**Московский государственный технический**

**университет им. Н.Э. Баумана.**

Факультет «Информатика и управление»

Кафедра ИУ5. Курс «Основы программирования»

Отчет по лабораторной работе № 7

«Обработка символьных строк»

| Выполнил: Чернев Николай Андреевич |  | Проверил: |
| --- | --- | --- |
| студент группы ИУ5-14Б |  | преподаватель каф. ИУ5 |
|  |  |  |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2023 г.

Задание

Провести кодирование и декодирование текста (массива символов) при помощи кода Цезаря[[1]](#footnote-0) 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. (На основе кодового блокнота целесообразно сначала сформировать по заданному правилу целочисленный массив ключей, который затем использовать при кодировании. Эти действия оформить в виде отдельной функции.)

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 stat[256], Для большей достоверности статистических результатов в качестве исходного текста и кодового блокнота использовать текстовые файлы размером около 1 Кбайта.

Разработка алгоритма

Описание переменных

Функция setKeys:

* ifstream fin\_kw - переменная для считывания данных из файла keys.txt
* int key - текущее значение ключа
* int i - текущий индекс
* char c - вводимый символ

Функция print\_table:

* ofstream fout\_table - переменная для вывода таблицы в файл table.txt
* int p, int k - счетчики циклов

Функция encoding:

* char c - вводимый символ
* char encoded\_c - закодированный символ
* int i - индекс символа
* int ind\_col - индекс 26-х символов
* ifstream fin\_encoding - переменная для считывания данных из файла input.txt
* ofstream fout\_encoding - переменная для вывода в файл encoded.txt

Функция decoding:

* char encoded\_c - вводимый закодированный символ
* char c - раскодированный символ
* int i - индекс символа
* int ind\_col - индекс 26-х символов
* ifstream fin\_encoding - переменная для считывания данных из файла input.txt
* ofstream fout\_encoding - переменная для вывода в файл encoded.txt

main:

* int kw\_count - кол-во ключей
* int keys[1000] - массив ключей
* int collisions[1000] - массив индексов 26-х элементов
* int count[256][256] - двумерный массив статистики

Описание функций

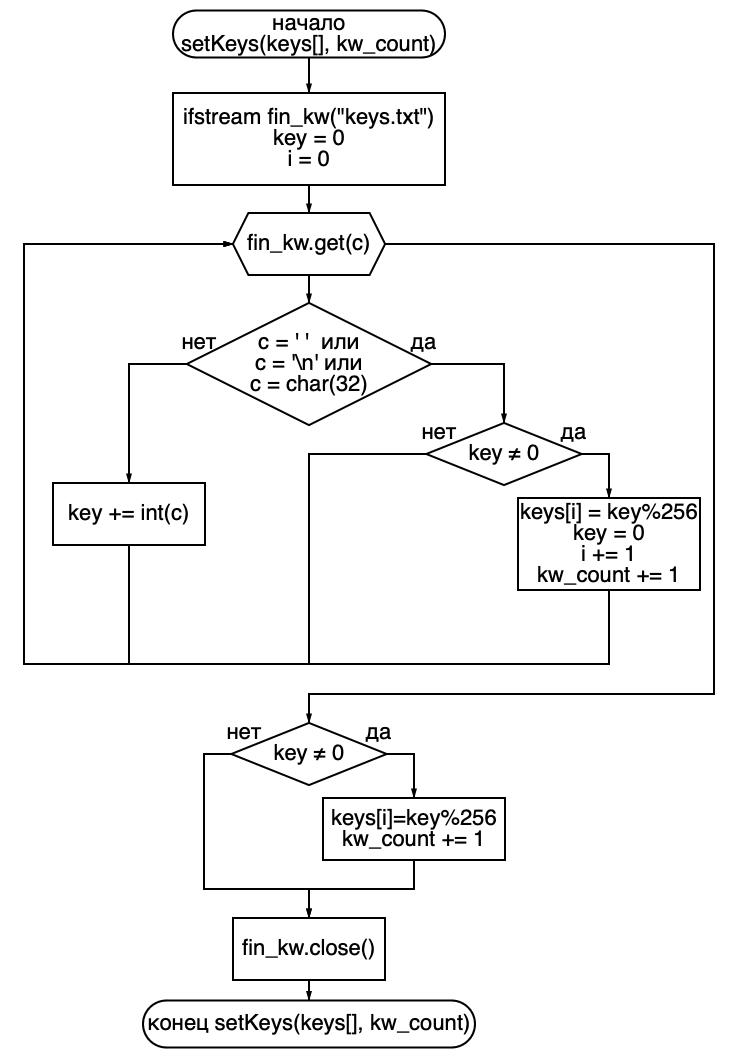
void setKeys(int\* keys, int& kw\_count) - принимает массив ключей и копию кол-ва ключей, считывает из файла ключи и записывает их в массив, ничего не возвращает

