЫ ОДЦЦШЫ.

2. ПЧЛЧЬЦШЕ ЭАПЩШ, Й СЧЛЧЩШЫ ЭАПЩЧЙДЦАЕ ВЩЧЙЧОАЛМХ ЦДО СДНОШУ ИАЛЧУ ТАИЧ ИДРЛЧУ АМЫЧОЦЧКЧ (ЧЛСЩШЛЧКЧ) ЛЕСМЛД М АМВЧТЖГЧЙДЦАЕУ КДУУАЩЧЙДЦАХ.

ПЧЛЧЬЦШР ЭАПЩ УЧНЕЛ ИШЛЖ ТЕКСЧ МЧГОДЦ ЦД ЧМЦЧЙЕ ИТЧЬЦЧКЧ (ЦДВЩАУЕЩ, ГОСТ 28147-89 Й ЩЕНАУЕ КДУУАЩЧЙДЦАХ), ГДВЪЗЕЦЦЧКЧ Й МВЕБАДТЖЦЧУ ЩЕНАУЕ.

БЧТЖЭАЦМЛЙЧ МАУУЕЛЩАЬЦШЫ ЭАПЩЧЙ АМВЧТЖГЪЯЛ МТЧНЦЪЯ СЧУИАЦДБАЯ ИЧТЖЭЧКЧ СЧТАЬЕМЛЙД ВЧОМЛДЦЧЙЧС А ВЕЩЕМЛДЦЧЙЧС. МЦЧКАЕ ЛДСАЕ ЭАПЩШ АМВЧТЦХЯЛМХ Й

ЦЕМСЧТЖСЧ ВЩЧЫЧОЧЙ, АМВЧТЖГЪХ ЦД СДНОЧУ ВЩЧЫЧОЕ «СТЯЬ ВЩЧЫЧОД». МЦЧНЕМЛЙЧ «СТЯЬЕР ВЩЧЫЧОД» ОТХ ЙМЕЫ ВЩЧЫЧОЧЙ ЦДГШЙДЕЛМХ

«ЩДМВАМДЦАЕУ СТЯЬЕР». КДС ВЩДЙАТЧ, ЧЦЧ МЧГОДЕЛМХ АГ СТЯЬД ЙШВЧТЦЕЦАЕУ ЦДО ЦАУ ЦЕСАЫ ЧВЕЩДБАР, Й ЛЧУ ЬАМТЕ ВЕЩЕМЛДЦЧЙЧС А ВЧОМЛДЦЧЙЧС.

Листинг программы:

#include <stdio.h>

#include <malloc.h>

#include <Windows.h>

#include <locale.h>

/\*

—â®¡ë à ¡®â « àãááª¨© ï§ëª ­ ¤®

¨§¬¥­¨âì èà¨äâ ¢ ª®­á®«¨

\*/

struct Let

{

char text;

int count;

}typedef Alph;

void Automatically()

{

char str[33]={

'®','¥',' ','¨','­','â','á','à','¢','«','ª','¬','¤','¯','ã',

'ï','ë','ì','£','§','¡','ç','©','å','¦','è','î','æ','é','í',

'ä','ê','ñ'

};

}

void ChangeLetter(char \*str, char From, char To)

{

int i;

for(i=0; str[i]!=0; i++)

if(str[i]==From) str[i]=To;

}

void SCopy(char \*str1, char \*str2)

{

int i;

for(i=0; str1[i] != 0; i++)

if(str1[i] != str2[i]) str2[i]=str1[i];

}

char \*GetText(char \*\*Test)

{

int length;

char \*str;

FILE \*Output;

Output=fopen("Test.txt", "rb+");

if(!Output) return NULL;

fseek (Output, 0, SEEK\_END);

length = ftell(Output);

fseek (Output, 0, SEEK\_SET);

str=(char\*)malloc(sizeof(char)\*(length+1));

\*Test=(char\*)malloc(sizeof(char)\*(length+1));

fread(str, 1, length, Output);

fseek (Output, 0, SEEK\_SET);

fread(\*Test, 1, length, Output);

str[length]=0;

