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 ИУ «Информатика и системы управления»

КАФЕДРА ИУ-7 «Программное обеспечение ЭВМ и информационные технологии»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1

по дисциплины «Защита информации»

«Шифрофальная машина Энигма»

Студент группы **ИУ7-76Б** _____ **В. М. Мансуров**
(Подпись, дата) (И.О. Фамилия)

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

2023 г.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Шифрование информации — занятие, которым человек занимался ещё до начала первого тысячелетия, занятие, позволяющее защитить информацию от посторонних лиц.

Шифровальная машина «Энигма» — одна из самых известрых шифровальнных машин, использовавшихся для шифрования и расшифровывания секретных сообщений [1].

Целью данного лаборотоной работы является проектирование и разработка программную реализацию машины «Энигмы».

Чтобы достигнуть поставленной цели, требуется решить следующие задачи:

— провести анализ работы шифровальной машины «Энигмы»;

— описать алгоритм шифрования;

— реализовать и протестировать реализацию алгоритма шифрования.

1 Аналитическая часть

В этом разделе будут рассмотрены классический алгоритм работы шифровальной машины «Энигма», а также её вариант, использованный во время Второй мировой войны, приведён пример преобразования буквы, а также подсчитано количество комбинаций «Энигмы» с 3 роторами.

1.1 Механизмы шифрования

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 тремя основными механизмами.

1) Роторы — сердце всех шифровальных машин, которое со стороны классической криптографии они реализуют полиалфавитный алгоритм шифрования, а их определённо выстроенная позиция представляет собой один из основных ключей шифрования. Каждый ротор не эквивалентен другому ротору, потому как обладает своей специфичной настройкой. Военным на выбор давалось пять роторов, три из которых они вставляли в «Энигму».
2) Рефлектор — статичный механизм, позволяющий шифровальным машинам типа «Энигма» не вводить помимо операции шифрования дополнительную операцию расшифрования. Связано это с тем, что в терминологии классической криптографии рефлектор представляет собой просто частный случай моноалфавитного шифра.
3) Коммутатор позволяет оператору шифровальной машины варьировать содержимое проводов, попарно соединяющих буквы английского алфа-

вита.

1.2 Алгоритм работы шифрования

В данной работе будет подразумеваться, что у оператора машины состоит из 3 роторов и 1 рефлекторов, а также 26 соединительных проводов для коммутационной панели:

— на вход поступает файл с данными и посимвольно считывается;

— каждый символ поступает в коммутационную панель, благодаря чему поставляется постановленый парный код символа;

— затем данный код поступает в каждый ротор, где осуществляется преобразованые в новый код символа;

— после 3 роторов код символа поступает в рефлектор и сопоставляется парный код символа;

— данный код в обратном направлении проходит через все роторы;

— новый код поступает в коммутатор и ему сопоставляется соотвествующая пара;

— получаем шифрованную букву;

— первый ротор поворачивается на одну позицию, если один ротор совершит полный оборот всех позиций, то менять позицию начнет следующий ротор.

2 Конструкторская часть

В этом разделе представлена схема алгоритма шифровальной машины «Энигма».

2.1 Разработка алгоритмов

На рисунке 2.1 приведена схема работы шифровальной машины Энигма.

