Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика, искусственный интеллект и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

ОТЧЕТ

по лабораторной работе № 1

по курсу «Защита информации»

на тему: «Программная реализация электронного аналога «Энигмы»

Студент ИУ7-73Б (Группа) _____ (Подпись, дата) Марченко В. (И. О. Фамилия)

Преподаватель _____ (Подпись, дата) Чиж И. С. (И. О. Фамилия)

2023 г.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

«Энигма» — переносная шифровальная машина, использовавшаяся для шифрования и расшифрования секретных сообщений.

Целью данной лабораторной работы является программная реализация алгоритма шифрования, который использовался в шифровальной машине «Энигма».

Задачи лабораторной работы:

1) изучить принцип работы шифровальной машины «Энигма»;
2) разработать программное обеспечение для шифрования текста из файла с помощью алгоритма шифрования «Энигмы»;
3) протестировать разработанное программное обеспечение.

1 Шифровальная машина «Энигма»

Первую версию роторной шифровальной машины запатентовал в 1918 году Артур Шербиус [1].

«Энигма» состояла из комбинации механических и электрических систем. Механическая часть включала в себя клавиатуру, набор вращающихся дисков — роторов — которые были расположены вдоль вала и прилегали к нему, и ступенчатого механизма, двигающего один или несколько роторов при каждом нажатии на клавишу. Электрическая часть состояла из электрической схемы, соединяющей между собой клавиатуру, коммутационную панель, лампочки и роторы [1].

Общий принцип функционирования «Энигмы»: при каждом нажатии на клавишу самый правый ротор сдвигается на одну позицию, а при определенных условиях сдвигаются и другие роторы. Движение роторов приводит к различным криптографическим преобразованиям при каждом следующем нажатии на клавишу на клавиатуре [2].

Основные части «Энигмы» — клавиатура, коммутационная панель, три ротора (иногда больше) и рефлектор.

Кабель, помещенный на коммутационную панель, соединял буквы попарно, например, «E» и «Q» могли быть соединены в пару. Эффект состоял в перестановке этих букв до и после прохождения сигнала через роторы. Например, когда оператор нажимал «E», сигнал направлялся в «Q», и только после этого уже во входной ротор [1].

Рефлектор соединял контакты последнего ротора попарно, коммутируя ток через роторы в обратном направлении, но по другому маршруту [3]. Наличие рефлектора гарантировало, что преобразование, осуществляемое «Энигмой», есть инволюция, то есть расшифрование представляет собой то же самое, что и шифрование [4]. Однако наличие рефлектора делает невозможным шифрование какой-либо буквы через саму себя. Это было серьезным концептуальным недостатком, впоследствии пригодившимся дешифровщикам [1].

2 Алгоритм шифрования

На рисунке 2.1 показан пример шифрования буквы «Z». В примере используется коммутационная панель, три ротора типов I, II, и III и рефлектор. Все вычисления выполняются в кольце вычетов по модулю 26 (кол-во символов латинского алфавита).