\*(\*Test+length)=0;

fclose(Output);

return str;

}

void Frequency(char \*Output)

{

float dump;

int i,r,key;

int Count\_Let=1;

Alph \*str=(Alph\*)malloc(sizeof(Alph));

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

str->text=Output[0];

str->count=0;

for(i=0; Output[i]!=0; i++)

{

key=1;

for(r=0; r<Count\_Let; r++)

if(str[r].text==Output[i]){ str[r].count+=1; r=Count\_Let; key=0;}

if(key)

{

Count\_Let+=1;

str=(Alph\*)realloc(str, sizeof(Alph)\*Count\_Let);

str[Count\_Let-1].text=Output[i];

str[Count\_Let-1].count=1;

}

}

setlocale(LC\_ALL,"Rus");

for(i=0; i<Count\_Let; i++)

printf("Letters %c: %d\n",str[i].text,str[i].count);

}

void Automatic(char \*Output)

{

float dump;

int r,key;

int Count\_Let=1;

//--------------

int i = 0;

int p = 32;

//char n\_alfavit[] = {'о','е','а','и','н','т','р','с','л','в','к','п','м','у','д','я','ы','ь','з','б','г','ю','ч','й','х','ж','ш','щ','ц','ф','э','ъ'};

//char n\_alfavit[] = {'e','t','a','o','i','n','s','h','r','d','l','c','u','m','w','f','g','y','p','b','v','k','x','j','q','z'};

char n\_alfavit[] = {-18,-27,-32,-24,-19,-14,-16,-15,-21,-30,-22,-17,-20,-13,-28,-1,-5,-4,-25,-31,-29,-2,-9,-23,-11,-26,-8,-7,-10,-12,-3};

int a = 0;

int al = 0;

float chastot = 0;

//-----------

Alph \*str=(Alph\*)malloc(sizeof(Alph));

setlocale(LC\_ALL,"Rus");

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

str->text=Output[0];

str->count=0;

for(i=0; Output[i]!=0; i++)

{

key=1;

for(r=0; r<Count\_Let; r++)

if(str[r].text==Output[i]){ str[r].count+=1; r=Count\_Let; key=0;}

if(key)

{

Count\_Let+=1;

str=(Alph\*)realloc(str, sizeof(Alph)\*Count\_Let);

str[Count\_Let-1].text=Output[i];

str[Count\_Let-1].count=1;

}

}

/\*for(i=0; i<Count\_Let; i++)

printf("Letters %c: %d\n",str[i].text,str[i].count);\*/

//-------------------------

while (p != 0)

{

for (int ch = 0; ch < 33; ch++)

{

if (str[ch].count > chastot)

{

chastot = str[ch].count;

a = ch;

}

}

a--;

while (Output[i] != '\0')

{

if (Output[i] == str[a].text)

{

Output[i] = n\_alfavit[al];

}

i++;

}

p--;

str[a+1].count = 0;

chastot = 0;

i = 0;

al++;

}

int u = 0;

setlocale(LC\_ALL,"Rus");

while (Output[u] != '\0')

{

printf("%c", Output[u]);

u++;

}

}

int main()

{

int i;

char c=0;

char e;

char \*Origin, \*Copy;

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

Origin=GetText(&Copy);

if(!Origin) return 0;

printf("Choose mode:\n");

printf("1) Manually\n2) Automatically\n");

scanf("%d",&i);

if(i!=2)

{

while(c!='e')

{

system("cls");

printf("1) Show text\n");

printf("2) Frequency analysis by letter\n");

printf("3) Previously change\n");

printf("4) Change letter\n");

scanf("%d",&i);

system("cls");

switch (i)

{

case 1:

{

printf("%s",Origin);

break;

}

case 2:

{

Frequency(Origin);

break;

}

case 3:

{

SCopy(Copy, Origin);

printf("Done");

break;

}

case 4:

{

SCopy(Origin, Copy);

getchar();

printf("Enter the letter to change:\n");

scanf("%c",&c);

printf("Letter is:\n");

getchar();

scanf("%c",&e);

ChangeLetter(Origin, c, e);

break;

}

default:

{

printf("Incorrect\n");

c='e';

break;

