**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Отчет по лабораторной работе №7

«**Обработка символьных строк**»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Выполнил:** | |  | **Принял:** | |
| ФИО: | \_Цыпышев Т. А.\_\_\_\_\_ |  | ФИО: | \_**Самохвалов А. Э.**\_\_\_ |
| Группа: | \_ИУ5-11Б\_\_\_\_\_\_\_\_\_\_\_ |  | Должность: | \_Преподаватель\_\_\_\_ |
| Дата: | \_16.11.2022\_\_\_\_\_\_\_\_\_ |  | Дата: | \_16.11.2022\_\_\_\_\_\_\_\_\_ |
| Подпись: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  | Подпись: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

Москва, 2022 г.

**Постановка задачи**

Провести кодирование и декодирование текста (массива символов) при помощи кода Цезаря с переменным сдвигом по таблице ASCII-кодов. Величина сдвига для каждой позиции в исходном тексте - сумма (по модулю 256) кодов символов слова кодового блокнота, стоящего в блокноте на той же позиции. Если кодовый блокнот имеет слов меньше, чем количество символов в исходном тексте, то по исчерпании слов в нём перейти к первому слову и продолжить.

Исследовать повторяемость символов в закодированном тексте (сколько каких кодов одного и того же исходного символа получено) в зависимости от кодового блокнота и длины исходного текста. Результаты исследования представить в виде таблицы (продумать формат таблицы). Исследование и вывод таблицы результатов следует выполнять в режиме диалога, последовательно вычисляя и выводя результаты для запрашиваемого символа. Статистические данные хранить в массиве int array\_statistic[size][2], Для большей достоверности статистических результатов в качестве исходного текста и кодового блокнота использовать текстовые файлы размером около 1 Кбайта.

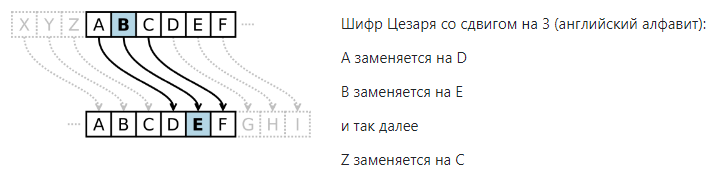
**Разработка алгоритма**

**Переменные:**

* int count\_keys - количество ключей для кодирования и раскодирования
* int size - кол-во символов в оригинальном тексте
* char tmp, unsigned char buffer - временные переменные
* ifstream file\_keys, ifstream original\_text, ofstream encrypted\_text- потоки файлов
* unsigned char symbol - символ для сбора статистики
* int array\_statistic[size][2] - массив для сбора статистики
* int last\_position - позиция последнего элемента в массиве
* int \*array\_keys - указатель на массив ключей
* int code\_symbol - код закодированного символа

**Шифр Цезаря:**

Шифр Цезаря — это вид шифра подстановки, в котором каждый символ в открытом тексте заменяется символом находящимся на некотором постоянном числе позиций левее или правее него в алфавите.



**Текст программы**

**Main.cpp**

/// Copyright 2022 ttsypyshev <ttsypyshev01@gmail.com>

#include <fstream>

#include <iostream>

#include <iomanip>

using namespace std;

int main() {

int count\_keys = 1, size = 0;

char tmp;

unsigned char buffer;

///discovery all files

ifstream file\_keys("../texts/keys.txt", ios\_base::in);

if (!(file\_keys)) {

cout << "Error of file1 opening." << endl;

return 1;

} else {

while (file\_keys.get(tmp)) {

if (tmp == ' ') { count\_keys++; }

}

file\_keys.clear();

file\_keys.seekg(0, ios\_base::beg);

}

ifstream original\_text("../texts/original.txt", ios\_base::in);

if (!(original\_text)) {

cout << "Error of file2 opening." << endl;

return 1;

}

ofstream encrypted\_text("../texts/encrypted.txt", ios\_base::trunc | ios\_base::out);

if (!(encrypted\_text)) { encrypted\_text.open("../texts/encrypted.txt"); }

///getting a letter for statistics

unsigned char symbol;

cout << "Enter the letter: ";

cin >> symbol;

int array\_statistic[size][2];

int last\_position = 0;

///creating am array of keys

int \*array\_keys = new int[count\_keys]();

int num = 0;

while (file\_keys.get(tmp)) {

if (tmp == ' ') {

num++;

} else {

buffer = tmp;

array\_keys[num] += buffer;

}

}

///encoding of the original text

int code\_symbol;

bool flag;

for (int i = 0; original\_text.get(tmp); ++i) {

buffer = tmp;

code\_symbol = (buffer + array\_keys[i % count\_keys]) % 256;

encrypted\_text << (unsigned char) code\_symbol;

flag = true;

if (buffer == symbol) {

for (int j = 0; j <= last\_position; ++j) {

if (array\_statistic[j][0] == code\_symbol) {

++array\_statistic[j][1];

flag = false;

break;

