МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Изучение устройства и функциональных особенностей шифровальной машины «Энигма»

Студент: Евсеенко В. П.

ФИТ 3 курс 4 группа

Преподаватель: Савельева М. Г.

Минск 2025

Содержание

[Введение 3](#_Toc201142483)

[1 Практическое задание 4](#_Toc201142484)

[Задание 1 4](#_Toc201142485)

[Задание 2 4](#_Toc201142486)

[2 Гистограммы 9](#_Toc201142487)

[3 Криптостойкость 10](#_Toc201142488)

[Вывод 11](#_Toc201142489)

[Приложение А 12](#_Toc201142490)

# **Введение**

Цель лабораторной работы:

* изучение и приобретение практических навыков разработки и использования приложений для реализации перестановочных шифров.

Задачи:

* закрепить теоретические знания по алгебраическому описанию, алгоритмам реализации операций зашифрования/расшифрования и оценке криптостойкости подстановочно-перестановочных шифров;
* изучить структуру, принципы функционирования, реализацию процедур зашифрования сообщений в машинах семейства «Энигма»;
* изучить и приобрести практические навыки выполнения криптопреобразований информации на платформе «Энигма», реализованной в виде симуляторов;
* получить практические навыки оценки криптостойкости подстановочных и перестановочных шифров на платформе «Энигма».

# **1 Практическое задание**

Машина «Энигма» – это электромеханическое устройство. Как и другие роторные машины, «Энигма» состоит из комбинации механических и электрических подсистем. Механическая часть включает в себя клавиатуру, набор вращающихся дисков – роторов, которые расположены вдоль вала и прилегают к нему, и ступенчатого механизма, двигающего один или несколько роторов при каждом нажатии на клавишу. Электрическая часть, в свою очередь, состоит из электрической схемы, соединяющей между собой клавиатуру, коммутационную панель, лампочки и роторы (для соединения роторов использовались скользящие контакты).

При каждом нажатии на клавишу самый правый ротор сдвигался на одну позицию, а при определенных условиях сдвигались и другие роторы. Движение роторов приводило к различным криптографическим преобразованиям при каждом следующем нажатии на клавишу на клавиатуре, т. е. зашифрование/расшифрование сообщений основано на выполнении ряда замен (подстановок) одного символа другим.

Механические части двигались и, замыкая контакты, образовывали меняющийся электрический контур. При нажатии на клавишу клавиатуры контур замыкается, ток проходит через созданную (для зашифрования/расшифрования одного конкретного символа сообщения) цепь и в результате включает одну из набора лампочек, отображающую искомую букву шифртекста (или расшифрованного сообщения

# **Задание 1**

Произвести зашифрование сообщения (собственные имя, отчество, фамилия) при 8–10 различных настройках машины-симулятора. Оценить частотные свойства символов в шифртекстах и сравнить этот параметр с частотными свойствами символов для исходного текста.

Для выполнения зашифрования сообщения был выбран ресурс симуляторы энигмы «[Enigma Simulator](https://piotte13.github.io/enigma-cipher/)». Результаты зашифрования приведены в Приложении А.

# **Задание 2**

Разработать приложение-симулятор шифровальной машины, состоящей из клавиатуры, трех роторов и отражателя. Типы роторов (*L* – *M* – *R*) и отражателя *Re* следует выбрать из рисунке 1.1 и 1.2.



Рисунок 1.1 – Спецификация на роторы «Энигмы»

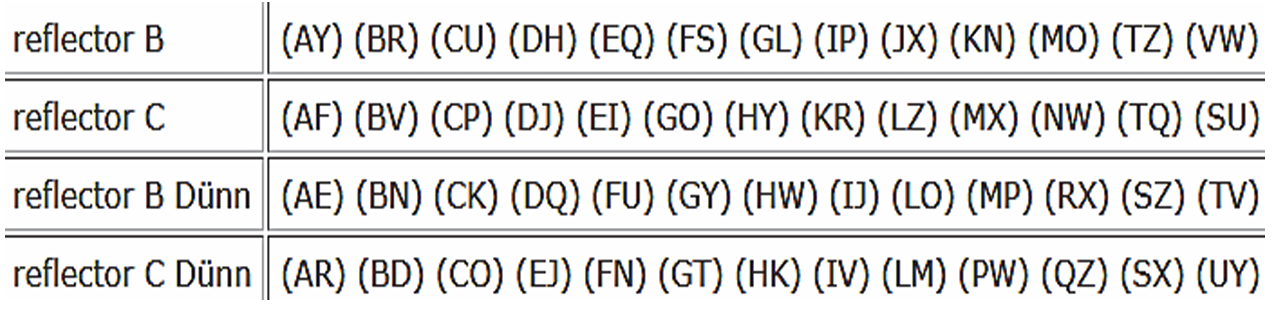


Рисунок 1.2 – Спецификация на рефлекторы «Энигмы»

Выбирать типы роторов и отражателя следует в соответствии со своим вариантом, представленным в таблице 1.1. Крайний правый столбец этой таблицы показывает, на какое число шагов (букв, *i*) перемещается соответствующий ротор при зашифровании одного (текущего) символа; число 0 означает перемещение соответствующего ротора на один шаг при условии, что рас положенный правее ротор совершит один оборот.

