Министерство образования и науки Российской Федерации

Санкт-Петербургский Политехнический Университет Петра Великого

Институт кибербезопасности и защиты информации

**Основы информационной безопасности**

Отчёт по лабораторной работе №2

«Основы частотного криптоанализа»

**Работу выполнила:**

Похожаева Е. М.

**Группа:**

4851001/00002

**Преподаватель:**

Калинин М. О.

Санкт-Петербург

2021

**1. Цель работы**

Приобретение навыков криптоанализа, ознакомление со способом дешифрования криптограмм на примере применения метода частотного криптоанализа.

**2. Теоретические сведения**

Работа алгоритма шифрования при помощи шифра моноалфавитной подстановки заключается в следующем:

* выбирается нормативный алфавит – набор символов, которые будут использоваться для составления сообщений
* определяется алфавит шифрования путём взаимооднозначного соответствия с символами нормативного алфавита

Моноалфавитные подстановки обладают важным свойством – они не нарушают частот появления символов, характерных для данного языка. Все естественные языки имеют характерное частотное распределение символов. Например, буква «О» встречается в русском языке чаще других, а буква «Ъ» − самая редкая.

Для того, чтобы получить открытый текст при помощи частотного анализа, необходимо сопоставить частоты появления символов шифра с вероятностями появления букв используемого алфавита. После этого наиболее частые символы криптограммы заменяются наиболее вероятными символами алфавита, остальные замены производятся на основе вероятных слов и знания синтаксических правил используемого языка.

**3. Ход работы**

* **Исходный текст криптограммы:**

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

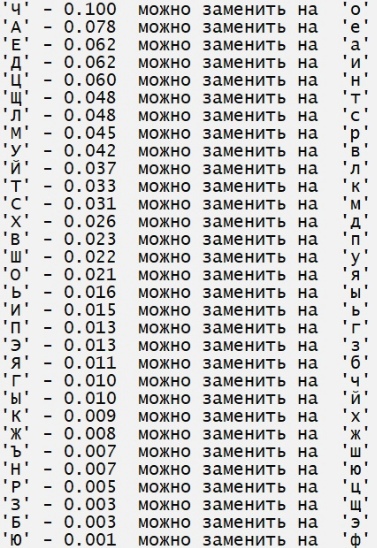
* **Частотный анализ текста.**

Для реализации этой функции потребовались следующие действия:

1. Создаётся массив русского алфавита в зависимости от частоты встречаемости каждой буквы (от самой встречаемой до самой редкой)
2. Создаётся массив, заполненный символами зашифрованного текста
3. Создаётся массив с подсчётом встречаемости каждого элемента символьного массива в тексте
4. Символьный массив сортируется в соответствии с количеством повторений одного и того же символа в тексте
5. Выводится символ, его частота встречаемости, а также буква русского алфавита с приблизительной частотой встречаемости в тексте

Все операции выполняются в функции analitika.

Результат её работы:

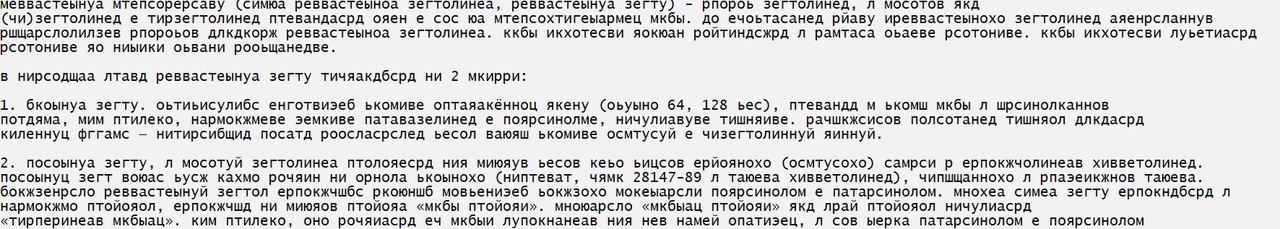


* **Автозамена.**

Эта функция заменяет символы согласно частотному анализу текста

Реализуется в функции analitika

Результат её работы:



* **Вывод слов в порядке возрастания нерасшифрованных букв в слове.**

Для реализации этой функции потребовались следующие действия:

1. Создаётся матрица, каждая строка которой это отдельное слово. В таком случае словом считается набор символов, ограниченных пробелами, \t, \n с двух сторон
2. Создаётся массив с количеством нерасшифрованных символов в каждом слове
3. На основе пункта 2 матрица сортируется и выводится на экран построчно

