ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Московский институт электроники и математики им. А. Н. Тихонова

ПРОГРАММНАЯ РЕАЛИЗАЦИЯ ШИФРОВАЛЬНОЙ МАШИНЫ «ЭНИГМА» НА ЯЗЫКЕ СИ

Программный проект
по специальности 10.05.01 «Компьютерная безопасность» по предмету
«Программирование логических интегральных схем» выполнен
студентом 3 курса СКБ212
Нагаевой Вероникой Игоревной

Москва 2024 г.

АННОТАЦИЯ

Целью работы является создание программной реализации шифровальной машины «Энигма» времен Второй Мировой войны. Кодирование происходит на английском языке. Требуется следующее:

1. Написать структуру данных, отражающую модель шифровальной машины.
2. Разработать аналог алгоритма шифрования/дешифрования на языке C:
    a. Роторы
    b. Рефлектор
3. Соблюсти принципы инкапсуляции.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Шифровальная машина «Энигма» внешне выглядит как печатающая машинка, за исключением того факта, что шифруемые символы не печатаются автоматически на определённый лист бумаги, а указываются на панели посредством загорания лампочки. Шифровальная машина «Энигма» обладает двумя основными механизмами:

1. Роторы. Они реализуют полиалфавитный алгоритм шифрования, а их определённо выстроенная позиция представляет собой ключ шифрования. Каждый ротор уникален и обладает собственной настройкой. Военным на выбор давалось пять роторов, три из которых они вставляли в «Энигму». Выбор позиций, для вставки роторов, также играл свою роль, потому как устройство сохраняло свойство некоммутативности. Каждый ротор обладал 26-тью гранями, где каждая грань представляла собой нумерацию английского алфавита.

2. Рефлектор (отражатель). Статичный механизм, позволяющий шифровальным машинам типа «Энигма» не вводить помимо операции шифрования дополнительную операцию расшифрования. Рефлектор также гарантировал, что буква не будет зашифрована сама собой.

Мною были реализованы алгоритм шифрования/дешифрования и функция подбора ключа при известных конфигурациях роторов и рефлектора.

ГЛАВА 1. КОМПИЛЯЦИЯ И ИСПОЛЬЗОВАНИЕ ПРОГРАММЫ

1.1. Компиляция

Для удачной компиляции программы пользователь должен импортировать файл “enigma.c”, написать алгоритм использования в main и скомпилировать программу. Например, ввести в терминал команду: *gcc filename.c -o out*

1.2. Использование

Входной файл – программа не работает с входными файлами. Пользовательский ввод отсутствует. Взаимодействие с программой происходит через импорт необходимых функций.

Выходной файл – файл стандартного ввода/вывода, в который будут записаны данные после зашифрования/расшифрования.

1.3. Пример использования

Зашифрование текст: *import “enigma.c”*

*crypt(struct enigma_t *enigma, const char *msg, char *key, char *alphabet);*

Расшифрование текста: *import “enigma.c”*

*crypt(struct enigma_t *enigma, const char *cypher, char *key, char *alphabet);*

ГЛАВА 2. СТРУКТУРА И ОСНОВНЫЕ ФУНКЦИИ “enigma.c”

enigma_t: структура, являющаяся моделью физической реализации энигмы.

Содержит:

```
int size_rotor; - размер ротора
int num_rotors; - количество роторов
char *reflector; - массив с данными для рефлектора
char **rotors; - матрица с массивами данных для роторов
```

Функции:

```
extern enigma_t *constructor(int size_rotor, int num_rotors)
```

Конструктор, выделяющий память под все слоты структуры.

Параметры: *int size_rotor, int num_rotors.*

Возвращаемое значение: *enigma_t *.*

```
extern void set_reflector(enigma_t *enigma, char *reflector)
```

Инициализация рефлектора для структуры.

Параметры: *enigma_t *enigma, char *reflector.*

Возвращаемое значение: *void.*

```
extern void set_rotor(enigma_t *enigma, int num, char *rotor)
```

Инициализация ротора для структуры.

Параметры: *enigma_t *enigma, int num, char *rotor.*

num – номер инициализируемого ротора.

Возвращаемое значение: *void.*

```
static int get_alpha_code(char letter)
```

Возвращение кода символа из таблицы ASCII.

Параметры: *char letter.*

Возвращаемое значение: *int.*

```
static int get_index_alpha(enigma_t *enigma, char letter, int num_rotor)
```

Возвращение позиции символа в роторе.

Параметры: *enigma_t *enigma, int num_rotor, char letter.*

Возвращаемое значение: *int.*

```
extern char *crypt(enigma_t *enigma, const char *phrase, char *key, char *alphabet)
```

Алгоритм шифрования/дешифрования.

Параметры: *enigma_t *enigma, const char *phrase, char *key, char *alphabet.*

Возвращаемое значение: *char.*

```
extern void destructor(enigma_t *enigma)
```

Деструктор, осуществляет очистку выделенной в конструкторе памяти.

