**1.Реализовать в программе шифрование и дешифрацию содержимого файла по методу Цезаря с ключевым словом.**

**(С++)**

**const alf='ABCDEFGHIJKLMNOPQRSTUVWXYZ';*//алфавит***

**var**

**Form1: TForm1;**

**s,s1:string;**

**implementation**

***{$R \*.dfm}***

**procedure TForm1.Button1Click(Sender: TObject);**

**var k,i,j:integer;**

**kl,s2:string;**

**begin**

**s:=AnsiUpperCase(Edit1.Text);**

**kl:=AnsiUpperCase(Edit1.Text);**

**Val(Edit3.Text,k,i);**

**if(i<>0)or not(k in [0..24])then**

**begin**

**ShowMessage('Неверно введено смещение');**

**Edit3.Clear;**

**Edit3.SetFocus;**

**Exit;**

**end;**

**s1:=kl; *//пишем в строку ключ***

**for i:=1 to k do s1:=' '+s1;*//в начало 17 пробелов***

**s2:='';*//алфавит без букв ключевого слова***

**for i:=1 to 26 do**

**if pos(alf[i],kl)=0 then s2:=s2+alf[i];**

**i:=length(s1)+1;*//конец строки+1***

**j:=length(s1);*//длина строки***

**while length(s1)<=26 do *//пока не 26***

**begin**

**s1:=s1+s2[i-j];*//добавляем первые буквы алфавита***

**i:=i+1;**

**end;**

**while s1[1]=' ' do delete(s1,1,1);**

**s1:=copy(s2,i-j,length(s2))+s1;**

***//шифрование***

**for i:=1 to length(s) do**

**s[i]:=s1[pos(s[i],alf)];**

**Edit4.Text:=s;**

**end;**

**procedure TForm1.Button2Click(Sender: TObject);**

**var i:Integer;**

**begin**

**for i:=1 to length(s) do**

**s[i]:=alf[pos(s[i],s1)];**

**Edit5.Text:=s;**

**end;**

**2. Реализовать шифрование и дешифрацию файла по методу Виженера. Ключевая фраза вводится. Реализовать в программе частотный криптоанализ зашифрованного текста.**