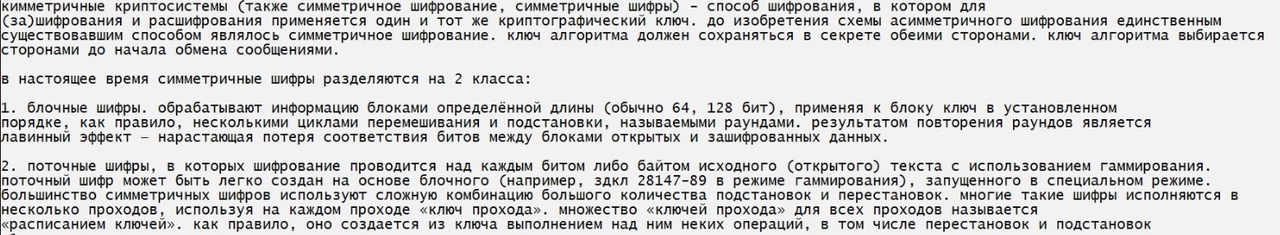
Всё это реализуется в функции zumotsor

* **Замена букв и её отмена. История. Ключ.**

Эти две возможности реализуются в функции zumotsor. Осуществляются с помощью истории, которая представляет из себя два символьных массива (первый – символы, которые заменили; второй – символы, на которые заменили). Когда производится замена, то в эти массивы записываются данные. С помощью счётчика, который отвечает за очередь действий, можно осуществить отмену, то есть из счётчика вычесть единицу и сделать действия обратные замене (при замене к счётчику прибавляют единицу).

Ключ – вывод двух массивов.

* **Результат (расшифрованный текст)**



* **Ключ**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| А | Б | В | Г | Д | Е | Ж | З | И | Й | К | Л | М | Н | О | П | Р |
| Д | И | Й | К | О | Е | Н | Г | А | Р | С | Т | У | Ц | Ч | В | Щ |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| С | Т | У | Ф | Х | Ц | Ч | Ш | Щ | Ь | Э | Ю | Я |
| М | Л | Ъ | П | Ы | Б | Ь | Э | З | Ж | Ю | Я | Х |



**4. Контрольные вопросы.**

1. *Что такое шифр моноалфавитной подстановки?*

Моноалфавитный шифр подстановки — шифр, при котором каждый символ открытого текста заменяется на некоторый, фиксированный при данном ключе, символ того же алфавита.

1. *Укажите недостатки шифра моноалфавитной подстановки.*

Так как буквы в исходном тексте и зашифрованном тексте находятся в отношении один к одному, данный текст легко расшифровать при помощи частотного анализа

1. *Какова сложность дешифрации методом прямого перебора для сообщения, зашифрованного шифром моноалфавитной подстановки?*

Сложность дешифрации методом прямого перебора зависит от количества всех возможных решений задачи.

В зависимости от количества всех возможных решений прямой перебор может потребовать экспоненциального времени работы.

1. *Какие условия упрощают частотный анализ?*

Частотный анализ может зависеть от особенностей языка шифрования: частота встречаемости пар букв, союзы, предлоги и т.д.

1. *Получится ли правильно расшифрованная криптограмма, если произвести все замены в соответствии с частотами появления букв в русском языке?*

Нет, так как частота употребления символа может зависеть от характера текста и не соответствовать частоте появления букв в русском языке, поэтому последующие замены должны производится на основе вероятных слов и знаний синтаксических правил языка.

**5. Вывод.**

В ходе лабораторной работы был изучен шифр моноалфавитной подстановки, его свойства, а также методы частотного анализа, которые применяются для расшифровки моноалфавитного шифра в криптографии.

Код программы:

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <locale.h>

#include <windows.h>

char text[100000], rus[100], cript[100], j, s[100], rusa[] = "оеаинтсрвлкмдпуяыьгзбчйхжшюцщэфъё";;

int y = 0, len;

void key()

{

int i = 0;

for (i = 0; cript[i] != '\0'; i++) printf("%c = %c\n", rus[i], cript[i]);

}

void reset()

{

int i, j;

for (i = 0; i < 1000; i++) text[i] = '\0';

for (i = 0; i < 100; i++)

{

rus[i] = '\0';

cript[i] = '\0';

s[i] = '\0';

}

}

void zumotsor(int len, int vu)

{

int i = 0, j = 0, v = 0, w = 0, k, max = 0, ind, q, nword;

char msimb[1000][100];

int nsimb[1000];

if (vu == 5)

{

for (i = 0; i < 1000; i++)

{

for (j = 0; j < 100; j++) msimb[i][j] = '\0';

}

for (i = 0; i < 1000; i++) nsimb[i] = '\0';

for (i = 0; i < len; i++)

{

if (text[i] == ' ')

{

if (j > 0) nsimb[v] = j;

v++;

w = 0;

j = 0;

}

else

{

msimb[v][w] = text[i];

j++;

for (q = 0; q < 33; q++)

{

if (msimb[v][w] == rusa[q]) j--;