}

}

getchar();

getchar();

}

}else{

Automatic(Origin);

getchar();

getchar();

}

FILE \*Output;

Output=fopen("Test.txt", "wb+");

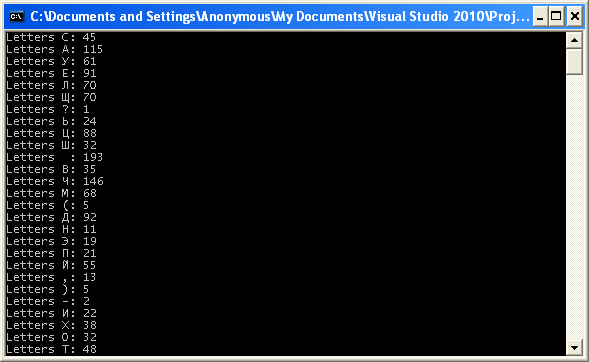
for(i=0; Origin[i]!=0; i++)

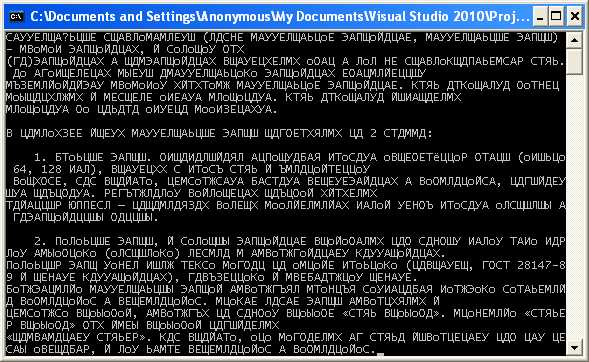
fwrite(&Origin[i], 1, 1, Output);

fclose(Output);

}

3) Результат работы программы:



Заменим букву Ч на о , т.к. Ч самая частая буква.  


После этого начинаем искать и заменять другие буквы.  
После небольших изменений, у нас получился такой текст:

кимметри?чные криптосистемы (также симметричное шифрование, симметричные шифры) - способ шифрования, в котором для (за)шифрования и расшифрования применяется один и тот же криптографический ключ. ао изобретения схемы асимметричного шифрования единственным

существовавшим способом являлось симметричное шифрование. глюч алгоритма должен сохраняться в секрете обеими сторонами. глюч алгоритма выбирается сторонами до начала обмена сообщениями.п настоящее время симметричные шифры разделяются на 2 класса:

1. цлочные шифры. дбрабатывают информацию блоками определённой длины (обычно 64, 128 бит), применяя к блоку ключ в установленном

порядке, как правило, несколькими циклами перемешивания и подстановки, называемыми раундами. йезультатом повторения раундов является

лавинный Юффект — нарастающая потеря соответствия битов между блоками открытых и зашифрованных данных.

2. фоточные шифры, в которых шифрование проводится над каждым битом либо байтом исходного (открытого) текста с использованием гаммирования.

фоточный шифр может быть легко создан на основе блочного (например, здкл 28147-89 в режиме гаммирования), запущенного в специальном режиме.

цольшинство симметричных шифров используют сложную комбинацию большого количества подстановок и перестановок. сногие такие шифры исполняются в

несколько проходов, используя на каждом проходе «ключ прохода». сножество «ключей прохода» для всех проходов называется

«расписанием ключей». гак правило, оно создается из ключа выполнением над ним неких операций, в том числе перестановок и подстановок.

Ответы на контрольные вопросы:

1) Это наиболее простой вид шифрования, заключающийся в замене символов исходного текста на другие (того же алфавита) по более или менее сложному правилу.

2) Данный шифр легко поддаётся взлому. Взломать этот шифр может практически любой человек, без специальной квалификации.

3) Сложность равна размерности алфавита.

4) Большой объём шифротекста. Знание, по какой теме данный текст.

5) Нет, не получится, особенно если размер текста небольшой. Также, автор данного текста, скорее всего учёл это и усложнил нам работу.   
В данном случае автоматическая дешифрация не удалась. Нам пришлось расшифровывать текст самому.