Таблица 1.1 – Варианты задания

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Вариант задания | *L* | *M* | *R* | *Re* | *Li*-*Mi****-****Ri* |
| 1 | I | II | III | B | 0-2-2 |
| 2 | II | III | V | C | 1-2-2 |
| 3 | III | VII | I | B Dunn | 1-0-1 |
| 4 | IV | III | II | C Dunn | 0-0-4 |
| 5 | I | Beta | Gamma | B | 3-1-3 |
| 6 | II | Gamma | IV | C | 1-1-1 |
| 7 | Beta | Gamma | V | B Dunn | 0-2-2 |
| 8 | V | VI | VII | C Dunn | 1-2-2 |
| 9 | VIII | II | IV | B | 1-0-1 |
| 10 | Gamma | III | Beta | C | 0-0-4 |
| 11 | Beta | VIII | I | B Dunn | 3-1-3 |
| 12 | III | Gamma | V | C Dunn | 1-1-2 |
| 13 | VI | IV | II | B | 1-2-2 |
| 14 | II | Beta | VIII | C | 0-2-2 |
| 15 | VII | Gamma | II | B Dunn | 1-2-2 |

Метод rotorEncrypt принимает на вход шифруемый символ алфавита, алфавит из которого будет осуществляться преобразования, алфавит в который будет осуществляться преобразование (алфавит ротора) и текущее смещение ротора. Код метода представлен в листинге 1.1.

|  |
| --- |
| private rotorEncrypt = (letter: string, originalAlphabet: string, encryptionAlphabet: string, currentOffset: number): string => {  let originalIndex: number = originalAlphabet.indexOf(letter);    return encryptionAlphabet[(originalIndex + currentOffset) % this.ROTOR\_LENGTH];  } |

Листинг 1.1 – Код метода шифрования символа с помощью ротора

Далее были разработаны методы directPath и reversePath, которые представляют собой прямой и обратный пути преобразования соответственно. Код методов представлен в листинге 1.2.

|  |
| --- |
| private directPath = (letter: string, operation: Operation): string => {let afterRight; let afterMiddle          switch (operation) {              case Operation.ENCRYPT:                  afterRight = this.rotorEncrypt(letter, this.ALPHABET, this.R\_ROTOR, this.rCurrentPosition);                  afterMiddle = this.rotorEncrypt(afterRight, this.ALPHABET, this.M\_ROTOR, this.mCurrentPosition);                  return this.rotorEncrypt(afterMiddle, this.ALPHABET, this.L\_ROTOR, this.lCurrentPosition);              case Operation.DECRYPT:                  afterRight = this.rotorDecrypt(letter, this.ALPHABET, this.R\_ROTOR, this.rCurrentPosition);                  afterMiddle = this.rotorDecrypt(afterRight, this.ALPHABET, this.M\_ROTOR, this.mCurrentPosition);                  return this.rotorDecrypt(afterMiddle, this.ALPHABET, this.L\_ROTOR, this.lCurrentPosition);}}      private reversePath = (letter: string, operation: Operation): string => {let afterLeft; let afterMiddle          switch (operation) {            case Operation.ENCRYPT: afterLeft = this.rotorEncrypt(letter, this.L\_ROTOR, this.ALPHABET, this.lCurrentPosition);                  afterMiddle = this.rotorEncrypt(afterLeft, this.M\_ROTOR, this.ALPHABET, this.mCurrentPosition);                  return this.rotorEncrypt(afterMiddle, this.R\_ROTOR, this.ALPHABET, this.rCurrentPosition);              case Operation.DECRYPT:                  afterLeft = this.rotorDecrypt(letter, this.L\_ROTOR, this.ALPHABET, this.lCurrentPosition);                  afterMiddle = this.rotorDecrypt(afterLeft, this.M\_ROTOR, this.ALPHABET, this.mCurrentPosition);                  return this.rotorDecrypt(afterMiddle, this.R\_ROTOR, this.ALPHABET, this.rCurrentPosition);}} |