}

w++;

}

if ((i == len - 1) && (j > 0)) nsimb[v] = j;

}

nword = v + 1;

for (k = 0; k < nword; k++)

{

for (i = k; i < nword; i++)

{

if (nsimb[i] >= max)

{

max = nsimb[i];

strcpy (s, msimb[i]);

ind = i;

}

}

nsimb[ind] = nsimb[k];

strcpy (msimb[ind], msimb[k]);

nsimb[k] = max;

strcpy(msimb[k], s);

max = 0;

}

for (i = nword - 1; i >= 0; i--) puts(msimb[i]);

}

if (vu == 3)

{

printf("Введите букву, которую нужно заменить: ");

getchar();

scanf("%c", &cript[y]);

printf("Введите букву, на которую нужно заменить: ");

getchar();

scanf("%c", &rus[y]);

for (i = 0; i < len; i++)

{

if (text[i] == cript[y]) text[i] = rus[y];

printf("%c", text[i]);

}

printf("\n");

y++;

}

if (vu == 4)

{

if (y == 0) printf("error\n");

else

{

y--;

for (i = 0; i < len; i++)

{

if (text[i] == rus[y]) text[i] = cript[y];

printf("%c", text[i]);

}

printf("\n");

}

}

}

void analitika(int len, int vu)

{

int i=0, j, z, k = 0, o = 0, nz = 0, sum = 0, krai = 0, kl;

char rusalf[] = " оеаинтсрвлкмдпуяыьгзбчйхжшюцщэфъё";

char a, zn, bkrai, bmax, c;

char simbol[100] = { 0 };

double n[100] = { 0 }, zkrai, zmax;

int nn[100] = { 0 }, max = 0, ind = 0;

for (i = 0; i < len; i++)

{

a = text[i];

kl = a;

if (((kl < -32) && (kl > -65)) || (kl == 32))

{

for (z = 0; z < o; z++)

{

if (a == simbol[z]) k++;

}

if (k == 0)

{

simbol[o] = a;

o++;

}

else k = 0;

}

}

for (i = 0; i < o; i++)

{

a = simbol[i];

for (j=0; j<len; j++)

{

zn = text[j];

if (a == zn) nz++;

}

n[i] = nz;

nn[i] = nz;

sum += nz;

nz = 0;

}

for (k = 0; k < o; k++)

{

for (i = k; i < o; i++)

{

if (nn[i] > max)

{

max = nn[i];

zmax = n[i];

bmax = simbol[i];

ind = i;

}

}

krai = nn[k];

zkrai = n[k];

bkrai = simbol[k];

nn[ind] = krai;

n[ind] = zkrai;

simbol[ind] = bkrai;

nn[k] = max;

n[k] = zmax;

simbol[k] = bmax;

max = 0;

}

if (vu == 1)

{

for (i = 1; i < o; i++)

{

printf("'%c' - %.3lf можно заменить на '%c'\n", simbol[i], n[i] / sum, rusalf[i]);

}

}

if (vu == 6)

{

int zxc=0;

for (i = 0; i < len; i++)

{

for (j = 0; j < o; j++)

{

if (text[i] == simbol[j])

{

printf("%c", rusalf[j]);

zxc = 1;

}

}

if (zxc == 0) printf("%c", text[i]);

zxc = 0;

}

printf("\n");

}

}

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

char c;

int vu;

reset();

FILE\* input = fopen("TextFile1.txt", "r");

int i = 0, len = 0, j = 0, v = 0, w = 0, paint;

for (c = fgetc(input); c != EOF; c = fgetc(input))

{

if (c != '?')

{

text[i] = c;

i++;

}

else i++;

}

fclose(input);

len = i;

for (i = 0; i < len; i++)

{

if (text[i] == '.')

{

while (text[i + 1] == ' ' || text[i + 1] == '\n' || text[i + 1] == '\t' || text[i + 1] == '2' || text[i + 1] == '.')

{

i++;

}

text[i + 1] = text[i + 1] + 32;

}

}

for (;;)

{

printf(" 1 - анализ текста\n");

printf(" 2 - вывести текст\n");

printf(" 3 - замена буквы\n");

printf(" 4 - отменить замену\n");

printf(" 5 - вывод слов в порядке возрастания нерасшифрованных букв в слове\n");

printf(" 6 - автозамена\n");

printf(" 7 - ключ\n");

printf(" 8 - завершить работу\n");

scanf("%d", &vu);

if (vu == 1 || vu == 6) analitika(len, vu);

if (vu == 2)

{

for (i = 0; i < len; i++)

{

printf("%c", text[i]);

}

printf("\n");

}

if (vu == 3 || vu == 4 || vu == 5) zumotsor(len, vu);

if (vu == 7) key();

if (vu == 8) return 0;

}

}