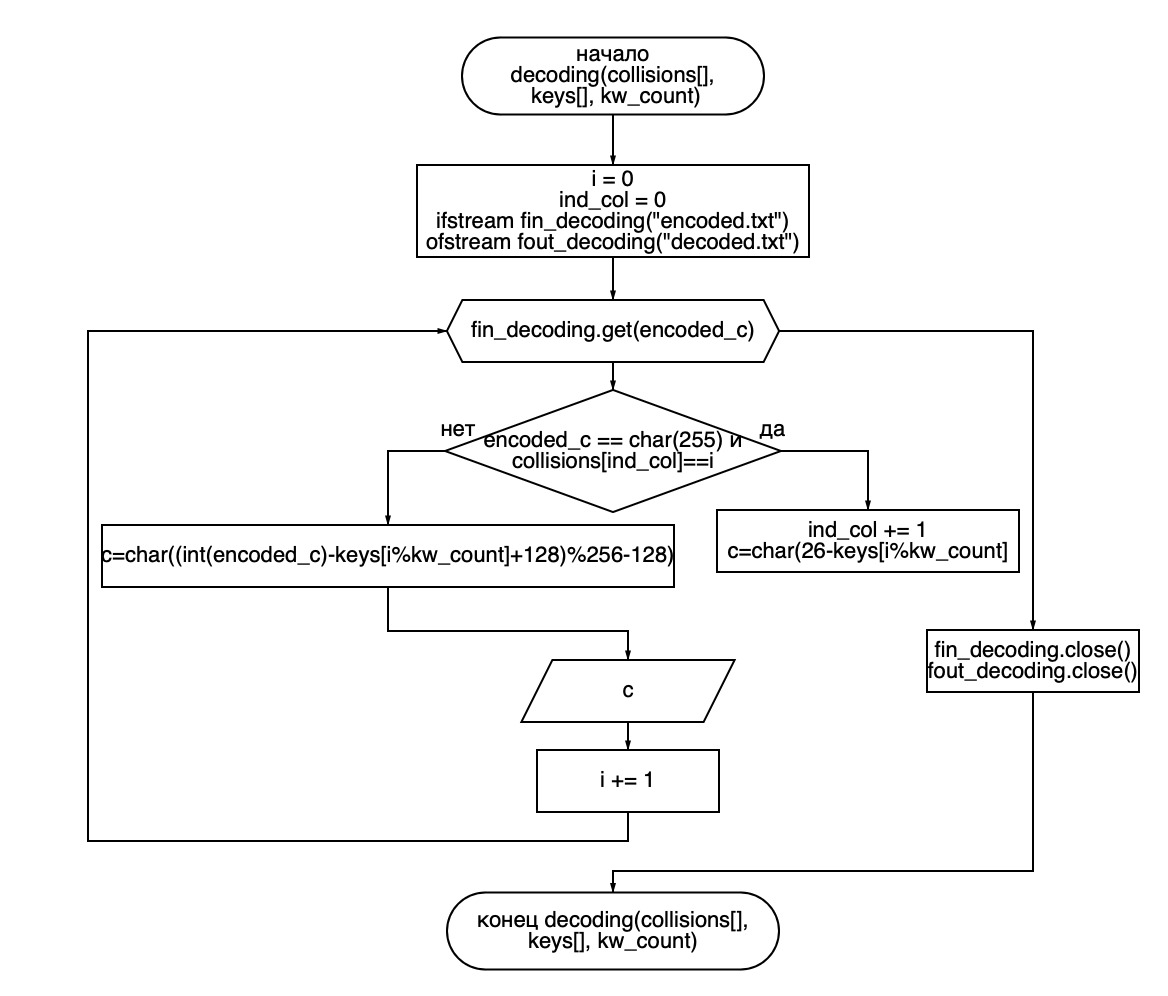
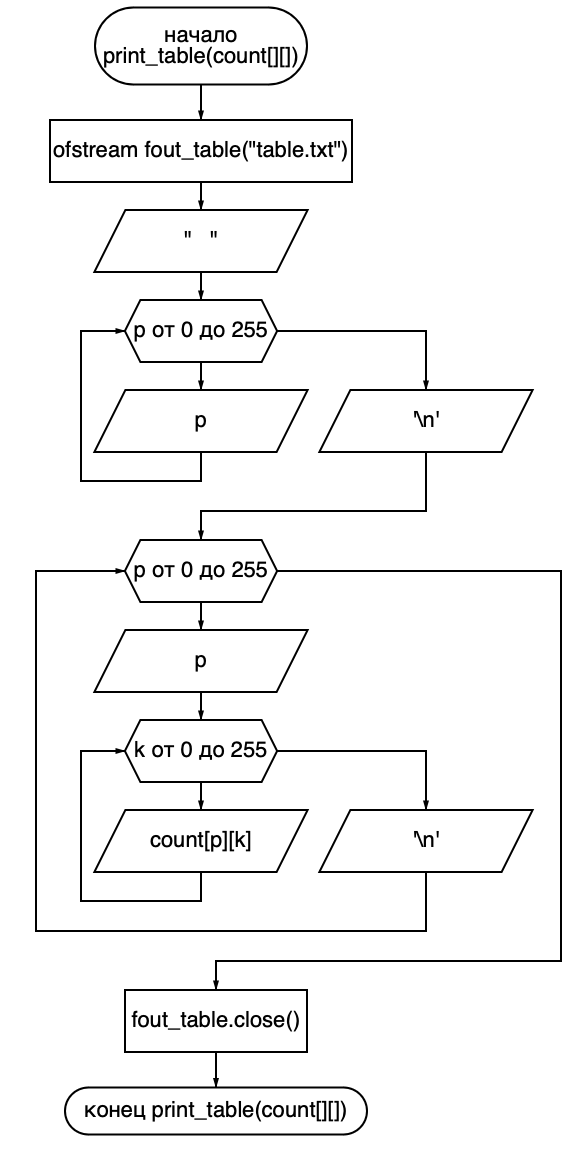
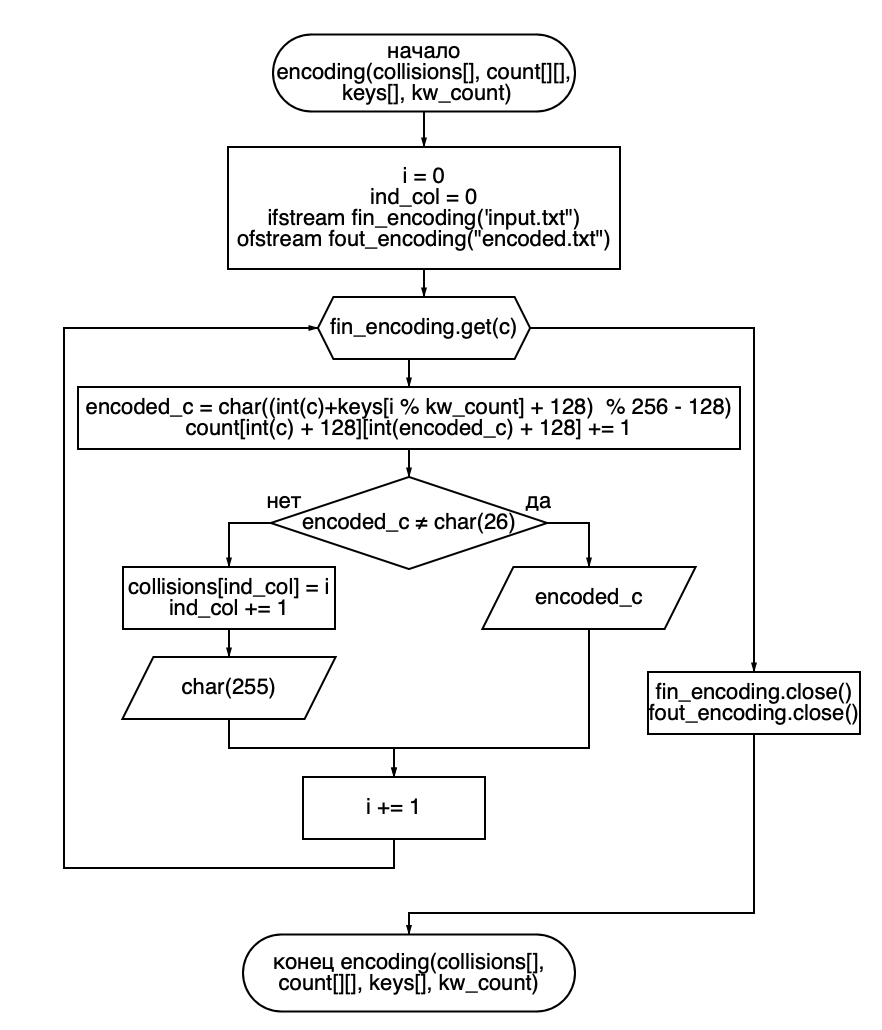
void encoding(int collisions[1000], int count[256][256], int keys[4000], int kw\_count) - принимает массив индексов 26х символов (collisions), двумерный массив для сбора статистики(count), массив ключей(keys), кол-во ключей kw\_count, считывает сообщение из файла input.txt, кодирует его и записывает в файл encoded.txt, ничего не возвращает