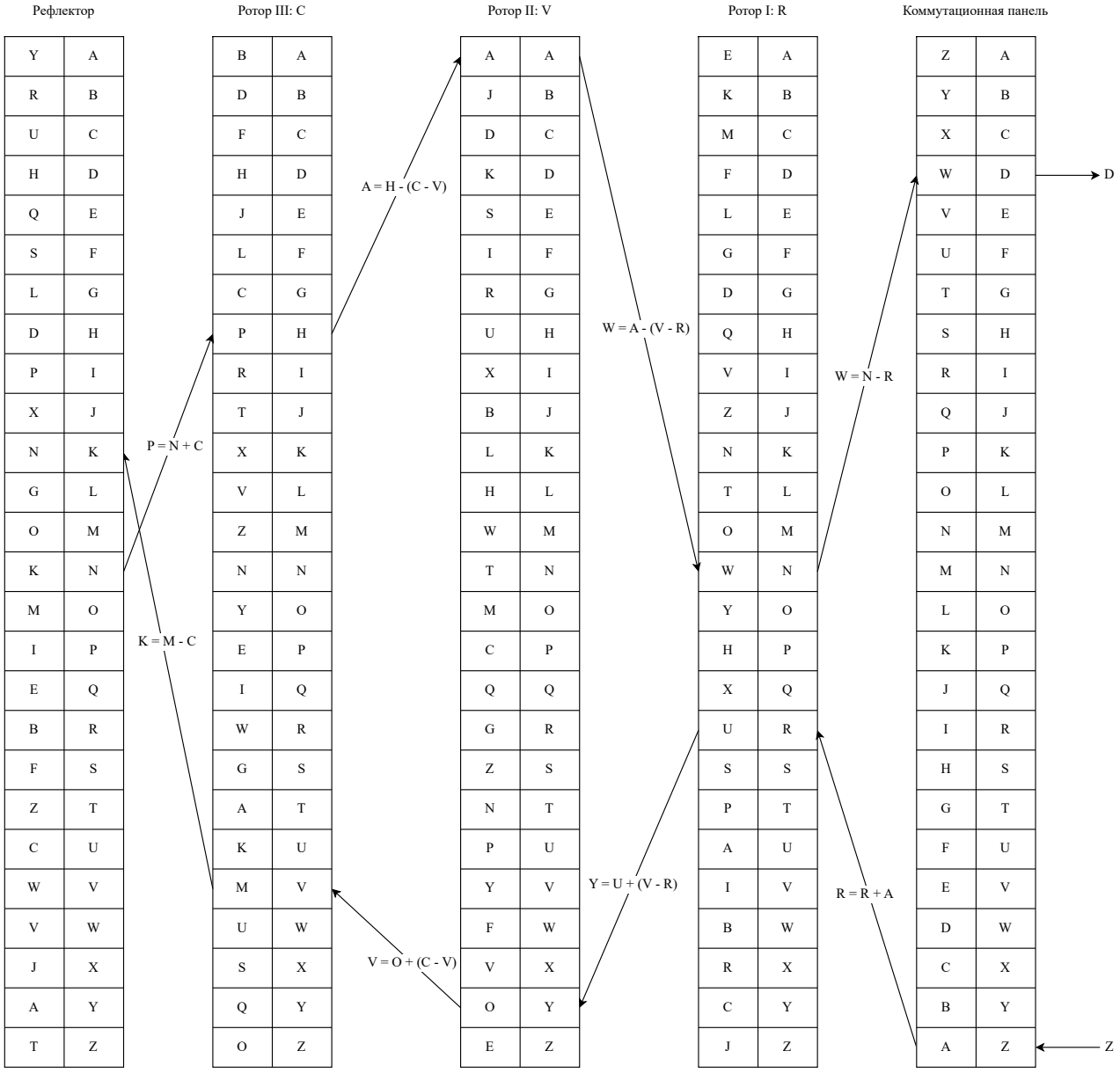

Рисунок 2.1 – Пример шифрования буквы с помощью «Энигмы»

У ротора любого типа есть определенная буква, при повороте которой сдвигается соседний левый ротор. У некоторых типов роторов таких букв может быть две или даже три. Таким образом, каждая буква проходит следующие преобразования: через коммутационную панель, через три ротора,

через рефлектор, через три ротора в обратном порядке и еще раз через коммутационную панель. Притом правый ротор сдвигается при каждом нажатии на клавишу.

3 Программная реализация

Требования к входным данным. Программа принимает два обязательных и два дополнительных аргумента командной строки. Первый дополнительный аргумент — путь к файлу, который содержит конфигурацию коммутационной панели. Второй дополнительный аргумент — путь к файлу, который содержит конфигурацию рефлектора. Конфигурация — пары больших символов латинского алфавита. Каждая пара на следующей строке. Например, «AU». Тогда «A» будет заменяться на «U», а «U» — на «A». Третий обязательный аргумент — путь к файлу, содержащий текст, который нужно зашифровать. Четвертый обязательный аргумент — путь к файлу, в который будет записан зашифрованный текст.

Если первые два параметра не указываются, будет установлена конфигурация коммутационной панели и рефлектора по-умолчанию.

Данная реализация «Энигмы» работает только с буквами латинского алфавита. На вход можно подавать буквы любого регистра. На выходе всегда будет латинская буква в верхнем регистре.

При наличии ошибок в аргументах командной строки или в тексте, который нужно зашифровать, программа выдаст сообщение об ошибке и завершится.

В каталоге config есть три текстовых файла с конфигурациями роторов. В первой строке расположены все буквы ротора, во второй — т. н. «notch», а в третьей — значение начальной позиции ротора.

Программное обеспечение для шифрования текста с помощью алгоритма «Энигмы» было написано на языке программирования C++.

Программа состоит из точки входа — функции main — и классов Steckerbrett (коммутационная панель), Rotor, Reflector и Enigma. Класс Enigma является главным в программе. Он содержит в себе объект класса Steckerbrett, Reflector и три объекта класса Rotor. В листингах 3.1–3.4 представлены интерфейсы этих классов.

