Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Пермский национальный исследовательский политехнический университет»

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы»

Направление 09.03.01 – «Информатика и вычислительная техника»

Дисциплина: «Защита информации»

Профиль: «Автоматизированные системы обработки информации и управления»

Семестр 7

ОТЧЕТ

по лабораторной работе №1

Тема: «Шифры перестановки и замены»

Выполнил: студент группы АСУ-17-1б

Хохряков Денис

Проверил: доцент кафедры ИТАС

Шереметьев В.Г.\_\_\_\_\_\_\_\_\_\_

Дата\_\_\_\_\_\_

# ЦЕЛЬ РАБОТЫ

Получить практические навыки по использованию шифров замены. Испытать, как приходилось защищать информацию пару веков назад.

# ЗАДАНИЕ

Реализовать шифрование текстового сообщения, используя систему одноразового шифра.

# КРАТКИЕ ТЕОРИТИЧЕСКИЕ СВЕДЕНИЯ

Одноразовая система изобретена Дж. Моборном и Г. Вернамом. Ее суть довольно проста – для каждого сообщения составляется страница, которая подшивается в одноразовый блокнот, который имеется у той стороны, которая отправляет сообщение, и у той, которая принимает. При чем каждая страница используется при шифровании и дешифрованнии только один раз, после чего уничтожается. Такая схема позволяет получить безусловно надежный шифр, так как, не зная ключа, даже при наличии неограниченных вычислительных возможностей, восстановить исходный текст невозможно, так как таких вариантов будет бесконечное количество.

Казалось бы – вот она, идеальная система шифрования. Однако здесь заключается проблема в передаче этого самого блокнота. Ведь длина ключа должна быть не меньше самого сообщения. И если для передачи коротких сообщений это еще выполнимо, то для того количества информации, которая передается сегодня, это затруднительно.

Также существует вариант, что вместо одноразового блокнота используется указание места в определенной книге. И с этого места начинается ключ. Этот вариант весьма удобен, однако, так как ключ не является случайным, существует возможность восстановить сообщение, используя, например, информацию о частотах букв естественного языка.

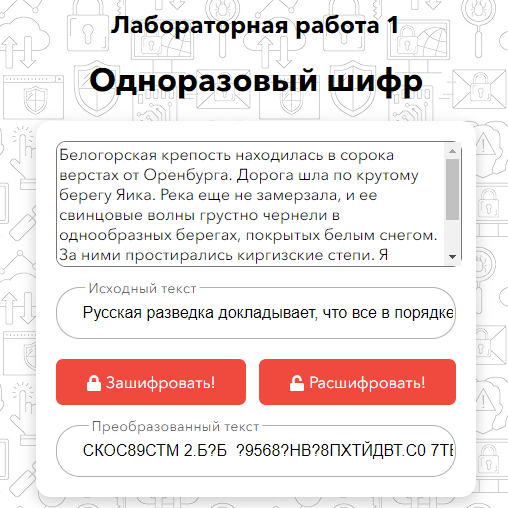
В моей реализации данной системы шифровании используется алфавит из букв русского алфавита (только верхнего регистра), цифр и знаков препинания. С помощью данных символов удобно передавать информацию для русскоязычного народа.

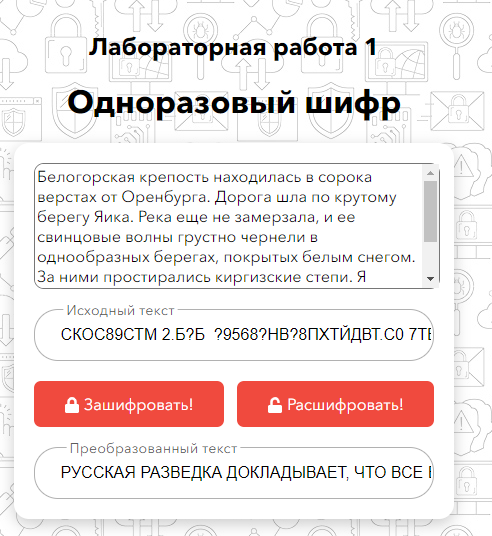
Исходный текст побуквенно складывается по модулю с текстом ключа. Таким образом, получается криптостойкий шифр, который невозможно получить без знания ключа.

# ХОД РАБОТЫ

Весь интерфейс оформим в виде формы, в которую записывается ключ (например, текст из книги), а также исходный текст, который нужно либо зашифровать, либо расшифровать.

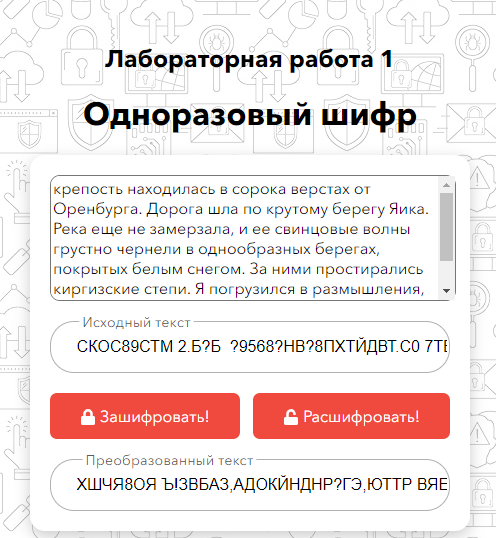
Ввод осуществляется в любом регистре, а после обработки все переводится в верхний регистр. Все символы, которых нет в алфавите (латиница и пр.) заменяются простым пробелом.

В качестве примера в качестве ключа возьмем отрывок из книги «Капитанская дочка», а в качестве текста какую-нибудь случайную фразу, которая не должна попасть третьему лицу. Как видим, зашифрованный текст представляет собой какой-то набор символов, который можно расшифровать бесконечным количеством способов.

Дальше вторая сторона получает зашифрованный текст, и, зная ключ, начинает расшифровывать.

Таким образом, информация передана и, даже если был перехвачен канал связи, информация не попадет третьей стороне.

Затем ключ уничтожается и больше не используется



Также допустим, что третья сторона знает о литературных увлечениях шифровальщиков и попробует использовать текст из «Капитанской дочки», однако ошибется с позицией текста. Как видно из рисунка, получился какой-то еще более странный набор символов, из которого совершенно непонятны намерения шифровальщиков.

# ЛИСТИНГ ПРОГРАММЫ (Язык – JavaScript)

export const encrypt = (input, key) => {

let output = '';

const \_input = input.toUpperCase();

for(let i = 0; i < input.length; i++){

const k = convert(\_input.charCodeAt(i));

const k2 = convert(key.charCodeAt(i));

let k3 = k+k2;

if(k3 > 46) k3 -= 46;

console.log(unConvert(k3));

output += String.fromCharCode(unConvert(k3));

}

return output;

}

export const decrypt = (input, key) => {

let output = '';

const \_input = input.toUpperCase();

for(let i = 0; i < input.length; i++){

const k = convert(\_input.charCodeAt(i));

const k2 = convert(key.charCodeAt(i));

let k3 = k-k2;

if(k3 < 0) k3 += 46;

console.log(unConvert(k3));

output += String.fromCharCode(unConvert(k3));

}

return output;

}