**(С++)**

**#define \_CRT\_SECURE\_NO\_DEPRECATE**

**#include <iostream>**

**#include <stdio.h>**

**#include <conio.h>**

**using namespace std;**

**int main()**

**{**

**setlocale(LC\_ALL, "RUSSIAN");**

**//определение файлов ввода -вывода**

**FILE \*f1;**

**FILE \*f2;**

**FILE \*f3;**

**char fname[100];**

**char key[100];**

**char c;**

**int cur = 0;**

**f1 = fopen("file.txt", "r");**

**f2 = fopen("shifr.txt", "w");**

**if (f1 == NULL) //если входной файл пустой вывести след.**

**{**

**printf("Файл не найден\n");**

**return 0;**

**}**

**printf("Ключ: ");//иначе предложить ввод ключа**

**scanf("%s", key); //строка символов %s, key**

**/////////////////////////////////////////////шифровка**

**while (!feof(f1))//файлы открываются в двоичном режиме,feof() используется для проверки наличия конца файла**

**{**

**if (fread(&c, 1, 1, f1)) //указатель, размер, считать 1 символ из файла f1**

**{**

**if ((c >= 97) && (c <= 122)) //**

**{**

**if ((c + key[cur] - 97) <= 122) c = c + key[cur] - 97; else**

**c = key[cur] + c - 123;**

**fprintf(f2, "%c", c);**

**}**

**else if (c == 32)**

**{**

**fprintf(f2, "%c", c);**

**}**

**else**

**{**

**printf("только от a-z");**

**break;**

**}**

**}**

**}**

**fclose(f1);**

**fclose(f2);**

**//////////////////////////////////////////дешифровка**

**f3 = fopen("deshifr.txt", "w+");**

**f2 = fopen("shifr.txt", "r");**

**cur = 0;**

**while (!feof(f2))**

**{**

**if (fread(&c, 1, 1, f2))**

**{**

**if ((c >= 97) && (c <= 122))**

**{**

**if ((c - key[cur] + 97) >= 97) c = c - key[cur] + 97;**

**else**

**c = c - key[cur] + 123;**

**fprintf(f3, "%c", c);**

**}**

**else if (c == 32)**

**{**

**fprintf(f3, "%c", c);**

**}**

**}**

**}**

**fclose(f2); //закрываем шифратор**

**fclose(f3); //дешифратор**

**system("PAUSE");**

**return 0;**

***/\*-%c - Символ***

***%s - Строка символов***

***%d, i - Целое десятичное со знаком***

***%o - Целое восьмеричное***

***%u - Целое десятичное без знака***

***%x, X - Целое шестнадцатеричное***

***%f - Дробное число в фиксированном формате***

***%e, E - Дробное число в научном формате***

***%g, G - Дробное число в научном или фиксированном формате***

***%p - Указатель(в шестнадцатеричном виде)\*/***

**}**

**3. Реализовать шифрование и дешифрацию файла с использованием метода биграмм. Ключевое слово вводится.**

**Python**

**text = input('Введите текст : ')**

**text = [x for x in text]**

**res = ''**

**m = [**

**["Н","И","К","О","Л","А"],**

**["Б","В","Г","Д","Е","Ё"],**

**["Ж","З","Й","М","П","Р"],**

**["С","Т","У","Ф","Х","Ц"],**

**["Ч","Ш","Щ","Ь","Ы","Ъ"],**

**["Э","Ю","Я",".","-","\_"],**

**]**

**for i in range(1,len(text)):**

**if text[i] == text[i-1]:**

**text.insert(i,'Я')**

**if len(text)%2 != 0:**

**text.append('Я')**

**print(text)**

**for k in range(0,len(text),2):*# Берём по два аргумента***

**x1=-1;x2=-1;y1=-1;y2=-1**

**for i in range(6):**

**for j in range(6):**

**if x1 == -1:**

**if text[k] == m[i][j]:**

**x1=i; y1=j**

**if x2 == -1:**

**if text[k+1] == m[i][j]:**

**x2=i; y2=j**

**if x1== x2:**

**if y1==5:**

**y1=0**

**y2+=1**

**elif y2 == 5:**

**y1+=1**

**y2=0**

**else:**

**y1+=1**

**y2+=1**

**elif y1 == y2:**

**if x1 ==5 and x2 ==5:**

**x1 = 0; x2= 0**

**elif x1 == 5:**

**x1=0; x2+=1**

**elif x2 == 5:**

**x2=0; x1+=1**

**else:**

**x2+=1; x1+=1**

**else:**

**temp=0**

**temp=y1**

**y1=y2**

**y2=temp**

**text[k]=m[x1][y1]**

**text[k+1]=m[x2][y2]**

**print(text)**

**4. Реализовать в программе шифрование и дешифрацию файла с использованием квадрата Полибия, обеспечив его случайное заполнение.**

**С++**

**#include <iostream>**

**#include <vector>**

**#include <string>**

**#include <Windows.h>**

**#include <fstream>**

**#include <cassert>**

**#include <locale>**

**using namespace std;**

**//------------------------------------------------------------------**

**void fllng(const char beg, const char end, vector<string> &alph) {**

**for (char sym = beg; sym <= end; ++sym) {**

**if (alph.back().size() == 6)**

**alph.emplace\_back();**

**alph.back().append(1, sym);**

**}**

**}**

**//-----------------------------------------------------------------**

**bool serch\_sym(fstream &fail, vector<string> &alph, int &sym, int fl) {**

**for (auto iter = alph.begin(); iter < alph.end(); ++iter) {**

**int ind = iter->find(sym, 0);**

**if (ind != string::npos) {**

**fail.seekp(-1, fstream::cur);**

**if (fl)**

**if (++iter == alph.end() || iter->size() < ind)**

**sym = alph.begin()->at(ind);**

**else**

**sym = iter->at(ind);**

**else**

**if (iter == alph.begin() && (--alph.end())->size() < ind)**

**sym = (alph.end() - 2)->at(ind);**

**else if (iter == alph.begin())**

**sym = (--alph.end())->at(ind);**

**else**

**sym = (--iter)->at(ind);**

**if (fail << static\_cast<char>(sym) << flush)**

**return true;**

**return false;**

**}**

**}**

**return false;**

**}**

**//-----------------------------------------------------------------**

**int main() {**

**SetConsoleCP(1251);**

**SetConsoleOutputCP(1251);**

**setlocale(LC\_CTYPE, "rus");**

**vector<string> llatin(1);**

**vector<string> lkiril(1);**

**vector<string> ulatin(1);**

**vector<string> ukiril(1);**

**fllng('A', 'Z', ulatin);**

**fllng('А', 'Я', ukiril);**

**fllng('a', 'z', llatin);**

**fllng('а', 'я', lkiril);**

**fstream fail("Polib.txt", fstream::ate | fstream::in | fstream::out);**

**if (!fail.is\_open())**

**fail.open("Polib.txt", fstream::app | fstream::in | fstream::out);**

**string sent;**

**cout << "Введите строки для кодировки, для конца ввода Ctrl + Z\n";**

**while (getline(cin, sent))**

**fail << sent + '\n' << flush;**

**cin.clear();**

**fail.seekg(0, fstream::beg);**

**int sym, fl = -1;**

**while (fl < 0 || fl > 1) {**

**cout << "Кодировать 1 разкодировать 0\n";**

**cin >> fl;**

**if (!cin.good()) {**

**while (cin.get() != '\n');**

**cin.clear();**

**}**

**}**

**while ((sym = fail.get()) != EOF) {**

**if (sym >= 'A' && sym <= 'Z')**

**assert(serch\_sym(fail, ulatin, sym, fl));**

**else if (sym >= 192 && sym <= 223)**

**assert(serch\_sym(fail, ukiril, sym, fl));**

**else if (sym >= 'a' && sym <= 'z')**

**assert(serch\_sym(fail, llatin, sym, fl));**

**else if (sym >= 224 && sym <= 255)**

**assert(serch\_sym(fail, lkiril, sym, fl));**

**}**

**fail.close();**

**return 0;**

**}**

**5. Реализовать в программе шифрование и дешифрацию файла с использованием квадрата Кардано размером 4х4.**

**(С++)**

**#include <iostream.h>**

**int main(int argc, char\* argv[])**

**{**

**const int SIZE = 5;**

**char \*buf[SIZE] = {"guhoe",**

**"llxoe",**

**"bnce.",**

**"utp!w",**

**"nqwmp"};**

**int grid[SIZE][SIZE] ={{0, 0, 1, 0, 1},**

**{1, 1, 0, 1, 0},**

**{0, 0, 0, 0, 0},**

**{0, 0, 0, 1, 0},**

**{0, 0, 0, 0, 0}};**

**for (int i = 0; i < SIZE; i++)**

**cout << buf[i] << endl;**

**cout << endl;**

**cout << "0:" << endl;**

**for (int i = 0; i < SIZE; i++)**

**for (int j = 0; j < SIZE; j++)**

**if (grid[i][j] == 1)**

**cout << buf[i][j];**

**cout << endl;**

**cout << "90:" << endl;**

**for (int i = 0; i < SIZE; i++)**

**for (int j = 0; j < SIZE; j++)**

**if (grid[SIZE-j-1][i] == 1)**

**cout << buf[i][j];**

**cout << endl;**

**cout << "180:" << endl;**

**for (int i = 0; i < SIZE; i++)**

**for (int j = 0; j < SIZE; j++)**

**if (grid[SIZE-i-1][SIZE-j-1] == 1)**

**cout << buf[i][j];**

**cout << endl;**

**cout << "270:" << endl;**

**for (int i = 0; i < SIZE; i++)**

**for (int j = 0; j < SIZE; j++)**

**if (grid[j][SIZE-i-1] == 1)**

**cout << buf[i][j];**

**cout << endl;**

**system("pause");**

**return 0;**

**}**

**6. Реализовать в программе шифрование и дешифрацию файла методом биграмм с двойным квадратом. Квадраты генерировать динамически для каждого шифрования.**

