БЕЛОРУССКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

***Факультет информационных технологий и робототехники***

Кафедра Программного обеспечения информационных систем и технологий

**Отчет по лабораторной работе № 2**

по дисциплине: «Теория информации»

на тему: ***«Использование частотного метода в криптографических системах работающих по методу подстановки»***

Выполнили студентки группы 10701118:Д.В.Алейников

И.А.Воробей

Преподаватель доц.**:** А.А. Нессенчук

Минск 2019

**Цель работы:** Взломать шифр с применением частотного статистического метода.

1. **Краткие теоретические сведения**

Криптографические системы, основанные на методе подстановки, разделяются на четыре основных класса:

1) monoalphabetic – символ исходного текста заменяется другим символом таким образом, что между ними существует однозначное соответствие;

2) homophonic – имеется несколько вариантов замены исходного символа;

3) polyalphabetic – основаны на использовании нескольких различных ключей;

4) polygram – характеризуются подстановкой не одного, а нескольких символов в исходном тексте. В общем случае n символов исходного текста заменяются n символами шифротекста.

В лабораторной работе №1 была использована криптографическая система класса monoalphabetic, где был использован метод подстановки основанный на умножении номера каждого символа исходного текста на значение ключа k1 и суммирование ключа k0, описываемый отношением:

*Ek(i) = (i\*k*1 *+ k*0*)* mod *n*, (1.1)

где *i* - номер символа исходного текста, *n* - количество символов в исходном алфавите (*n* = 26 для английского алфавита и *n* = 256 для ASCII-кодов), *k*, *k0* - ключ, (*n*, *k*)=1.

Криптоанализ (от др.-греч. κρυπτός — скрытый и анализ) — наука о методах расшифровки зашифрованной информации без предназначенного для такой расшифровки ключа.

Термин был введён американским криптографом Уильямом Ф. Фридманом в 1920 году в рамках его книги «Элементы криптоанализа». Неформально криптоанализ называют также взломом шифра.

В большинстве случаев под криптоанализом понимается выяснение ключа; криптоанализ включает также методы выявления уязвимости криптографических алгоритмов или протоколов.

Попытку раскрытия конкретного шифра с применением методов криптоанализа называют криптографической атакой на этот шифр. Криптографическую атаку, в ходе которой раскрыть шифр удалось, называют взломом или вскрытием.

Частотный анализ — основной инструмент для взлома большинства классических шифров перестановки или замены. Данный метод основывается на предположении о существовании нетривиального статистического распределения символов, а также их последовательностей одновременно и в открытом тексте, и в шифротексте. Причём данное распределение будет сохраняться с точностью до замены символов как в процессе шифрования, так и в процессе дешифрования.

Стоит отметить, что при условии достаточно большой длины шифрованного сообщения моноалфавитные шифры легко поддаются частотному анализу: если частота появления буквы в языке и частота появления некоторого присутствующего в шифротексте символа приблизительно равны, то в этом случае с большой долей вероятности можно предположить, что данный символ и будет этой самой буквой.

Самым простым примером частотного анализа может служить банальный подсчёт количества каждого из встречающихся символов, затем следуют процедуры деления полученного числа символов на количество всех символов в тексте и умножение результата на сто, чтобы представить окончательный ответ в процентах. Далее полученные процентные значения сравниваются с таблицей вероятностного распределения букв для предполагаемого языка оригинала.

Ниже приведена таблица встречаемости букв в латинском алфавите.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Высокая | | Средняя | | Низкая | |
| Буква | Частота,  % | Буква | Частота,  % | Буква | Частота,  % |
| E | 12,31 | L | 4,03 | B | 1,62 |
| T | 9,59 | D | 3,65 | G | 1,61 |
| A | 8,05 | C | 3,20 | V | 0,93 |
| O | 7,94 | U | 3,10 | K | 0,52 |
| N | 7,19 | P | 2,29 | Q | 0,20 |
| I | 7,18 | F | 2,28 | X | 0,20 |
| S | 6,59 | H | 2,25 | Y | 0,10 |
| R | 6,03 | W | 2,03 | Z | 0,09 |
| H | 5,14 | Y | 1,88 |  |  |