Листинг 1.2 – Код методов прямого и обратного пути зашифрования/расшифрования

Метод passThroughReflector принимает текущий символ и осуществляет подстановку на основе С Dunn рефлектора. Код метода представлен в листинге 1.3.

|  |
| --- |
| private REFLECTOR: { [key: string]: string } = {          a: 'r', b: 'd', c: 'o', d: 'b', e: 'j', f: 'n',          g: 't', h: 'k', i: 'v', j: 'e', k: 'h', l: 'm',          m: 'l', n: 'f', o: 'c', p: 'w', q: 'z', r: 'a',          s: 'x', t: 'g', u: 'y', v: 'i', w: 'p', x: 's',          y: 'u', z: 'q'      }  private passThroughReflector = (letter: string) => {          return this.REFLECTOR[letter];      } |

Листинг 1.3 – Код метода рефлектора

Метод shiftRotors выполняется после зашифрования/расшифрования одного символа и меняет положения каждого из роторов. Код метода представлен в листинге 1.4.

|  |
| --- |
| private shiftRotors = (): void => {          this.lCurrentPosition = (this.lCurrentPosition + this.L\_SHIFT) % this.ROTOR\_LENGTH;          this.mCurrentPosition = (this.mCurrentPosition + this.M\_SHIFT) % this.ROTOR\_LENGTH;          this.rCurrentPosition = (this.rCurrentPosition + this.R\_SHIFT) % this.ROTOR\_LENGTH;      } |

Листинг 1.4 – Код метода, смещающего роторы

Итоговый метод шифрования encrypt принимает на вход оригинальный текст и выполняет шифрование с помощью метода прямого пути, метода рефлектора и метода обратного пути. После каждой итерации осуществляется смещение роторов. Код метода представлен в листинге 1.5.

|  |
| --- |
| public encrypt = (text: string): string => {          let result: string = "";for (let s of text) {              if (this.ALPHABET.includes(s)) {  let afterDirect = this.directPath(s, Operation.ENCRYPT);  let afterReflector = this.passThroughReflector(afterDirect);                  result += this.reversePath(afterReflector, Operation.ENCRYPT); this.shiftRotors();}}          return result;} |

Листинг 1.5 – Код метода шифрования

Метод расшифрования decrypt принимает на вход зашифрованный текст. Он выполняет аналогичные методу encrypt операции, только на прямом и обратном пути осуществляет расшифрование. Код метода представлен в листинге 1.6.

|  |
| --- |
| public decrypt = (encryptedText: string): string => {          let result: string = "";          for (let s of encryptedText) {              if (this.ALPHABET.includes(s)) {                  let afterDirect = this.directPath(s, Operation.DECRYPT);                  let afterReflector = this.passThroughReflector(afterDirect);                  result += this.reversePath(afterReflector, Operation.DECRYPT);                  this.shiftRotors();}          }          return result;} |

Листинг 1.6 – Код метода расшифрования

Кроме того, класс Enigma позволяет устанавливать начальные положения роторов в своём конструкторе.

Пример зашифрования одного символа, если *L* – IV, *M* – III, *R* – II, *Re* – C Dunn, *Li*-*Mi*-*Ri* = 0-0-4:

1. Шифруемый символ: a, исходный алфавит: abcdefghijklmnopqrstuvwxyz;
2. Ротор *R* в положении 4: siruxblhwtmcqgznpyfvoeajdk. Положение символа a в исходном алфавите: 0. Производим замену a → s;
3. Ротор *M* в положении 0: bdfhjlcprtxvznyeiwgakmusqo. Положение символа s в исходном алфавите: 18. Производим замену s → g;
4. Ротор *L* в положении 0: esovpzjayquirhxlnftgkdcmwb. Положение символа g в исходном алфавите: 6. Производим замену g → j;
5. Далее символ j проходит через рефлектор. Производим замену: j → e;
6. Обратный ход через ротор *L*. Положение символа e в алфавите ротора *L*: 0. Производим замену e → a;
7. Обратный ход через ротор *M*. Положение символа a в алфавите ротора *M*: 19. Производим замену a → t;
8. Обратный ход через ротор *R*. Положение символа t в алфавите ротора *R*: 9. Производим последнюю замену t → j.
9. Далее роторы смещаются и начинается шифрование следующего символа.

Результат работы приложения для роторов в позиции 0-0-4 приведен на рисунке 1.3.