**С++**

**// Шифрование биграммами с двойным квадратом.cpp: определяет точку входа для консольного приложения.**

**//**

**#include "stdafx.h"**

**#include <iostream>**

**#include <string.h>**

**using namespace std;**

**int main()**

**{//инициализация 1-го массива и вывод его на экран**

**const int A=6,B=6;**

**char Mass[A][B]={"ABCDE","FGHIJ","KLMNO","PQRST","UVWXY","Z,-.!"};**

**int i,j;**

**cout<<"--- 1 Massiv ---"<<endl;**

**for (i=0;i<A;i++) {cout<<endl;**

**for (j=0;j<B;j++) {cout<<Mass[i][j]<<" ";}**

**}**

**//инициализация 2-го массива и вывод его на экран**

**const int A1=6,B1=6;**

**char Mass1[A1][B1]={"ABCDE","FGHIJ","KLMNO","PQRST","UVWXY","Z,-.!"};**

**int i1,j1;**

**cout<<endl<<endl<<"--- 2 Massiv ---"<<endl;**

**for (i1=0;i1<A1;i1++) {cout<<endl;**

**for (j1=0;j1<B1;j1++) {cout<<Mass1[i1][j1]<<" ";}**

**}**

**//вводим ключ для первого массива и вывод исходного массива**

**cout<<endl<<"Input key for 1-st Massiv (5 symbols) ---> ";**

**char key1[A];//КЛЮЧ**

**cin>>key1;**

**char g1=Mass[0][0];//ДЛЯ 1-Й БУКВЫ**

**char g2=Mass[0][1];//ДЛЯ 2-Й БУКВЫ**

**char g3=Mass[0][2];//ДЛЯ 3-Й БУКВЫ**

**char g4=Mass[0][3];//ДЛЯ 4-Й БУКВЫ**

**char g5=Mass[0][4];//ДЛЯ 5-Й БУКВЫ**

**for (i=0;i<A;i++) {cout<<endl;**

**for (j=0;j<B;j++) {**

**if (Mass[i][j]==key1[0]) {key1[0]=Mass[0][0]; Mass[0][0]=Mass[i][j];Mass[i][j]=g1;}**

**if (Mass[i][j]==key1[1]) {key1[1]=Mass[0][1]; Mass[0][1]=Mass[i][j];Mass[i][j]=g2;}//**

**if (Mass[i][j]==key1[2]) {key1[2]=Mass[0][2]; Mass[0][2]=Mass[i][j];Mass[i][j]=g3;}**

**if (Mass[i][j]==key1[3]) {key1[3]=Mass[0][3]; Mass[0][3]=Mass[i][j];Mass[i][j]=g4;}//**

**if (Mass[i][j]==key1[4]) {key1[4]=Mass[0][4]; Mass[0][4]=Mass[i][j];Mass[i][j]=g5;}**

**}**

**}**

**cout<<"--- 1 Massiv (NEW) ---"<<endl;**

**for (i=0;i<A;i++) {cout<<endl;**

**for (j=0;j<B;j++) {cout<<Mass[i][j]<<" ";}**

**}**

**//вводим ключ для второго массива и вывод исходного массива**

**cout<<endl<<"Input key for 2-d Massiv (5 symbols) ---> ";**

**char key2[A1];//КЛЮЧ**

**cin>>key2;**

**char G1=Mass1[0][0];//ДЛЯ 1-Й БУКВЫ**

**char G2=Mass1[0][1];//ДЛЯ 2-Й БУКВЫ**

**char G3=Mass1[0][2];//ДЛЯ 3-Й БУКВЫ**

**char G4=Mass1[0][3];//ДЛЯ 4-Й БУКВЫ**

**char G5=Mass1[0][4];//ДЛЯ 5-Й БУКВЫ**

**for (i1=0;i1<A1;i1++) {cout<<endl;**

**for (j1=0;j1<B1;j1++) {**

**if (Mass1[i1][j1]==key2[0]) {key2[0]=Mass1[0][0]; Mass1[0][0]=Mass1[i1][j1];Mass1[i1][j1]=G1;}**

**if (Mass1[i1][j1]==key2[1]) {key2[1]=Mass1[0][1]; Mass1[0][1]=Mass1[i1][j1];Mass1[i1][j1]=G2;}//**

**if (Mass1[i1][j1]==key2[2]) {key2[2]=Mass1[0][2]; Mass1[0][2]=Mass1[i1][j1];Mass1[i1][j1]=G3;}**

**if (Mass1[i1][j1]==key2[3]) {key2[3]=Mass1[0][3]; Mass1[0][3]=Mass1[i1][j1];Mass1[i1][j1]=G4;}//**

**if (Mass1[i1][j1]==key2[4]) {key2[4]=Mass1[0][4]; Mass1[0][4]=Mass1[i1][j1];Mass1[i1][j1]=G5;}**

**}**

**}**

**cout<<"--- 2 Massiv (NEW) ---"<<endl;**

**for (i1=0;i1<A1;i1++) {cout<<endl;**

**for (j1=0;j1<B1;j1++) {cout<<Mass1[i1][j1]<<" ";}**

**}**

**//ВВОДИМ И ШИФРУЕМ СЛОВО**

**cout<<endl<<"Input word(limit 10)"<<endl;**

**char word[6];**

**cin>>word[6];**

**//-------поиск в 1-ом и во 2-ом ------------------------**

**int x,y,X,Y;//1-2**

**int x1,y1,X1,Y1;//3-4**

**int x2,y2,X2,Y2;//5-6**

**int x3,y3,X3,Y3;//7-8**

**int x4,y4,X4,Y4;//9-10**

**char w0=word[0];**

**char w1=word[1];**

**for (i=0;i<A;i++) {//1-й (1-я буква) [1-й массив 1-я буква]**

**for (j=0;j<B;j++)**

**{ if (Mass[i][j]==w0) {x=i;y=j;}**

**}**

**}**

**for (i1=0;i1<A1;i1++) {//2-й (2-я буква)**

**for (j1=0;j1<B1;j1++)**

**{ if (Mass1[i1][j1]==w1) {X=i1;Y=j1;}**

**}**

**}**

**cout<<"Number 1 ---- ["<<x<<" "<<y<<"]"<<endl;**

**cout<<"Number 2 ---- ["<<X<<" "<<Y<<"]"<<endl;**

***/\*for (i=0;i<A;i++) {//1-й (3-я буква)***

***for (j=0;j<B;j++)***

***{ if (Mass[i][j]==word[3]) {x1=i;y1=j;}***

***}***

***}***

***for (i1=0;i1<A1;i1++) {//2-й (4-я буква)***

***for (j1=0;j1<B1;j1++)***

***{ if (Mass1[i1][j1]==word[4]) {X1=i1;Y1=j1;}***

***}***

***}***

***for (i=0;i<A;i++) {//1-й (5-я буква)***

***for (j=0;j<B;j++)***

***{ if (Mass[i][j]==word[5]) {x2=i;y2=j;}***

***}***

***}***

***for (i1=0;i1<A1;i1++) {//2-й (6-я буква)***

***for (j1=0;j1<B1;j1++)***

***{ if (Mass1[i1][j1]==word[6]) {X2=i1;Y2=j1;}***

***}***

***}***

***for (i=0;i<A;i++) {//1-й (7-я буква)***

***for (j=0;j<B;j++)***

***{ if (Mass[i][j]==word[7]) {x3=i;y3=j;}***

***}***

***}***

***for (i1=0;i1<A1;i1++) {//2-й (8-я буква)***

***for (j1=0;j1<B1;j1++)***

***{ if (Mass1[i1][j1]==word[8]) {X3=i1;Y3=j1;}***

***}***

***}***

***for (i=0;i<A;i++) {//1-й (9-я буква)***

***for (j=0;j<B;j++)***

***{ if (Mass[i][j]==word[9]) {x4=i;y4=j;}***

***}***

***}***

***for (i1=0;i1<A1;i1++) {//2-й (10-я буква)***

***for (j1=0;j1<B1;j1++)***

***{ if (Mass1[i1][j1]==word[10]) {X4=i1;Y4=j1;}***

***}***

***}\*/***

**cout<<endl<<"Vash shifrovanoe slovo - "<<endl;**

**cout<<Mass1[x][Y]<<" | "<<Mass[X][y]<<endl;**

**system("pause");**

**return 0;**

**}**

**7. Реализовать в программе шифрование и дешифрацию файла с использованием перестановочного шифра с ключевым словом. Ключевое слово вводится.**

**С++**

**#include "stdafx.h"**

**#include <string>**

**#include <iostream>**

**void permutationcode(std::string& text, std::string& const key)**

**{**

**int col = key.size();**

**int row = (text.size() % col) ? ((text.size() / col) + 1) : (text.size() / col);**

**int\*\* arrtext = new int\*[row];**

**for (int i = 0; i < row; ++i)**

**arrtext[i] = new int[col];**

**//inizialization array {0}**

**for (int i = 0; i < row; ++i)**

**for (int j = 0; j < col; ++j)**

**arrtext[i][j] = 0;**

**//fill array {text[i]}**

**for (int i = 0; i < row; ++i)**

**for (int j = 0; j < col; ++j)**

**arrtext[i][j] = text[(i\*col)+j];**

**int index = 0;**

**//permutation**

**for (int i = 0; i < 256; ++i)**

**for (int j = 0; j < key.size(); ++j)**

**if (key[j] == i)**

**for (int k = 0; k < row; ++k){**

**text[index] = arrtext[k][j];**

**++index;**

**}**

**for (int i = 0; i < row; ++i)**

**delete[] arrtext[i];**

**delete[] arrtext;**

**}**

***/\****

***Помогите, нужно сделать программу что будет зашифровывать и расшифровывать текст с помощью перестановочного шифра.***

***Буквы открытого текста записываются в клетки прямоугольной таблицы по ее строчкам.***

***Буквы ключевого слова пишутся над столбцами и указывают порядок этих столбцов (по возрастанию номеров букв в алфавите).***

***Чтобы получить зашифрованный текст, надо выписывать буквы по столбцам с учетом их нумерации:***

***\*/***

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

**{**

**std::string text("Hello programmers!");**

**std::string key("bac");**

**std::cout << "Text before: " << text << std::endl << std::endl;**

**permutationcode(text,key);**

**std::cout << "Text after: " << text << std::endl;**

**std::cin.get();**

**return 0;**

**}**

**8. Реализовать в программе шифрование и дешифрацию файла с использованием аффинной криптосистемы. Провести частотный анализ зашифрованного файла, осуществляя проверку по файлу с набором ключевых слов.**

**(С++)**

**#include <cstdlib>**

**#include <iostream>**

**#include <string>**

**#include <locale>**

**#include <fstream>**

**#include <conio.h>**

**#include <vector>**

**#include <iomanip>**

**#include <algorithm>**

**const std::string alphabet\_eng = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ﻿абвгдеёжзийклмнопрстуфхцчшщъыьэюяАБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ "!#$%&'()\*+,-./:;<=>?@0123456789[\]^\_`{|}~";**