Таблица 1 – частота появлений букв в латинском алфавите

1. **Задание по лабораторной работе**
2. Сформировать исходный текст из символов определённого алфавита (не менее одной страницы).
3. Зашифровать сформированный текст методом подстановки, рассмотренным в лабораторной работе №1 (по формуле (1.1)).
4. Взломать полученный шифр используя частотный метод. Для этого:
   * подсчитать частоту встречаемости каждой буквы в шифротексте;
   * сравнить полученные значения частот с табличным значением (таблица 1);
   * осуществить взлом шифротекста частотным методом с использованием логико-смыслового анализа исходных данных задачи.
5. **Алгоритм решения поставленной задачи**

На рисунке 1 представлена блок-схема общей логики программы

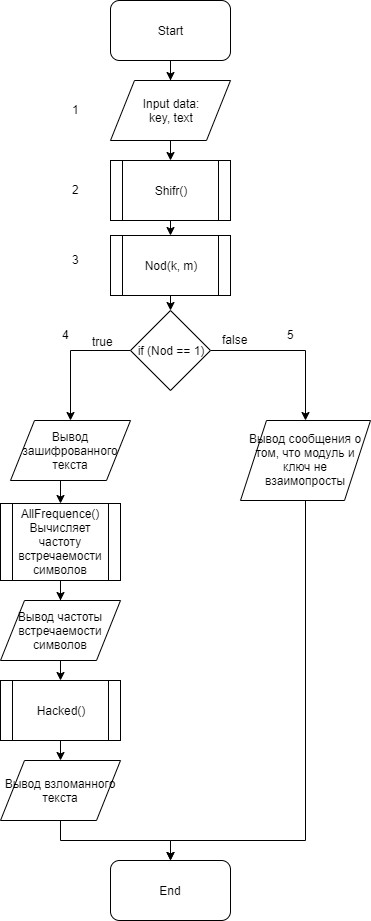


Рисунок 1 - Блок-схема алгоритма общей логики программы

Блок-схема 1 отображает следующую последовательность действий:

1. Начало работы алгоритма, ввод исходных данных (ключ, текст)
2. Шифрование исходного текста при вызове функции Shifr()
3. Проверка ключа и модуля на взаимную простоту при вызове функции Nod(k,m)
4. Если НОД(k, m) = 1, то происходит вывод зашифрованного текста на экран, вычисление частоты встречаемости каждого зашифрованного символа, вывод ее на экран, взлом зашифрованного текста
5. Если НОД(k, m) не равен 1, то выводится сообщение, о том, что модуль и ключ не являются взаимопростыми и просьбой, изменить ключ.

На рисунке 2 представлена блок-схема алгоритма, по которому осуществляется шифрование исходного текста.

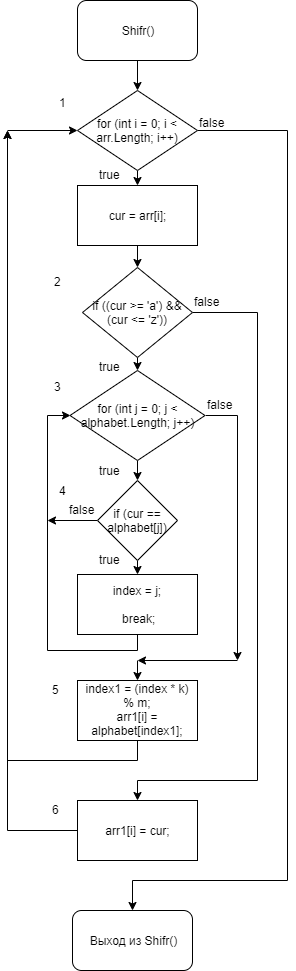


Рисунок 2 - Блок-схема алгоритма шифрования текста

Блок-схема шифрования на рисунке 2 изображает следующую последовательность действий:

1. Организуется цикл для шифрования каждого символа исходного текста, т.е. пока весь текст не будет обработан, выполняться алгоритм по ветке True, в противном случае по ветке False.
2. Проверка на небуквенный символ. Если результат пункта 3 – True, то программа пойдет по ветке True, в противном случае – по ветке False.
3. Если символ не равен небуквенному символу, то организуется цикл для нахождения соответствия символа исходного текста символу алфавита.
4. Если символ исходного текста равен символу алфавита, вычисляется числовое **i** зашифрованного символа исходного текста.
5. По формуле определяется значение символа по **i** зашифрованного символа и добавляется в переменную для зашифрованного текста.
6. В переменную для зашифрованного текста добавляется небуквенный символ.

На рисунке 3 представлена блок-схема алгоритма, осуществляющего взлом шифротекста частотным методом.

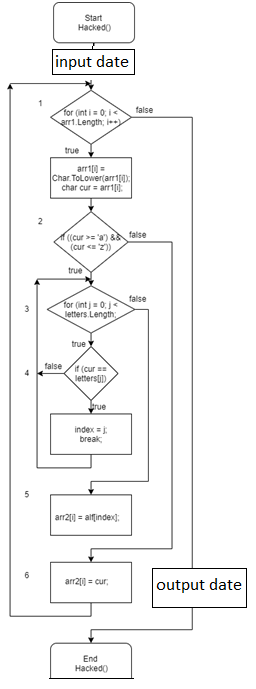


Рисунок 3 - Блок-схема алгоритма взлома шифротекста

При взломе зашифрованного текста была следующая последовательность действий:

1. Организуется цикл для взлома каждого символа исходного текста, т.е. пока весь текст не будет обработан, выполняться алгоритм по ветке True, в противном случае по ветке False.
2. Проверка на небуквенный символ. Если результат пункта 2 – True, то программа пойдет по ветке True, в противном случае – по ветке False.
3. Если символ не равен небуквенному символу, то организуется цикл для нахождения индекса, по которому символ зашифрованного текста соответствует символу списка, отсортированного в порядке возрастания встречаемости символов в зашифрованном тексте.
4. Если символ зашифрованного текста равен символу алфавита, вычисляется индексзашифрованного символа в сортированном списке.
5. В переменную для взломанного текста по найденному индексу добавляется символ из таблицы частоты встречаемости.
6. В переменную для взломанного текста добавляется небуквенный символ.

На рисунке 4 представлена блок-схема замены букв взломанного текста пользователем.

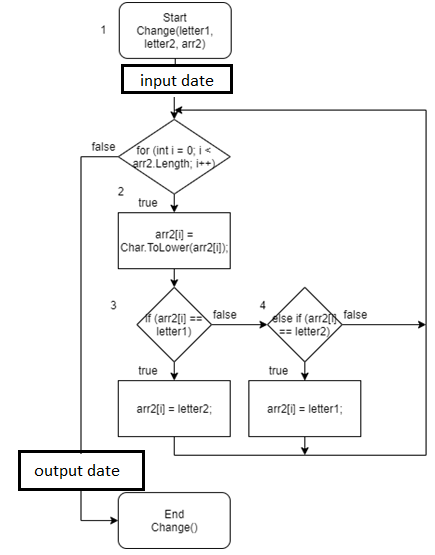


Рисунок 4 – Блок-схема пользовательской замены символа

Описание блок-схемы алгоритма пользовательской замены символов, изображенного на рисунке 4:

1. Данная функция принимает arr2 (взломанный текст), letter1 (замещаемая буква), letter2 (буква на которую заменяем).
2. Организуется цикл, в результате которого замещаемая буква в тексте приводится к нижнему регистру.
3. Если символ взломанного текста равен letter1, то мы заменяем его на letter2.
4. Если же символ взломанного текста равен letter2, то мы заменяем его на letter1.
5. **Описание входных/выходных данных программы**

Входные данные:

* Любой набор символов латинского алфавита
* Число, являющееся ключом шифра

Выходные данные:

* Зашифрованный текст
* Взломанный текст
* Частота встречаемости букв в тексе

1. **Контрольные примеры**

Контрольные примеры приведены на рисунках 5 и 6.