Листинг 3.1 – Интерфейс класса Steckerbrett

```
class Steckerbrett
{
public:
    Steckerbrett() = default;
```

```
    Steckerbrett(const Steckerbrett& steckerbrett);
    Steckerbrett(const std::string filename);
    char Encrypt(const char symbol);
private:
    std::vector<std::string> _symbols;
};
```

Листинг 3.2 – Интерфейс класса Rotor

```
class Rotor
{
public:
    Rotor();
    Rotor(const std::string symbols, const char notch, const
       char current_pos);
    Rotor& operator ++ ();
    char GetKeyByValue(const char index);
    char operator [] (const char index);
    char GetCurrentPos();
    char GetNotch();
private:
    const std::string _alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
    std::string _symbols;
    char _notch;
    char _current_pos;
};
```

Листинг 3.3 – Интерфейс класса Reflector

```
class Reflector
{
public:
    Reflector() = default;
    Reflector(const Reflector& reflector);
    Reflector(const std::string filename);
    char Reflect(const char symbol);
private:
    std::vector<std::string> _symbols;
};
```

Листинг 3.4 – Интерфейс класса Enigma

```
class Enigma
{
public:
```

```
    Enigma(const Steckerbrett& steckerbrett, const Reflector&
       _reflector);
    std::string Encrypt(std::string initial_string);
    void RotateRotors();
    char EncryptRotorRight(char symbol);
    char EncryptRotorMiddle(char symbol);
    char EncryptRotorLeft(char symbol);
    char EncryptRotorRightBack(char symbol);
    char EncryptRotorMiddleBack(char symbol);
    char EncryptRotorLeftBack(char symbol);
private:
    Steckerbrett _steckerbrett;
    Rotor _rotor_left;
    Rotor _rotor_middle;
    Rotor _rotor_right;
    Reflector _reflector;
};
```

В листинге 3.5 представлена реализация алгоритма шифрования «Энигмы».

Листинг 3.5 – Реализация алгоритма шифрования «Энигмы»

```
std::string Enigma::Encrypt(std::string initial_string)
{
    std::string encrypted_string;
    char symbol;
    for (std::size_t i = 0; i < initial_string.size(); ++i)
    {
        symbol = initial_string[i];
        symbol = this->_steckerbrett.Encrypt(symbol);
        RotateRotors();
        symbol = EncryptRotorRight(symbol);
        symbol = EncryptRotorMiddle(symbol);
        symbol = EncryptRotorLeft(symbol);
        symbol = this->_reflector.Reflect(module_div(symbol -
           INDEX -
                this->_rotor_left.GetCurrentPos(), MOD) +
                   INDEX);
        symbol = EncryptRotorLeftBack(symbol);
        symbol = EncryptRotorMiddleBack(symbol);
        symbol = EncryptRotorRightBack(symbol);
        symbol = this->_steckerbrett.Encrypt(module_div((symbol
```

```
            - INDEX -
                  this->_rotor_right.GetCurrentPos()), MOD) +
                     INDEX);
        encrypted_string += symbol;
    }
    return encrypted_string;
}
```

В листинге 3.6 показан пример запуска программы.

Листинг 3.6 – Пример запуска программы

```
./app.exe input.txt output.txt
./app.exe config/steckerbrett.txt config/reflector.txt input.txt
   output.txt
```

В листинге 3.7 показан пример работы программы. На первой строке показан исходный текст, на второй — зашифрованный, на третьей — расшифрованный. А на четвертой строке показано сообщение, которые является результатом расшифрования при неправильных начальных позициях роторов.

Листинг 3.7 – Пример работы программы

```
HELLOWORLD
LWCERDNNOU
HELLOWORLD
NGQJDASZKX
```

ЗАКЛЮЧЕНИЕ

В результате выполнения данной лабораторной работы был реализован алгоритм шифрования, который использовался в шифровальной машине «Энигма».

Были выполнены следующие задачи:

1) изучен принцип работы шифровальной машины «Энигма»;

2) разработано программное обеспечение для шифрования текста из файла с помощью алгоритма шифрования «Энигмы»;

3) протестировано разработанное программное обеспечение.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. *Википедия*. Энигма. — 2023. — (Дата обращения: 15.09.2023). `https://ru.wikipedia.org/wiki/РңР,,РчРуРёРө/`.
2. *Stripp A.* Codebreakers: The Inside Story of Bletchley Park. — 1993.
3. *Сингх С.* Книга шифров. Тайная история шифров и их расшифровки // Астрель. — 2007.
4. *Бауэр Ф.* Расшифрованные секреты. Методы и принципы криптологии // Мир. — 2007.