**const size\_t n\_eng = alphabet\_eng.size();**

**std::vector<std::string> read\_file(std::string file\_name)**

**{**

**std::vector<std::string> text;**

**std::string temp;**

**std::ifstream ist(file\_name, std::ios::in | std::ios::binary);**

**if (!ist) {**

**std::cout << "cant open the file with this name! return to menu..." << std::endl;**

**}**

**else {**

**while (!ist.eof()) {**

**std::getline(ist, temp);**

**text.push\_back(temp);**

**}**

**std::cout << "file has been successfully read" << std::endl;**

**}**

**ist.close(); //Закрываем файл**

**return text;**

**}**

**void write\_file(std::string file\_name, std::vector<std::string> text)**

**{**

**std::string temp;**

**std::ofstream ost(file\_name, std::ios::in | std::ios::binary);**

**if (!ost) {**

**std::cout << "cant open the file with this name" << std::endl;**

**}**

**else {**

**for (size\_t i = 0; i < text.size(); i++) {**

**ost << text[i] << std::endl;**

**}**

**std::cout << "file has been successfully writed" << std::endl;**

**}**

**ost.close(); //Закрываем файл**

**}**

**void print\_text(std::vector<std::string> text)**

**{**

**for (size\_t i = 0; i < text.size(); i++) {**

**std::cout << text[i] << std::endl;**

**}**

**}**

**std::vector<std::string> encrypt(std::vector<std::string> text, size\_t k)**

**{**

**k %= n\_eng;**

**std::string txt;**

**for (size\_t i = 0; i < text.size(); ++i) {**

**txt = text[i];**

**for (size\_t j = 0; j < txt.length(); ++j) {**

**if (isprint((unsigned char)txt[j])) {**

**txt[j] = alphabet\_eng[(alphabet\_eng.find(txt[j]) + k) % n\_eng];**

**}**

**}**

**text[i] = txt;**

**}**

**return text;**

**}**

**std::vector<std::string> decrypt(std::vector<std::string> text, size\_t k)**

**{**

**k %= n\_eng;**

**std::string txt;**

**for (size\_t i = 0; i < text.size(); ++i) {**

**txt = text[i];**

**for (size\_t j = 0; j < txt.length(); ++j) {**

**if (isprint((unsigned char)txt[j])) {**

**txt[j] = alphabet\_eng[(alphabet\_eng.find(txt[j]) - k + n\_eng) % n\_eng];**

**}**

**}**

**text[i] = txt;**

**}**

**return text;**

**}**

**std::vector<double> frequency\_analysis(std::vector<std::string> text)**

**{**

**std::vector<double> counters(59);**

**int count = 0;**

**std::string txt;**

**for (size\_t i = 0; i < text.size(); i++) {**

**txt = text[i];**

**for (size\_t j = 0; j < txt.length(); ++j) {**

**for (size\_t l = 0; l < 26; ++l) {**

**if (tolower((unsigned char)txt[j]) == alphabet\_eng[l]) {**

**counters[l]++;**

**}**

**}**

**for (size\_t m = 53; m < 86; ++m) {**

**if (txt[j] == alphabet\_eng[m] || txt[j] == alphabet\_eng[m+33]) {**

**counters[m - 27]++;**

**}**

**}**

**count++;**

**}**

**}**

**std::cout << "frequency analyse complete" << std::endl;**

**std::ofstream ost("graph.txt");**

**if (!ost) {**

**std::cout << "cant open the file with this name" << std::endl;**

**}**

**else {**

**for (size\_t i = 0; i < counters.size(); i++) {**

**counters[i] = counters[i] / count \* 100;**

**ost << std::setprecision(0) << std::fixed << int(counters[i]) << ',' << (counters[i] - int(counters[i])) \* 100 << std::endl;**