void print\_table(int count[256][256]) - принимает двумерный массив со статистикой, выводит таблицу со статистикой, ничего не возвращает

void decoding(int collisions[1000], int keys[4000], int kw\_count) - принимает массив индексов 26х символов collisions, массив ключей keys и кол-во ключей kw\_count, считывает закодированное сообщение из файла encoded.txt, раскодировывает его и выводит раскодированное сообщение в файл decoded.txt, ничего не возвращает

Блок-схема





Код программы

#include <fstream>

using namespace std;

void setKeys(int\* keys, int& kw\_count) {

ifstream fin\_kw("keys.txt");

int key = 0, i = 0;

char c;

while (fin\_kw.get(c)) {

if (c == ' ' || c == '\n' || c == char(32)) {

if (key != 0) {

keys[i] = key % 256;

key = 0;

i++;

kw\_count++;

}

}

else {

key += (int)c;

}

}

if (key != 0) {

keys[i] = key % 256;

kw\_count++;

}

fin\_kw.close();

}

void encoding(int collisions[1000], int count[256][256], int keys[4000], int kw\_count) {

char c, encoded\_c;

int i = 0;

int ind\_col = 0;

ifstream fin\_encoding("input.txt");

ofstream fout\_encoding("encoded.txt");

while (fin\_encoding.get(c)) {

encoded\_c = char((int(c) + keys[i % kw\_count] + 128) % 256 - 128);

count[int(c) + 128][int(encoded\_c) + 128]++;

if (encoded\_c != char(26)) {

fout\_encoding << encoded\_c;

}

else {

collisions[ind\_col++] = i;

fout\_encoding << char(255);

}

i++;

}

fin\_encoding.close();

fout\_encoding.close();

}

void print\_table(int count[256][256]) {

ofstream fout\_table("table.txt");

fout\_table << " ";

for (int p = 0; p < 256; p++){

fout\_table << setw(6) << p;