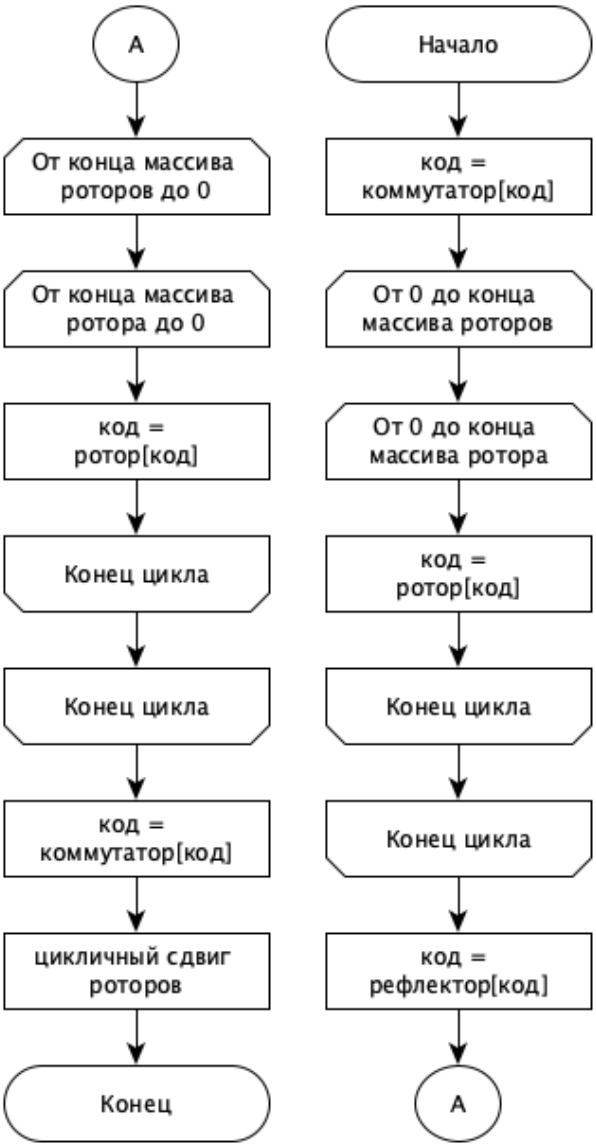

Рисунок 2.1 – Схема работы шифровальной машина Энигма

3 Технологическая часть

3.1 Средства реализации

Для программной реализации шифровальной машины был выбран язык C++ [2]. В данном языке есть все требующиеся инструменты для данной лабораторной работы. В качестве среды разработки была выбрана среда CLion [3].

3.2 Реализация алгоритма

Листинг 3.1 – Класс рефлектора

```
1  class Reflector {
2  public:
3      explicit Reflector();
4      explicit Reflector(const std::vector<uint8_t>& config);
5      explicit Reflector(const std::string& config);
6
7      uint8_t reflect(uint8_t symbol);
8      void printf_config();
9  private:
10     std::vector<uint8_t> _config;
11 };
12
13 Reflector::Reflector(const std::vector<uint8_t>& config):
       _config(config) {}
14
15 uint8_t Reflector::reflect(uint8_t symbol) {
16     return _config[symbol];
17 }
```

Листинг 3.2 – Класс ротора

```
1  class Rotor {
2  public:
3      explicit Rotor(const std::vector<uint8_t>& wiring);
4
5      uint8_t encrypt_left(uint8_t symbol);
6      uint8_t encrypt_right(uint8_t symbol);
7
8      void set_position(uint8_t position);
9      void reset_position();
10     void rotate();
11 private:
12     uint8_t find_index(uint8_t letter);
13     std::vector<uint8_t> _wiring;
14     std::vector<uint8_t> _start;
15     size_t _size;
16 };
17
18
19 Rotor::Rotor(const std::vector<uint8_t>& wiring):
       _wiring(wiring), _start(_wiring) {
20     this->_size = this->_wiring.size();
21 }
22
23 uint8_t Rotor::encrypt_right(uint8_t symbol) {
24     return _wiring[symbol];
25 }
26
27 uint8_t Rotor::encrypt_left(uint8_t symbol) {
28     return find_index(symbol);
29 }
30
31 uint8_t Rotor::find_index(uint8_t letter) {
32     for (int i = 0; i < _size; i++)
```

```
33      {
34          if (_wiring[i] == letter)
35          {
36              return i;
37          }
38      }
39
40      return -1;
41  }
42
43  void Rotor::reset_position() {
44      _wiring = _start;
45  }
46
47  void Rotor::rotate()
48  {
49      uint8_t temp = _wiring[_size - 1];
50      for (int i = _size - 1; i > 0; --i) {
51          _wiring[i] = _wiring[i - 1];
52      }
53      _wiring[0] = temp;
54  }
```