**}**

**std::cout << "file has been successfully writed" << std::endl;**

**}**

**ost.close();**

**return counters;**

**}**

**void showmenu()**

**{**

**std::cout << "Menu : " << std::endl;**

**std::cout << "1 - Open file\n";**

**std::cout << "2 - Write text from file\n";**

**std::cout << "3 - Crypt file\n";**

**std::cout << "4 - Decrypt file\n";**

**std::cout << "5 - Frequency analysis\n";**

**std::cout << "0 - Exit\n";**

**std::cout << std::endl;**

**}**

**int main()**

**{**

**setlocale(LC\_ALL, "");**

**size\_t k = 0;**

**std::vector<std::string> text; //текст считанный из файла**

**bool prisn;**

**prisn = true;**

**do**

**{**

**system("cls");**

**showmenu();**

**switch (\_getch())**

**{**

**case '1': //open file to string**

**{**

**std::cout << "write a file name" << std::endl;**

**std::string file\_name;**

**std::cin >> file\_name;**

**text = read\_file(file\_name);**

**break;**

**};**

**case '2': //crypted string**

**{**

**if (text.size() == 0) { std::cout << "text is empty! return to menu..." << std::endl; }**

**else {**

**print\_text(text);**

**}**

**break;**

**};**

**case '3': //crypted string**

**{**

**if (text.size() == 0) { std::cout << "text is empty! return to menu..." << std::endl; }**

**else {**

**std::cout << "write key:" << std::endl;**

**std::cin >> k;**

**std::string file\_name = "crypt.txt";**

**write\_file(file\_name, encrypt(text, k));**

**std::cout << "text crypted" << std::endl;**

**}**

**break;**

**};**

**case '4': //decrypted string**

**{**

**if (text.size() == 0) { std::cout << "text is empty! return to menu..." << std::endl; }**

**else {**

**std::cout << "write key:" << std::endl;**

**std::cin >> k;**

**std::string file\_name = "decrypt.txt";**

**write\_file(file\_name, decrypt(text, k));**

**std::cout << "text decrypted" << std::endl;**

**}**

**break;**

**};**

**case '5': //frequency analysis**

**{**

**if (text.size() == 0) { std::cout << "text is empty! return to menu..." << std::endl; }**

**else {**

**frequency\_analysis(text);**

**}**

**break;**

**};**

**case '0': //exit**

**{**

**prisn = false;**

**exit(0);**

**}**

**default:**

**{**

**std::cout << "Wrong input!" << std::endl;**

**break;**

**}**

**}**

**std::cout << std::endl << std::endl;**

**system("pause");**

**} while (prisn);**

**return 0;**

**}**

**9. Реализовать шифрование и дешифрацию файла по методу Виженера с составным ключом. Набор ключей вводится.**

**С#**

**char[] characters = new char[] { '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', 'Ь', 'Ы', 'Ъ',**

**'Э', 'Ю', 'Я', ' ', '1', '2', '3', '4', '5', '6', '7',**

**'8', '9', '0' };**

**N = characters.Length;**

**//зашифровать**

**private string Encode(string input, string keyword)**

**{**

**input = input.ToUpper();**

**keyword = keyword.ToUpper();**

**string result = "";**

**int keyword\_index = 0;**

**foreach (char symbol in input)**

**{**

**int c = (Array.IndexOf(characters, symbol) +**

**Array.IndexOf(characters, keyword[keyword\_index])) % N;**

**result += characters[c];**

**keyword\_index++;**

**if ((keyword\_index + 1) == keyword.Length)**

**keyword\_index = 0;**

**}**

**return result;**

**}**

**10. Реализовать в программе шифрование и дешифрацию содержимого файла по методу Цезаря. Провести частотный анализ зашифрованного файла, осуществляя проверку по файлу с набором ключевых слов.**

**С++**

**#include <cstdlib>**

**#include <iostream>**

**#include <string>**

**#include <locale>**

**#include <fstream>**

**#include <conio.h>**

**#include <vector>**

**#include <iomanip>**

**#include <algorithm>**

**const std::string alphabet\_eng = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ﻿абвгдеёжзийклмнопрстуфхцчшщъыьэюяАБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ "!#$%&'()\*+,-./:;<=>?@0123456789[\]^\_`{|}~";**

**const size\_t n\_eng = alphabet\_eng.size();**

**std::vector<std::string> read\_file(std::string file\_name)**

**{**

**std::vector<std::string> text;**

**std::string temp;**

**std::ifstream ist(file\_name, std::ios::in | std::ios::binary);**

**if (!ist) {**

**std::cout << "cant open the file with this name! return to menu..." << std::endl;**

**}**

**else {**

**while (!ist.eof()) {**

**std::getline(ist, temp);**

**text.push\_back(temp);**

**}**

**std::cout << "file has been successfully read" << std::endl;**

**}**

**ist.close(); //Закрываем файл**

**return text;**

**}**

**void write\_file(std::string file\_name, std::vector<std::string> text)**

**{**

**std::string temp;**

**std::ofstream ost(file\_name, std::ios::in | std::ios::binary);**

**if (!ost) {**

**std::cout << "cant open the file with this name" << std::endl;**

**}**

**else {**

**for (size\_t i = 0; i < text.size(); i++) {**

**ost << text[i] << std::endl;**

**}**

**std::cout << "file has been successfully writed" << std::endl;**

**}**

**ost.close(); //Закрываем файл**

**}**

**void print\_text(std::vector<std::string> text)**

**{**

**for (size\_t i = 0; i < text.size(); i++) {**

**std::cout << text[i] << std::endl;**

**}**

**}**

**std::vector<std::string> encrypt(std::vector<std::string> text, size\_t k)**

**{**

**k %= n\_eng;**

**std::string txt;**

**for (size\_t i = 0; i < text.size(); ++i) {**

**txt = text[i];**

**for (size\_t j = 0; j < txt.length(); ++j) {**

**if (isprint((unsigned char)txt[j])) {**

**txt[j] = alphabet\_eng[(alphabet\_eng.find(txt[j]) + k) % n\_eng];**

**}**

**}**

**text[i] = txt;**

**}**

**return text;**

**}**

**std::vector<std::string> decrypt(std::vector<std::string> text, size\_t k)**

**{**

**k %= n\_eng;**

**std::string txt;**

**for (size\_t i = 0; i < text.size(); ++i) {**

**txt = text[i];**

**for (size\_t j = 0; j < txt.length(); ++j) {**

**if (isprint((unsigned char)txt[j])) {**

**txt[j] = alphabet\_eng[(alphabet\_eng.find(txt[j]) - k + n\_eng) % n\_eng];**

**}**

**}**

**text[i] = txt;**

**}**

**return text;**

**}**

**std::vector<double> frequency\_analysis(std::vector<std::string> text)**

**{**

**std::vector<double> counters(59);**

**int count = 0;**

**std::string txt;**

**for (size\_t i = 0; i < text.size(); i++) {**

**txt = text[i];**

**for (size\_t j = 0; j < txt.length(); ++j) {**

**for (size\_t l = 0; l < 26; ++l) {**

**if (tolower((unsigned char)txt[j]) == alphabet\_eng[l]) {**

**counters[l]++;**

**}**

**}**

**for (size\_t m = 53; m < 86; ++m) {**

**if (txt[j] == alphabet\_eng[m] || txt[j] == alphabet\_eng[m+33]) {**

**counters[m - 27]++;**

**}**

**}**

**count++;**

**}**

**}**

**std::cout << "frequency analyse complete" << std::endl;**

**std::ofstream ost("graph.txt");**

**if (!ost) {**

**std::cout << "cant open the file with this name" << std::endl;**

**}**

**else {**

**for (size\_t i = 0; i < counters.size(); i++) {**

**counters[i] = counters[i] / count \* 100;**

**ost << std::setprecision(0) << std::fixed << int(counters[i]) << ',' << (counters[i] - int(counters[i])) \* 100 << std::endl;**