}

fout\_table << '\n';

for (int p = 0; p < 256; p++){

fout\_table << setw(3) << p;

for (int k = 0; k < 256; k++) {

fout\_table << setw(6) << count[p][k];

}

fout\_table << '\n';

}

fout\_table.close();

}

void decoding(int collisions[1000], int keys[4000], int kw\_count) {

ifstream fin\_decoding("encoded.txt");

ofstream fout\_decoding("decoded.txt");

int i = 0;

int ind\_col = 0;

char c, encoded\_c;

while (fin\_decoding.get(encoded\_c)) {

if (encoded\_c == (char)255 && collisions[ind\_col] == i) {

ind\_col++;

c = char(26 - keys[i % kw\_count]);

}

else {

c = char((int(encoded\_c) - keys[i % kw\_count] + 128) % 256 - 128);

}

i++;

fout\_decoding << c;

}

fin\_decoding.close();

fout\_decoding.close();

}

int main() {

int kw\_count = 0;

int keys[4000];

setKeys(keys, kw\_count);

int collisions[1000];

int count[256][256] = {0};

encoding(collisions, count, keys, kw\_count);

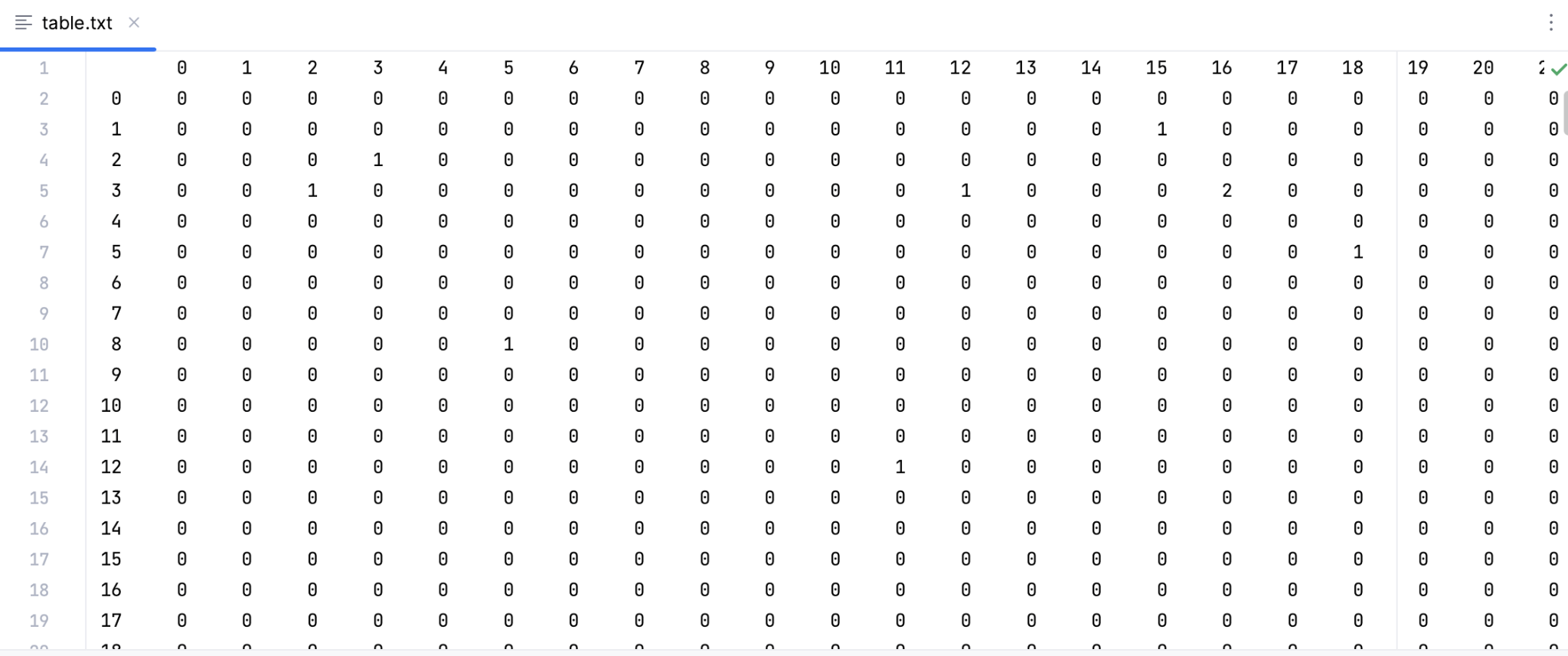
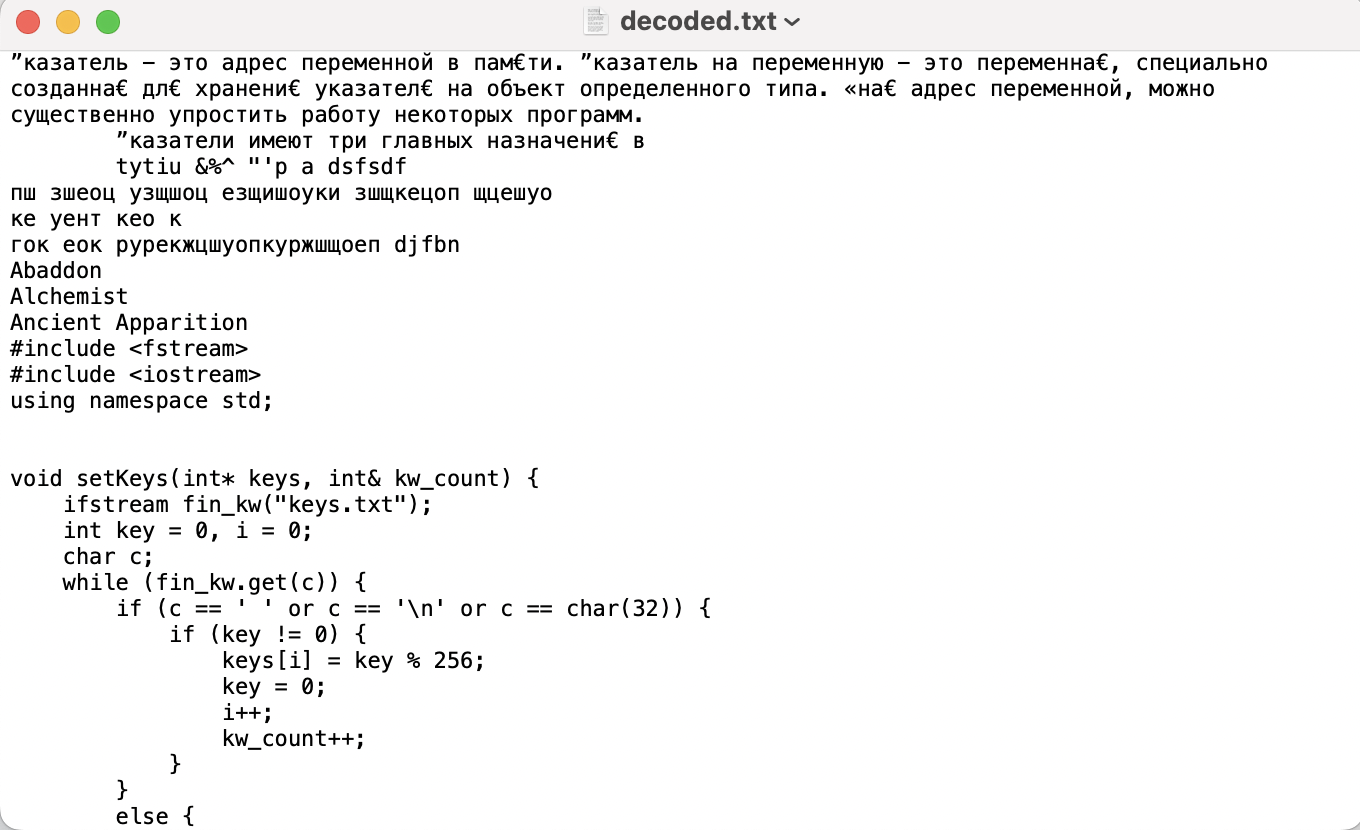
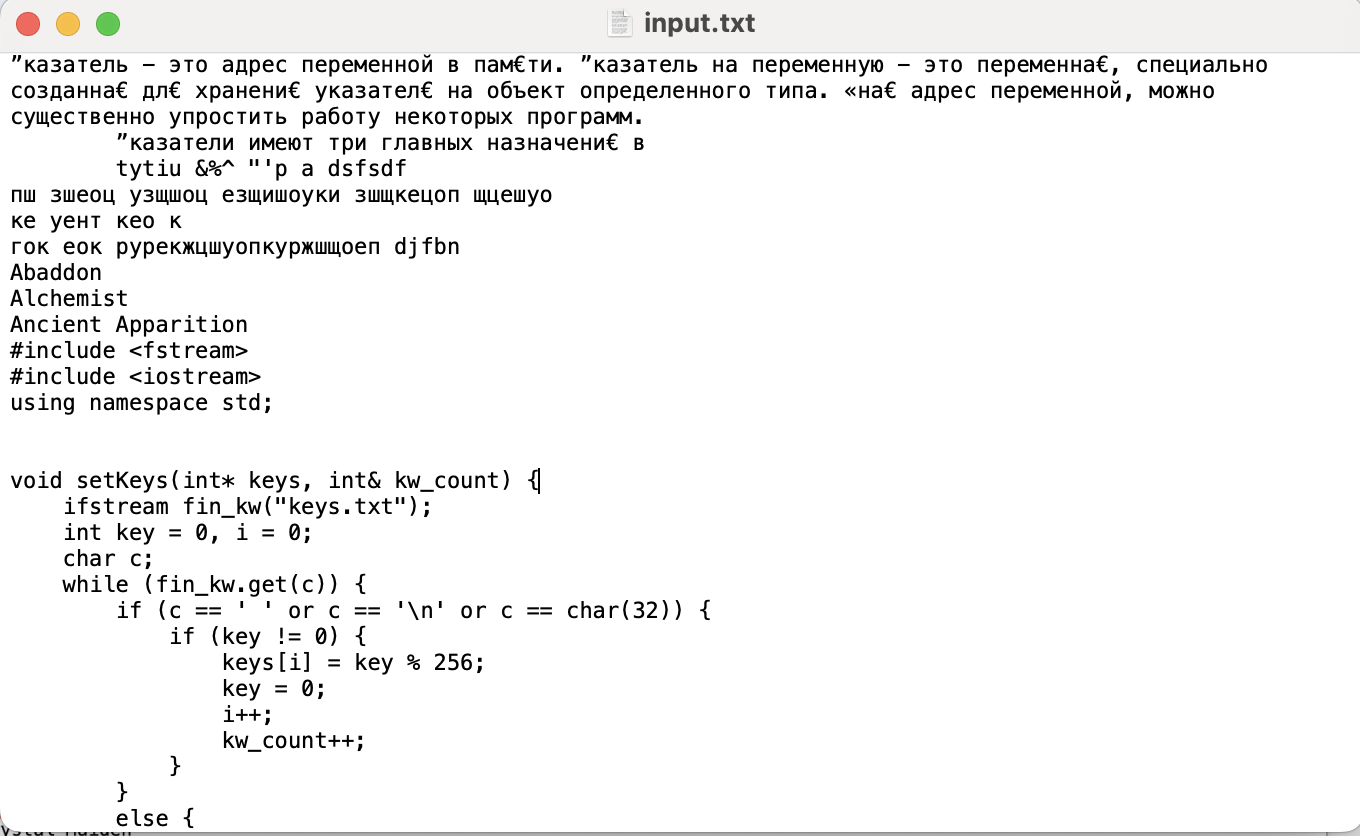
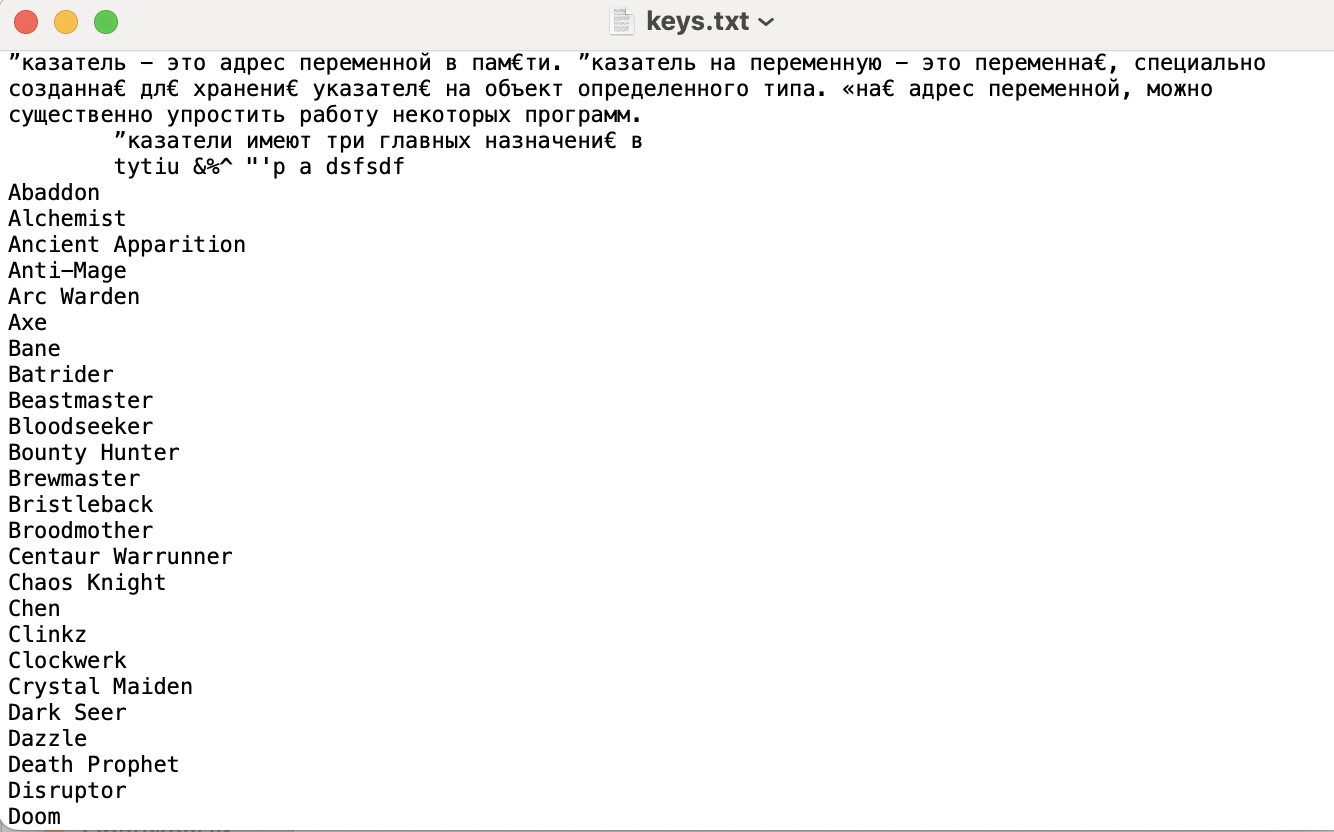
print\_table(count);

decoding(collisions, keys, kw\_count);

return 0;

}

Анализ результатов



1. Цезарь для шифрования своих посланий использовал следующий прием. При кодировании и декодировании писем он заменял буквы в письме на следующие по алфавиту буквы с постоянным смещением, равным 13, то есть к порядковому номеру буквы в латинском алфавите, содержащим 26 букв, он прибавлял 13 и получал порядковый номер буквы в кодируемом (декодируемом) письме. Если полученный таким образом номер буквы был больше 26, то он уменьшался на 26. [↑](#footnote-ref-0)