В программе использованы следующие значения:

* Ключ = 3
* Модуль = 26
* Исходный текст

НОД (Ключ 1, Модуль) = 1

На рисунке 4 в результате шифрования текста по формуле 1.1 был получен зашифрованный текст.

В правой части рисунка 4 размещена колонка «Frequence», в которой расположены буквы и информация к ним о частоте встречаемости на основе зашифрованного текста.

Под колонкой «Hacked text» расположена кнопка «change», в первое поле вводим букву, которую хотим заменить, а во второе поле вводим букву, на которую заменим.

На рисунке 5 в результате замены букв пользователем были получены некоторые слова, совпадающие с исходным текстом.

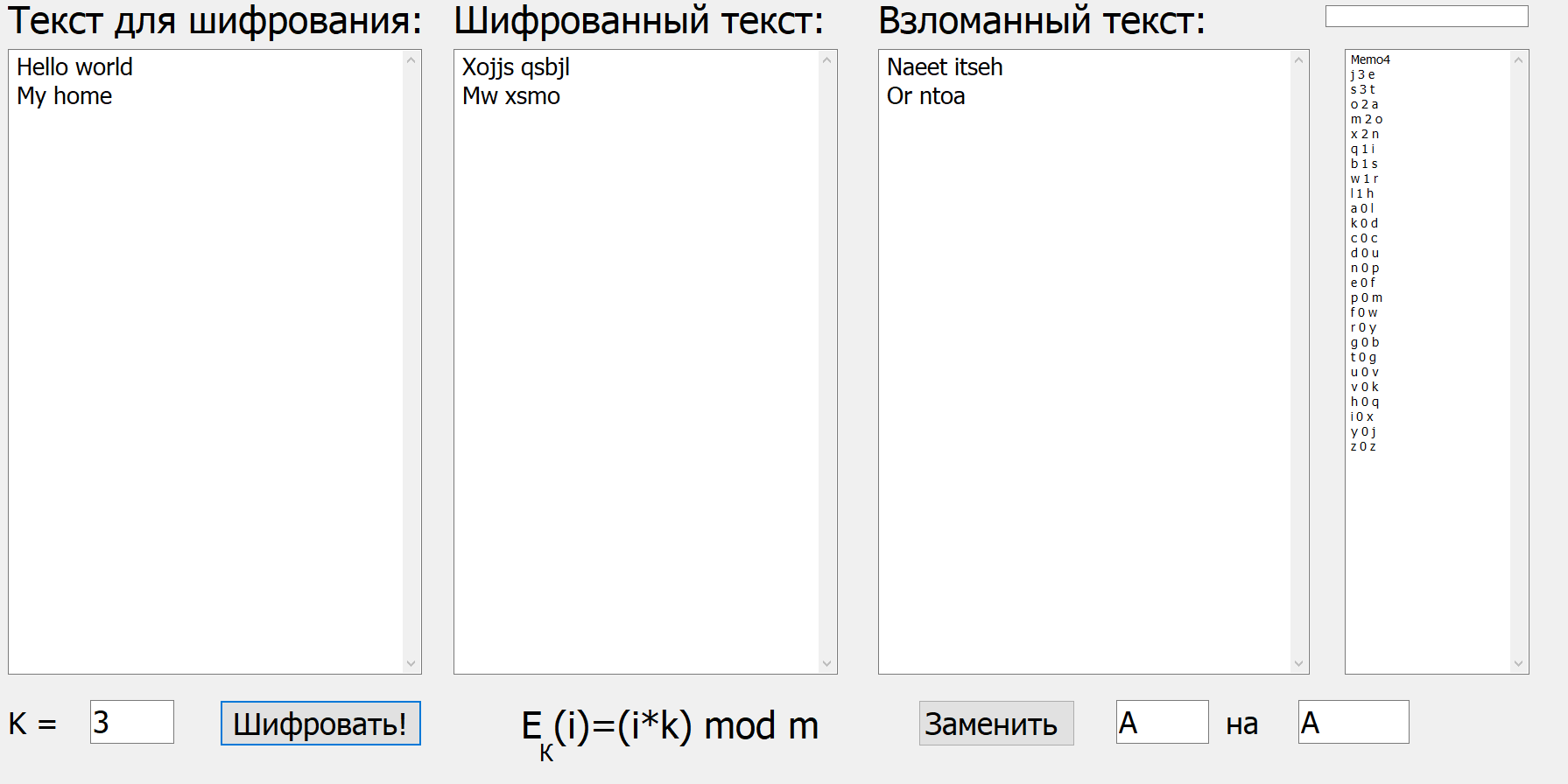


Рисунок 5 - Пример выполнения программы (начало)

1. **Выводы**
2. Изучили взламывать шифр с применением частотного статистического метода.
3. Разработали программу, выполняющую шифрование, а также взламывать зашифрованный текст с применением частотного статистического метода.

**Список использованной литературы**

1. Чмора А.А. Современная прикладная криптография. М., 2001.
2. Романец Ю.В., Тимофеев П.А., Шаньгин В.Ф. Защита информации в компьютерных системах и сетях. М., 2001.
3. Введение в криптографию. Под общей редакцией В.В. Ященко. М., 1999.

**ПРИЛОЖЕНИЕ A**

**Код программы для шифрования и взлома на основе частотного анализа**

**unit Unit1;**

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls;

type

TForm1 = class(TForm)

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

Label4: TLabel;

Memo1: TMemo;

Memo2: TMemo;

Memo3: TMemo;

Edit1: TEdit;

Edit2: TEdit;

Label7: TLabel;

Label8: TLabel;