**}**

**std::cout << "file has been successfully writed" << std::endl;**

**}**

**ost.close();**

**return counters;**

**}**

**void showmenu()**

**{**

**std::cout << "Menu : " << std::endl;**

**std::cout << "1 - Open file\n";**

**std::cout << "2 - Write text from file\n";**

**std::cout << "3 - Crypt file\n";**

**std::cout << "4 - Decrypt file\n";**

**std::cout << "5 - Frequency analysis\n";**

**std::cout << "0 - Exit\n";**

**std::cout << std::endl;**

**}**

**int main()**

**{**

**setlocale(LC\_ALL, "");**

**size\_t k = 0;**

**std::vector<std::string> text; //текст считанный из файла**

**bool prisn;**

**prisn = true;**

**do**

**{**

**system("cls");**

**showmenu();**

**switch (\_getch())**

**{**

**case '1': //open file to string**

**{**

**std::cout << "write a file name" << std::endl;**

**std::string file\_name;**

**std::cin >> file\_name;**

**text = read\_file(file\_name);**

**break;**

**};**

**case '2': //crypted string**

**{**

**if (text.size() == 0) { std::cout << "text is empty! return to menu..." << std::endl; }**

**else {**

**print\_text(text);**

**}**

**break;**

**};**

**case '3': //crypted string**

**{**

**if (text.size() == 0) { std::cout << "text is empty! return to menu..." << std::endl; }**

**else {**

**std::cout << "write key:" << std::endl;**

**std::cin >> k;**

**std::string file\_name = "crypt.txt";**

**write\_file(file\_name, encrypt(text, k));**

**std::cout << "text crypted" << std::endl;**

**}**

**break;**

**};**

**case '4': //decrypted string**

**{**

**if (text.size() == 0) { std::cout << "text is empty! return to menu..." << std::endl; }**

**else {**

**std::cout << "write key:" << std::endl;**

**std::cin >> k;**

**std::string file\_name = "decrypt.txt";**

**write\_file(file\_name, decrypt(text, k));**

**std::cout << "text decrypted" << std::endl;**

**}**

**break;**

**};**

**case '5': //frequency analysis**

**{**

**if (text.size() == 0) { std::cout << "text is empty! return to menu..." << std::endl; }**

**else {**

**frequency\_analysis(text);**

**}**

**break;**

**};**

**case '0': //exit**

**{**

**prisn = false;**

**exit(0);**

**}**

**default:**

**{**

**std::cout << "Wrong input!" << std::endl;**

**break;**

**}**

**}**

**std::cout << std::endl << std::endl;**

**system("pause");**

**} while (prisn);**

**return 0;**

**}**

**Контрольные вопросы**

**1. Дайте определение следующим понятиям: шифр, криптография, криптоанализ, ключ.**

Шифр — система обратимых преобразований, зависящая от некоторого секретного параметра ([ключа](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D1%8E%D1%87_(%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%8F))) и предназначенная для обеспечения секретности передаваемой информации.

Криптогра́фия — наука о методах обеспечения [конфиденциальности](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BD%D1%84%D0%B8%D0%B4%D0%B5%D0%BD%D1%86%D0%B8%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D1%81%D1%82%D1%8C) (невозможности прочтения информации посторонним), [целостности данных](https://ru.wikipedia.org/wiki/%D0%A6%D0%B5%D0%BB%D0%BE%D1%81%D1%82%D0%BD%D0%BE%D1%81%D1%82%D1%8C_%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D0%B8%D0%B8) (невозможности незаметного изменения информации), [аутентификации](https://ru.wikipedia.org/wiki/%D0%90%D1%83%D1%82%D0%B5%D0%BD%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F) (проверки подлинности авторства или иных свойств объекта), [шифрования](https://ru.wikipedia.org/wiki/%D0%A8%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) (кодировка данных).

Изначально криптография изучала методы [шифрования](https://ru.wikipedia.org/wiki/%D0%A8%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) информации — обратимого преобразования открытого (исходного) текста на основе секретного [алгоритма](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC) или [ключа](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D1%8E%D1%87_(%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%8F)) в шифрованный текст (шифротекст). Традиционная криптография образует раздел [симметричных криптосистем](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BC%D0%BC%D0%B5%D1%82%D1%80%D0%B8%D1%87%D0%BD%D1%8B%D0%B5_%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D1%8B), в которых зашифровывание и расшифровывание проводится с использованием одного и того же [секретного ключа](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D1%8E%D1%87_(%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%8F)).

Пример:Шифр [АТБАШ](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B1%D0%B0%D1%88), в котором ключом является перевёрнутый [алфавит](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D1%84%D0%B0%D0%B2%D0%B8%D1%82) того языка, на котором шифруется текст.

Помимо этого раздела современная криптография включает в себя [асимметричные криптосистемы](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%81_%D0%BE%D1%82%D0%BA%D1%80%D1%8B%D1%82%D1%8B%D0%BC_%D0%BA%D0%BB%D1%8E%D1%87%D0%BE%D0%BC), системы [электронной цифровой подписи](https://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D0%BF%D0%BE%D0%B4%D0%BF%D0%B8%D1%81%D1%8C) (ЭЦП), [хеш-функции](https://ru.wikipedia.org/wiki/%D0%A5%D0%B5%D1%88%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5), [управление ключами](https://ru.wikipedia.org/wiki/%D0%A3%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5_%D0%BA%D0%BB%D1%8E%D1%87%D0%B0%D0%BC%D0%B8), [получение скрытой информации](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D1%83%D1%87%D0%B5%D0%BD%D0%B8%D0%B5_%D1%81%D0%BA%D1%80%D1%8B%D1%82%D0%BE%D0%B9_%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D0%B8%D0%B8), [квантовую криптографию](https://ru.wikipedia.org/wiki/%D0%9A%D0%B2%D0%B0%D0%BD%D1%82%D0%BE%D0%B2%D0%B0%D1%8F_%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%8F).

Криптография не является защитой от [обмана](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D0%BC%D0%B0%D0%BD), [подкупа](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%B4%D0%BA%D1%83%D0%BF) или [шантажа](https://ru.wikipedia.org/wiki/%D0%A8%D0%B0%D0%BD%D1%82%D0%B0%D0%B6) законных абонентов, кражи ключей и других угроз информации, возникающих в защищённых системах [передачи данных](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D0%B5%D0%B4%D0%B0%D1%87%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85).

