Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра вычислительных технологий**

**ЛАБОРАТОРНАЯ РАБОТА №1**

**Дисциплина: Информационная безопасность**

Работу выполнил: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Романов В.В.

Направление подготовки: 02.03.02 Фундаментальная информатика и информационные технологии

Преподаватель: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Шиян В.И.

**Цель работы:** научиться проектировать алгоритмы поддержки информационной безопасности.

**Задание:** необходимо разработать алгоритмы и программы решения двух задач на языке C++.

**Ход работы:**

Вариант 14. Задания 1, 2.

1. Шифровка. Штирлиц хочет передать очень важное сообщение 𝑠 в штаб. Для этого он использует беспрефиксный код, зафиксированный по ГОСТ. К сожалению, противнику известен код, а канал связи Штирлица прослушивается. Чтобы отвести подозрения, Штирлиц хочет разбить шифровку на куски, каждый из которых нельзя расшифровать тем же кодом. Для максимальной безопасности Штирлиц хочет, чтобы количество кусков было максимально. Найдите максимальное количество кусков или определите, что разбить шифровку требуемым образом невозможно.

*Исходные данные:*

В первой строке записано одно целое число 𝑘 (1 ≤ 𝑘 ≤ 52) – количество символов в алфавите. Символы пронумерованы от 1 до 52 в порядке A-Za-z, в тексте сообщения будут использованы только символы с номерами от 1 до 𝑘.

Во второй строке записана строка 𝑠 длины до 106, состоящая из символов с номерами от 1 до 𝑘 (нумерация символов определена выше).

В следующих 𝑘 строках записаны двоичные коды символов алфавита согласно нумерации. Каждый символ алфавита кодируется последовательностью 0 и 1 длины не более 𝑘. Гарантируется, что никакой код не является префиксом другого кода.

*Результат:*

Выведите одно число – максимальное количество кусков, либо -1, если разбить на куски указанным способом невозможно.

|  |  |
| --- | --- |
| исходные данные | результат |
| 3  CACB 011  1  001 | 2 |
| 3  ACBABCAACABCAACC 0  10  110 | -1 |

*Замечания:*

В первом семпле зашифрованный текст выглядит как 0010110011. Единственный способ разбить его на куски, которые нельзя расшифровать – 0 010110011. Следовательно, ответ равен 2.

Во втором семпле можно показать по индукции, что любая строчка, которая заканчивается на 0 и не содержит трёх 1 подряд, расшифровывается данным кодом. Любой суффикс зашифрованного текста имеет такой вид, поэтому ответ – -1.

1. Шифрограмма. Слово зашифровано «циклическим сдвигом на 3 символа назад». Например, слово СЕЛО, шифр – ОСЕЛ, слово КООРДИНАТА, шифр – РДИНАТАКОО. Написать функцию decode, которая по заданному шифру восстанавливает зашифрованное слово. Задан зашифрованный как указано выше текст, например, в виде текстового файла. Слова в тексте разделяются одним или несколькими пробелами. Написать программу, использующую функцию decode, расшифровки этого текста, например, в результирующий текстовый файл.

Для решения поставленных задач были разработаны 2 программы на языке программирования C++ в среде разработки Visual Studio 2022.

На рисунках 1-2 представлены результаты работы программы №1 (декодирование сообщений Штирлица).

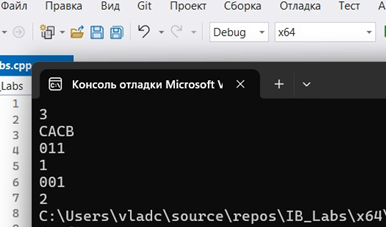


Рисунок 1 – Результат на первом наборе данных.

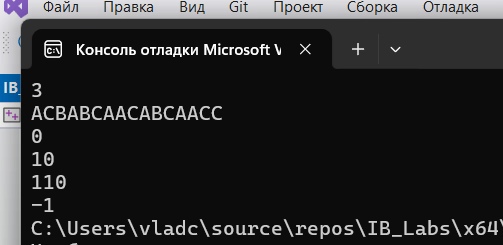


Рисунок 2 – Результат на втором наборе данных.

Далее на рисунке 3 представлен результат работы программы №2 (преобразование входного слова циклическим сдвигом символов на 3 и вывод результатов в текстовый файл).