Button1: TButton;

Button2: TButton;

Label6: TLabel;

Edit3: TEdit;

Memo4: TMemo;

Edit4: TEdit;

procedure FormCreate(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure Button1Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

Type

TSymbol = record

lett : char;

count : integer;

lett\_ans : char;

end;

var

symb : array [1..26] of TSymbol;

m : Integer;

alf : string;

var

Form1: TForm1;

implementation

{$R \*.dfm}

procedure TForm1.FormCreate(Sender: TObject);

begin

Form1.WindowState:=wsMaximized;

m:=26;

alf:='etaonisrhldcupfmwybgvkqxjz';

Memo1.Clear;

Memo1.Lines.Add('Çäåñü ââîäèòñÿ èñõîäíûé òåêñò.');

Memo2.Clear;

Memo2.Lines.Add('Çäåñü âûâîäèòñÿ çàøèôðîâàííûé òåêñò.');

Memo3.Clear;

Memo3.Lines.Add('Çäåñü âûâîäèòñÿ ðàñøèôðîâàííûé òåêñò.');

Edit1.Text:='A';

Edit2.Text:='A';

Edit4.Text:=' Èíôîðìàöèÿ î âçàèìíîïðîñòîòå';

end;

procedure LETTER();

var

temp : TSymbol;

i, j : Integer;

begin

for i:=1 to m-1 do

for j:=i+1 to m do

if symb[i].count<symb[j].count then

begin

temp:=symb[i];

symb[i]:=symb[j];

symb[j]:=temp;

end;

for i:=1 to m do

symb[i].lett\_ans:=alf[i];

end;

function NOD(a, b: Integer):Integer;

begin

while (a<>0) and (b<>0) do

if a>b then a:=a mod b

else b:=b mod a;

Result:=a+b;

end;

function NEW\_CHR(a : char; k : Integer) : Integer;

var

b:integer;

begin

b:=((Ord(a)-96)\*k) mod m;

if b=0 then b:=m;

Result:=(b+96);

end;

procedure TForm1.Button2Click(Sender: TObject);

var

i, j, k, nodk, f, q: Integer;

str, str2 : String;

c : char;

begin

for i:=1 to 26 do

with symb[i] do

begin

lett:=chr(i+96);

count:=0;

lett\_ans:='#';

end;