Криптография — одна из старейших наук, [её история](https://ru.wikipedia.org/wiki/%D0%98%D1%81%D1%82%D0%BE%D1%80%D0%B8%D1%8F_%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D0%B8) насчитывает несколько тысяч лет.

Криптоана́лиз — наука о методах [дешифровки](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%8F#%D0%A2%D0%B5%D1%80%D0%BC%D0%B8%D0%BD%D0%BE%D0%BB%D0%BE%D0%B3%D0%B8%D1%8F) зашифрованной информации без предназначенного для этого [ключа](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D1%8E%D1%87_(%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%8F)), а также сам процесс такой дешифровки.

В большинстве случаев под криптоанализом понимается выяснение [ключа](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D1%8E%D1%87_(%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%8F)); криптоанализ включает также методы выявления уязвимости криптографических алгоритмов или протоколов.

Попытку раскрытия конкретного [шифра](https://ru.wikipedia.org/wiki/%D0%A8%D0%B8%D1%84%D1%80) с применением методов криптоанализа называют криптографической атакой на этот шифр. Криптографическую атаку, в ходе которой раскрыть шифр удалось, называют «взломом» или «вскрытием» шифра.

Термин был введён американским криптографом [Уильямом Ф. Фридманом](https://ru.wikipedia.org/wiki/%D0%A4%D1%80%D0%B8%D0%B4%D0%BC%D0%B0%D0%BD,_%D0%A3%D0%B8%D0%BB%D1%8C%D1%8F%D0%BC_%D0%A4%D1%80%D0%B5%D0%B4%D0%B5%D1%80%D0%B8%D0%BA) в [1920 году](https://ru.wikipedia.org/wiki/1920_%D0%B3%D0%BE%D0%B4) в рамках его книги «Элементы криптоанализа»[[1]](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B0%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7#cite_note-KhanD-1).

Первоначально методы криптоанализа основывались на лингвистических закономерностях естественного текста и реализовывались с использованием только карандаша и бумаги. Со временем в криптоанализе нарастает роль чисто математических методов, для реализации которых используются специализированные [криптоаналитические компьютеры](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B0%D0%BD%D0%B0%D0%BB%D0%B8%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%BA%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80).

**2. Классифицируйте алгоритм, полученный в качестве задания к лабораторной работе.**

**3. Чем отличаются одно- и многоалфавитные методы шифрования?**

Полиалфавитный шифр (многоалфавитный шифр) — это совокупность [шифров простой замены](https://ru.wikipedia.org/wiki/%D0%A8%D0%B8%D1%84%D1%80_%D0%BF%D1%80%D0%BE%D1%81%D1%82%D0%BE%D0%B9_%D0%B7%D0%B0%D0%BC%D0%B5%D0%BD%D1%8B), которые используются для шифрования очередного символа [открытого текста](https://ru.wikipedia.org/wiki/%D0%9E%D1%82%D0%BA%D1%80%D1%8B%D1%82%D1%8B%D0%B9_%D1%82%D0%B5%D0%BA%D1%81%D1%82) согласно некоторому правилу.

Суть полиалфавитного шифра заключается в циклическом применении нескольких [моноалфавитных шифров](https://ru.wikipedia.org/wiki/%D0%A8%D0%B8%D1%84%D1%80_%D0%BF%D1%80%D0%BE%D1%81%D1%82%D0%BE%D0%B9_%D0%B7%D0%B0%D0%BC%D0%B5%D0%BD%D1%8B) к определённому числу букв шифруемого текста. Предположим, что имеется некоторое сообщение x1 , x2 , x3 , …, xn , …, x2n, …, которое необходимо зашифровать, а также для использования полиалфавитного шифра взяли n моноалфавитных шифров. В данном случае к первой букве применяется первый моноалфавитный шифр, ко второй букве — второй, к третьей — третий, …, к n-ой букве — n-ый, а к (n+1)-ой вновь первый, и так далее, пока все сообщение не будет зашифровано. Таким образом, получается довольно-таки сложная последовательность, вскрыть которую сложнее нежели моноалфавитный шифр. Важным эффектом, достигаемым при использовании полиалфавитного шифра, является маскировка частот появления тех или иных букв в тексте, чего лишены шифры простой замены.

Шифр простой замены, простой подстановочный шифр, моноалфавитный шифр — класс методов [шифрования](https://ru.wikipedia.org/wiki/%D0%A8%D0%B8%D1%84%D1%80), которые сводятся к созданию по определённому алгоритму таблицы шифрования, в которой для каждой буквы открытого текста существует единственная сопоставленная ей буква шифр-текста. Само шифрование заключается в замене букв согласно таблице. Для расшифровки достаточно иметь ту же таблицу, либо знать [алгоритм](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC), по которому она генерируется.

К шифрам простой замены относятся многие способы шифрования, возникшие в древности или средневековье, как, например, [Атбаш](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%B1%D0%B0%D1%88)ь (также читается как этбаш) или [шифр Цезаря](https://ru.wikipedia.org/wiki/%D0%A8%D0%B8%D1%84%D1%80_%D0%A6%D0%B5%D0%B7%D0%B0%D1%80%D1%8F). Для вскрытия подобных шифров используется [частотный криптоанализ](https://ru.wikipedia.org/wiki/%D0%A7%D0%B0%D1%81%D1%82%D0%BE%D1%82%D0%BD%D1%8B%D0%B9_%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B0%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7).

**4. В чем заключается основной принцип частотного криптоанализа?**

Частотный анализ, частотный криптоанализ — один из методов криптоанализа, основывающийся на предположении о существовании нетривиального статистического распределения отдельных символов и их последовательностей как в открытом тексте, так и в шифротексте, которое, с точностью до замены символов, будет сохраняться в процессе шифрования и дешифрования.

**5. Какой метод криптоанализа применим для вскрытия алгоритма, полученного вами в качестве задания к лабораторной работе?**

**6. Оцените мощность ключевого пространства вашего алгоритма.**