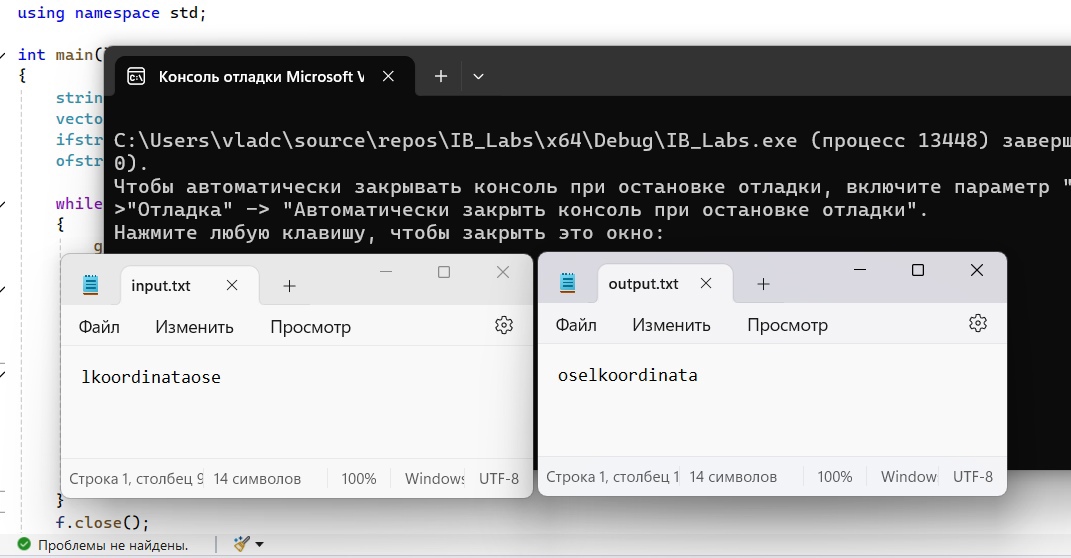


Рисунок 3 – Результат на втором наборе данных.

**Вывод:** научился проектировать алгоритмы поддержки информационной безопасности, а также реализовал задания 1 и 2.

**Листинг программы:**

**Файл task1.cpp:**

#include "iostream"

#include "vector"

#include <math.h>

#include <tuple>

#include <stdio.h>

#include <algorithm>

#include <queue>

using namespace std;

typedef long long ll;

//Проверяет, существует ли данная строка в списке кодов.

bool isCodsContainsWord(string s, vector<string>& cods) {

for (int i = 0; i < cods.size(); i++) {

if (s == cods[i]) {

return true;

}

}

return false;

}

//Определяет, может ли строка быть декодирована с использованием предоставленных кодов.

bool isCanDecode(string s, vector<string>& cods) {

if (s == "") {

return true;

}

bool flag = false;

for (int i = 1; i <= s.length() && i < 7; i++) {

if (isCodsContainsWord(s.substr(0, i), cods)) {

flag = isCanDecode(s.substr(i), cods);

if (flag) {

return true;

}

}

}

return false;

}

//Кодирует строку на основе пользовательских кодов.

string encode(string s, vector<string>& cods) {

string code = "";

for (int i = 0; i < s.length(); i++) {

code += cods[s[i] - 65];

}

return code;

}

int maxCount = -1;

//Пытаемся разделить закодированную строку на допустимые сегменты, которые могут быть декодированы.

void tryToSeparate(string s, vector<string>& cods, int count) {

for (int i = 1; i <= s.length() && i < 7; i++) {

if (!isCanDecode(s.substr(0, i), cods)) {

if (!isCanDecode(s.substr(i), cods)) {

if (maxCount < count + 1) {

maxCount = count + 1;

}

}

tryToSeparate(s.substr(i), cods, count + 1);

}

}

}

int main()

{

ll k;

cin >> k;

string s;

cin >> s;

vector<string> cods = vector<string>(k);

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

string tmp;

cin >> tmp;

cods[i] = tmp;

}

tryToSeparate(encode(s, cods), cods, 1);

cout << maxCount;

}

**Файл task2.cpp:**

#include <iostream>

#include <fstream>

#include <string>

#include <sstream>

#include <vector>

using namespace std;

int main()

{

string s, s2, t; int i;

vector <string> str;

ifstream f("C://Users/vladc/OneDrive/Рабочий стол/input.txt");

ofstream h("C://Users/vladc/OneDrive/Рабочий стол/output.txt");

while (!f.eof())

{

getline(f, s);

stringstream s1(s);

while (s1 >> s2)

{

str.push\_back(s2);

}

for (i = 0; i < str.size(); i++)

{

t = str[i];

rotate(t.rbegin(), t.rbegin() + 3, t.rend());

h << t;

}

}

f.close();

h.close();

}