// ØÈÔÐÎÂÊÀ

k:=StrToInt(Edit3.Text);

nodk:=NOD(k,m);

if (nodk<>1) then

begin

Edit4.Clear;

Edit4.Text := 'Íå âçàèìíî ïðîñòîé!';

end

else

Edit4.Clear;

if Memo1.Lines[0]<>'' then

begin

Memo2.Clear;

for i:=0 to Memo1.Lines.Count-1 do

begin

str:=Memo1.Lines[i];

str2:=str;

for j:=1 to Length(str) do

if (((str[j]>='A') and (str[j]<='Z')) or ((str[j]>='a') and (str[j]<='z'))) then

begin

f:=0;

if ((str[j]>='A') and (str[j]<='Z')) then

begin

str[j]:=Chr(Ord(str[j])+32);

f:=32;

end;

str2[j]:=Chr(NEW\_CHR(str[j],k)-f);

end;

Memo2.Lines.Add(str2);

end;

//ÐÀÑØÈÔÐÎÂÊÀ

Memo3.Clear;

for i:=0 to Memo2.Lines.Count-1 do

begin

str:=Memo2.Lines[i];

for j:=1 to Length(str) do

if (((str[j]>='A') and (str[j]<='Z')) or ((str[j]>='a') and (str[j]<='z'))) then

begin

if ((str[j]>='A') and (str[j]<='Z'))

then c:=Chr(Ord(str[j])+32)

else c:=str[j];

for q:=1 to m do

if (symb[q].lett=c) then

begin

Inc(symb[q].count);

break;

end;

end;

end;

LETTER();

for i:=1 to m do

with symb[i] do

Memo4.Lines.Add(lett+' '+IntToStr(count)+' ' + lett\_ans);

for i:=0 to Memo2.Lines.Count-1 do

begin

str:=Memo2.Lines[i];

str2:=str;

for j:=1 to Length(str) do

if (((str[j]>='A') and (str[j]<='Z')) or ((str[j]>='a') and (str[j]<='z'))) then

begin

f:=0;

if ((str[j]>='A') and (str[j]<='Z'))

then begin

c:=Chr(Ord(str[j])+32);

f:=32;

end else c:=str[j];

for q:=1 to m do

if c=symb[q].lett then

begin

str2[j]:=Chr(Ord(symb[q].lett\_ans)-f);

end;

end;

Memo3.Lines.Add(str2);

end;

end;

end;

procedure TForm1.Button1Click(Sender: TObject);

var

a, b, c :char;

i, j, f: Integer;

str : String;

begin

a:=Edit1.Text[1];

if (((a>='A') and (a<='Z')) or ((a>='a') and (a<='z'))) then

if ((a>='A') and (a<='Z'))

then a:=Chr(Ord(a)+32);

b:=Edit2.Text[1];

if (((b>='A') and (b<='Z')) or ((b>='a') and (b<='z'))) then

if ((b>='A') and (b<='Z'))

then b:=Chr(Ord(b)+32);

for i:=1 to m do

if (symb[i].lett\_ans=a) or (symb[i].lett\_ans=b) then

if symb[i].lett\_ans=a

then symb[i].lett\_ans:=b

else symb[i].lett\_ans:=a;

if Memo3.Lines[0]<>'' then

for i:=0 to Memo3.Lines.Count-1 do

begin

str:=Memo3.Lines[i];

for j:=1 to Length(str) do

if (((str[j]>='A') and (str[j]<='Z')) or

((str[j]>='a') and (str[j]<='z'))) then

begin

f:=0;

if ((str[j]>='A') and (str[j]<='Z'))

then begin

c:=Chr(Ord(str[j])+32);

f:=32;

end else c:=str[j];

if c=a then str[j]:=Chr(Ord(b)-f);

if c=b then str[j]:=Chr(Ord(a)-f);

end;

Memo3.Lines[i]:=str;

end;

end;

end.