Листинг 3.3 – Класс енигмы

```
1  class Enigma {
2  public:
3      Enigma();
4      Enigma(uint64_t size_rotor, uint8_t amount_rotors);
5
6      void set_reflector(Reflector& reflector);
7      void set_commutator(Reflector& reflector);
8      void set_rotor(Rotor& rotor);
9
10     size_t encrypt(FILE *fin, FILE *fout);
11     std::string encrypt(const std::string& message);
```

```
12      uint8_t encrypt(uint8_t symbol);
13      void reset_rotors();
14
15  void printf_config();
16      private:
17      void print_pair(uint8_t s1, uint8_t s2);
18      char normalize_sym(uint8_t symbol);
19      int _counter;
20      int _size_rotor;
21      uint8_t _amount_rotors;
22      std::unique_ptr<Reflector> _reflector;
23      std::unique_ptr<Reflector> _commutator;
24      std::vector<std::shared_ptr<Rotor>> _rotors;
25  };
26
27  Enigma::Enigma(uint64_t size_rotor, uint8_t amount_rotors) {
28      this->_counter = 0;
29      this->_size_rotor = (int) size_rotor;
30      this->_amount_rotors = amount_rotors;
31  }
32
33  void Enigma::set_reflector(Reflector &reflector) {
34      _reflector = std::make_unique<Reflector>(reflector);
35  }
36
37  void Enigma::set_commutator(Reflector &reflector) {
38      _commutator = std::make_unique<Reflector>(reflector);
39  }
40
41  void Enigma::set_rotor(Rotor &rotor) {
42      _rotors.push_back(std::make_shared<Rotor>(rotor));
43  }
44
45  uint8_t Enigma::encrypt(uint8_t symbol)
46  {
```

```
47      int rotor_queue = 0;
48      uint8_t new_symbol = symbol;
49
50      new_symbol = _commutator->reflect(new_symbol);
51
52      for (int i = 0; i < _amount_rotors; i++)
53      {
54          new_symbol = _rotors[i]->encrypt_left(new_symbol);
55      }
56
57      new_symbol = _reflector->reflect(new_symbol);
58
59
60      for (int i = _amount_rotors - 1; i >= 0; i--)
61      {
62          new_symbol = _rotors[i]->encrypt_right(new_symbol);
63      }
64
65      new_symbol = _commutator->reflect(new_symbol);
66
67      rotor_queue = 1;
68      this->_counter += 1;
69      for (int i = 0; i < _amount_rotors; ++i) {
70          if (_counter % rotor_queue == 0) {
71              _rotors[i]->rotate();
72          }
73          rotor_queue *= _size_rotor;
74      }
75
76      return new_symbol;
77  }
78
79  void Enigma::reset_rotors() {
80      for (auto rotor: _rotors)
81      {
```

```
82          rotor->reset_position();
83      }
84  }
85
86  std::string Enigma::encrypt(const std::string& message) {
87      std::string new_message;
88      for (char symbol: message)
89      {
90          new_message += static_cast<char>(encrypt(symbol));;
91      }
92
93      return new_message;
94  }
95
96  size_t Enigma::encrypt(FILE *fin, FILE *fout) {
97      if (fin == nullptr || fout == nullptr)
98      {
99          return -1;
100     }
101
102     std::wstring message;
103
104     char code;
105     fseek(fin , 0, SEEK_END);
106     size_t input_size = ftell(fin);
107     fseek(fin , 0, SEEK_SET);
108     size_t size = 0;
109
110
111     while (size < input_size) {
112         size += fread(&code, sizeof(char), 1, fin);
113         fseek(fin , SEEK_SET, SEEK_CUR);
114
115         uint8_t newcode = this->encrypt(code);
116
```

```
117             fwrite(&newcode, sizeof(char), 1, fout);
118         }
119
120         return size;
121 }
```

3.3 Тестирование

Таблица 3.1 – Функциональные тесты

Входная строка	Выходная строка
HeLlo WorLd	»кйД;поль6
ABOBA	ґµ;№M
ґµ;№M	ABOBA
A	ґ
«»	«»

ЗАКЛЮЧЕНИЕ

В результате лабораторной работы были изучены принципы работы шифровальной машины «Энигма», была реализована программа, способная шифровать и дешифровать текстовый файл. Были решены следующие задачи:

1) проведен анализ работы шифровальной машина «Энигма»;
2) описан алгоритм шифрования;
3) реализован и протестирован описанный алгоритм;

Список использованных источников

1. И.М. Шолин. Алгоритм переносной шифровальной машины энигма. — Кубанский государственный технологический университет.

2. Язык программирования C++. `https://learn.microsoft.com/en-us/cpp/cpp/cpp-language-reference?view=msvc-170`. дата обращения: 15.10.2023.

3. CLion. `jetbrains.com`. дата обращения: 15.10.2023.