}

}

if (flag) {

array\_statistic[last\_position][0] = code\_symbol;

array\_statistic[last\_position][1] = 1;

++last\_position;

}

}

}

///statistics output

if (last\_position != 0) {

cout << setw(8) << "Symbol" << setw(10) << "Code" << setw(10) << "Count\n";

for (int i = 0; i < last\_position; ++i) {

cout << setw(8) << char(array\_statistic[i][0]) << setw(10) << array\_statistic[i][0] << setw(10)

<< array\_statistic[i][1] << endl;

}

}

delete[] array\_keys;

file\_keys.close();

original\_text.close();

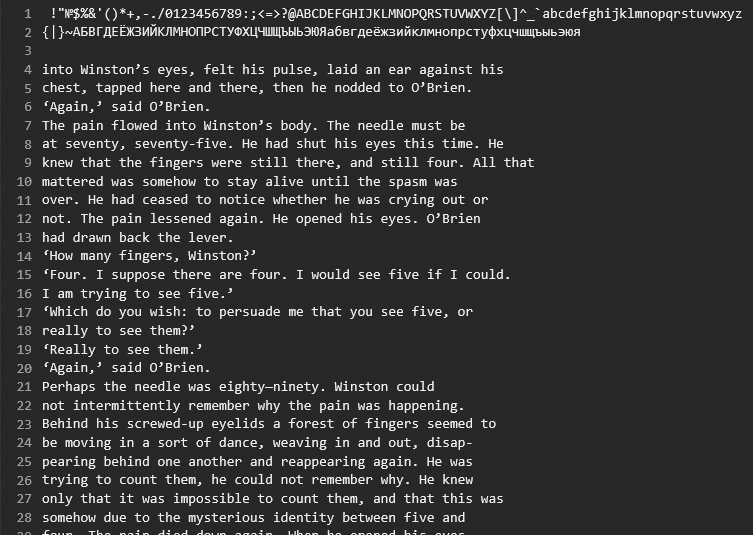
encrypted\_text.close();

return 0;

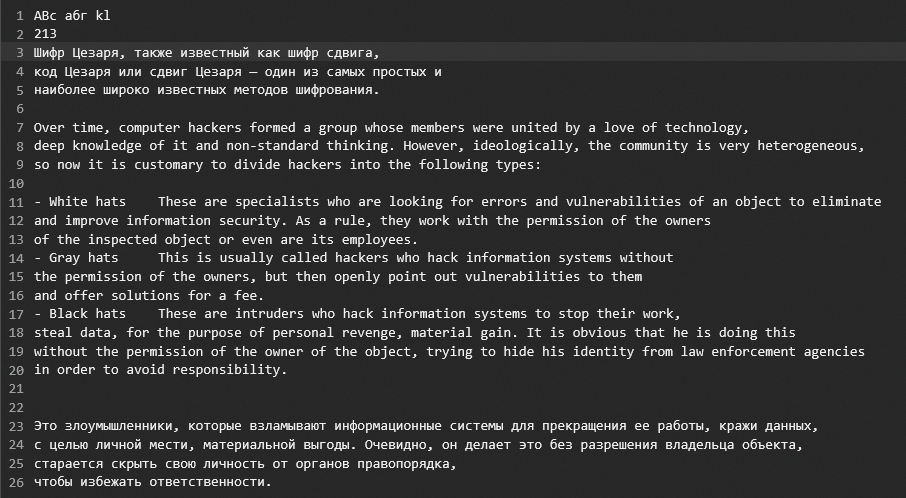
}

**Анализ результатов**

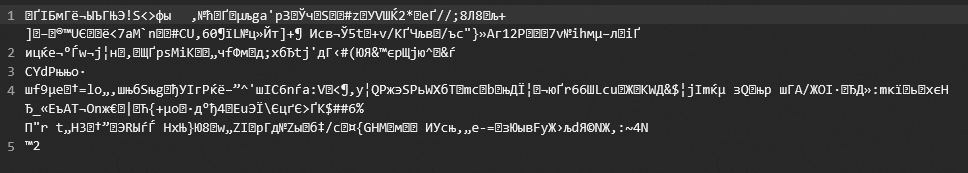
**Исходный файл:**

****

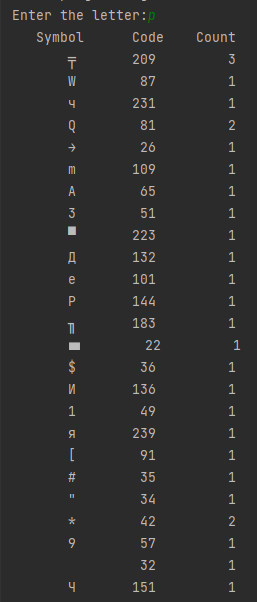
**Файл ключей:**

****

**Зашифрованный файл:**

****

**Статистика:**

****

**Вывод**

Я научился кодировать и декодировать текст (массива символов) при помощи кода с переменным сдвигом по таблице ASCII-кодов.