Параметры: *enigma_t *enigma.*

Возвращаемое значение: *void.*

ГЛАВА 3. АЛГОРИТМ РАБОТЫ МАШИНЫ “ЭНИГМА”

Как было описано во введении, базовая модель Энигмы состоит из двух основных частей:

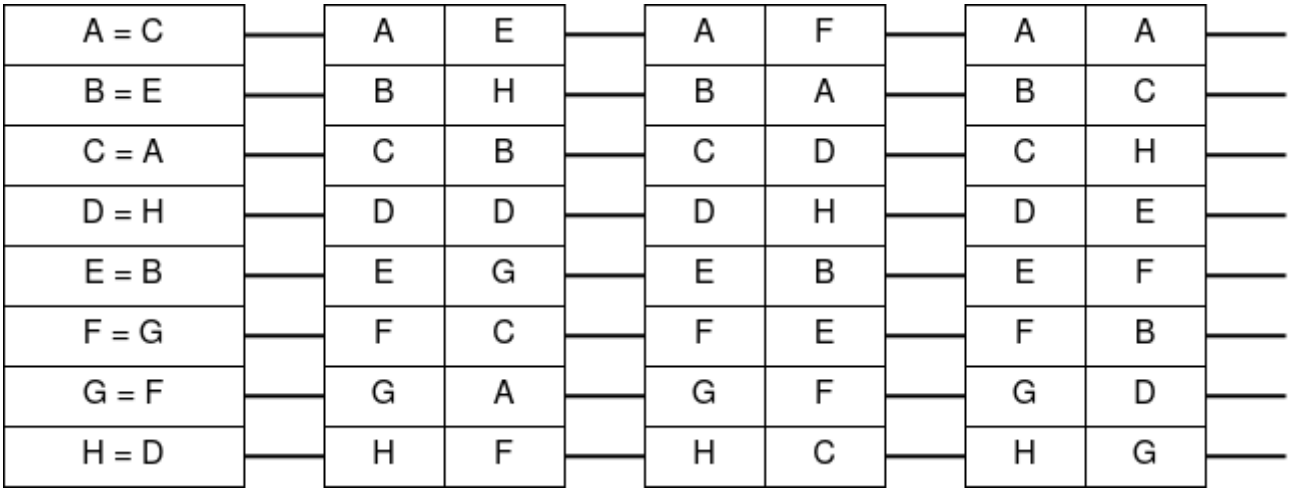

Рис. 1. Внутреннее строение Энигмы

Рефлектор (слева) является статичным элементом. Принцип его работы напоминает зеркало. Полученный символ отражается от рефлектора согласно прописанной настройке.

Отражатель подключен к трём последовательно соединенным роторам. Каждый ротор включает в себя 2 части: статичную и подвижную. Первый ротор (справа) проворачивается на одну позицию на каждый новый символ. Последующие два ротора поворачиваются на одну позицию каждый раз, когда предыдущий к ним ротор прошёл полный круг.

Например, если шифруется три раза символ A, то он станет равен сначала символу C, потом символу G, а потом символу E. Это можно показать на следующих схемах:

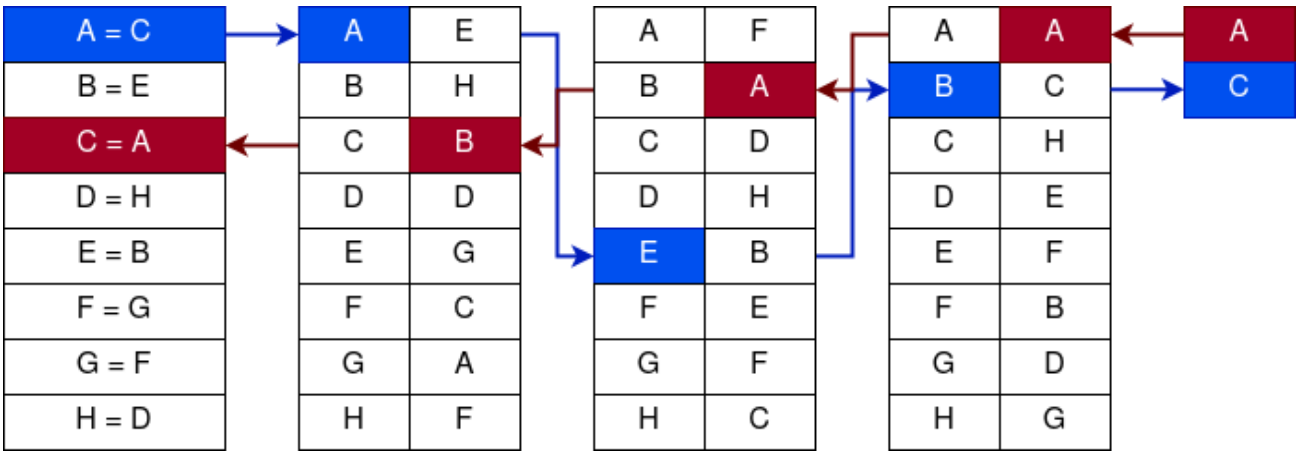

Рис. 2. Первый круг

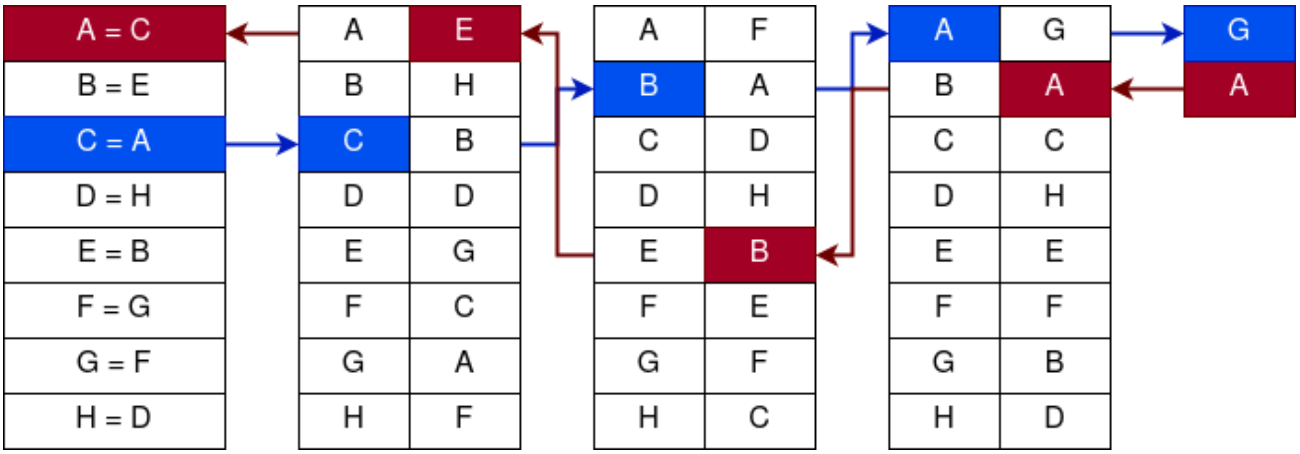

Рис. 3. Второй круг

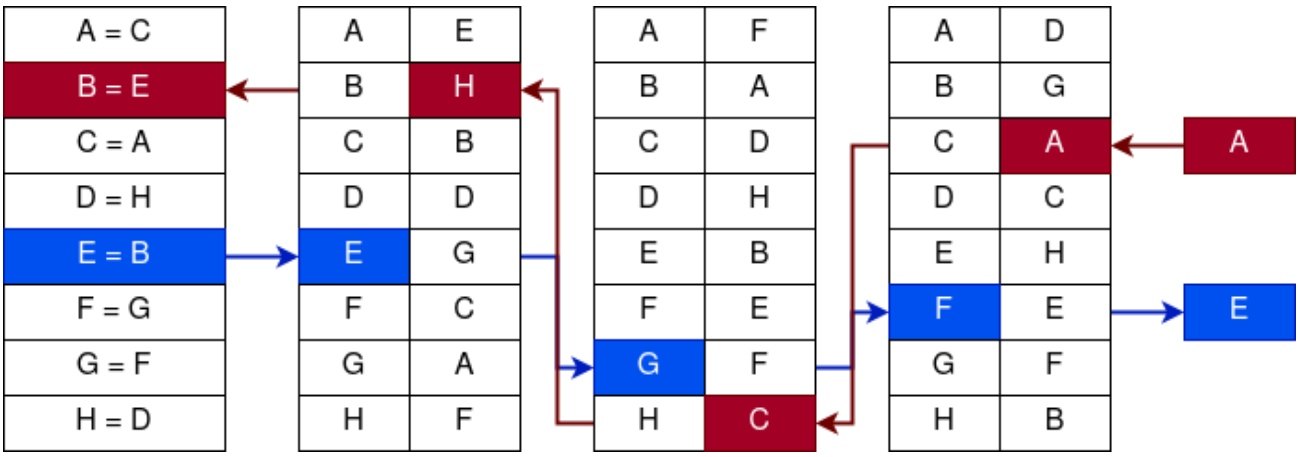

Рис. 4. Третий круг

ИСПОЛЬЗУЕМАЯ ЛИТЕРАТУРА

Книга Nigel Smart «Cryptography: An Introduction (3rd Edition)»
(*https://www.cs.umd.edu/~waa/414-F11/IntroToCrypto.pdf*)
Статья «Программная реализация шифровальной машины «Энигма»»
(https://habr.com/ru/articles/721790/)