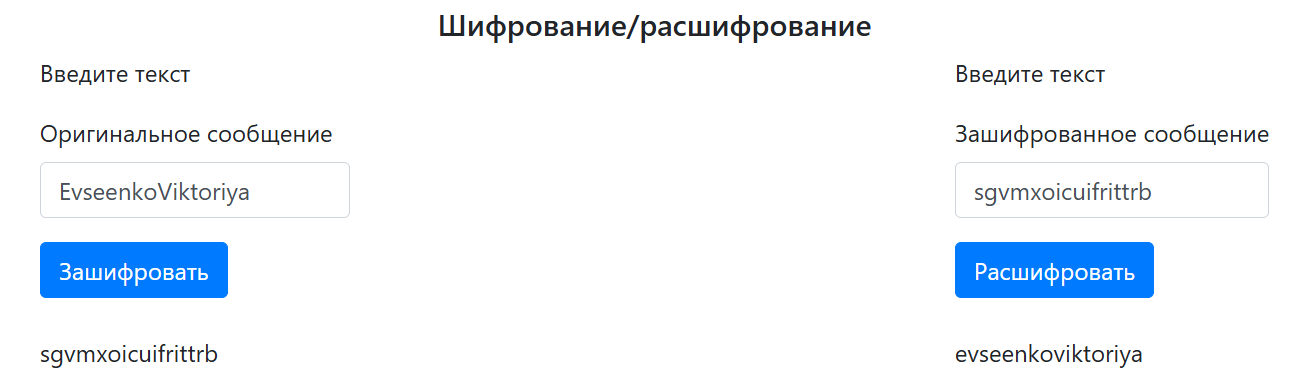


Рисунок 1.3 – Результат шифрования и дешифрования текста

# **2 Гистограммы**

Нужно было сформировать гистограммы частот появления символов для исходного и зашифрованного сообщений. Гистограмма частот появления символов исходного сообщения приведена на рисунке 2.1.



Рисунок 2.1 – Гистограмма частот появления символов исходного сообщения

Гистограмма частот появления символов зашифрованного сообщения приведена на рисунке 2.2.



Рисунок 2.2 – Гистограмма частот появления символов зашифрованного сообщения

Можно заметить, что частоты появления не совпадают, что значительно увеличивает криптостойкость шифра машины «Энигма».

# **3 Криптостойкость**

Рассмотрим пример для трехроторной «Энигмы» Вермахта с отражателем (по умолчанию – B, см. рис 1.2) и выбором из 5 роторов. Использовались 10 штекерных кабелей на коммутационной панели (количество кабелей, по умолчанию поставляемых с машиной). Чтобы выбрать 3 ротора из возможных 5, существует 60 комбинаций (5 · 4 · 3). Каждый ротор (его внутренняя проводка) может быть установлен в любом из 26 положений. Следовательно, с 3 роторами имеется 17 576 различных положений ротора (26 · 26 · 26). Кольцо на каждом роторе содержит маркировку ротора (что здесь неважно) и выемку, которая влияет на шаг перемещения расположенного левее ротора. Каждое кольцо может быть установлено в любом из 26 положений. Поскольку слева от третьего (наиболее левого) ротора нет ротора, на расчет влияют только кольца самого правого и среднего ротора. Это дает 676 комбинаций колец (26 · 26).

Коммутационная панель обеспечивает самый большой набор возможных настроек. Для первого кабеля одна сторона может иметь любое из 26 положений, а другая сторона – любое из 25 оставшихся положений (одна буква коммутируется с другой). Однако поскольку комбинация и ее обратная сторона идентичны (AB такая же, как BA), мы должны игнорировать все двойные числа во всех возможных комбинациях для одного кабеля, предоставляя (26 · 25) / (1! · 21), или 325 уникальных способов коммутаций одним кабелем. Для двух кабелей – (26 · 25) комбинаций для первого кабеля и, поскольку два разъема уже используются, то получается (24 · 23) комбинаций для второго кабеля. Следуя этой простой логике, получается (26 · 25 · 24 · 23) / (2! · 22) = = 44 850 уникальных способов коммутаций с использованием двух кабелей. Для трех кабелей – (26 · 25 · 24 · 23 · 22 · 21) /(3! × × 23) = 3 453 450 комбинаций и т. д. Таким образом, с использованием 10 кабелей на коммутационной панели получаются 150 738 274 937 250 различных комбинаций. Формула, где n равно количеству кабелей, равна 26! / (26 – 2n)! · n! · 2n. Численно это дает: 60 · 17 576 · 676 · 150 738 · 274 · 937 · 250 = = 107 458 687 327 250 619 360 000, или 1,07 · 1023.

Таким образом, практически рассматриваемая версия «Энигмы» (3 ротора с выбором из 5 роторов, отражатель В и 10 штекерных ка белей для коммутационной панели) может быть настроена на 1,07 · 1023 различных состояний, что сопоставимо с 77-битным криптографическим ключом.

# **Вывод**

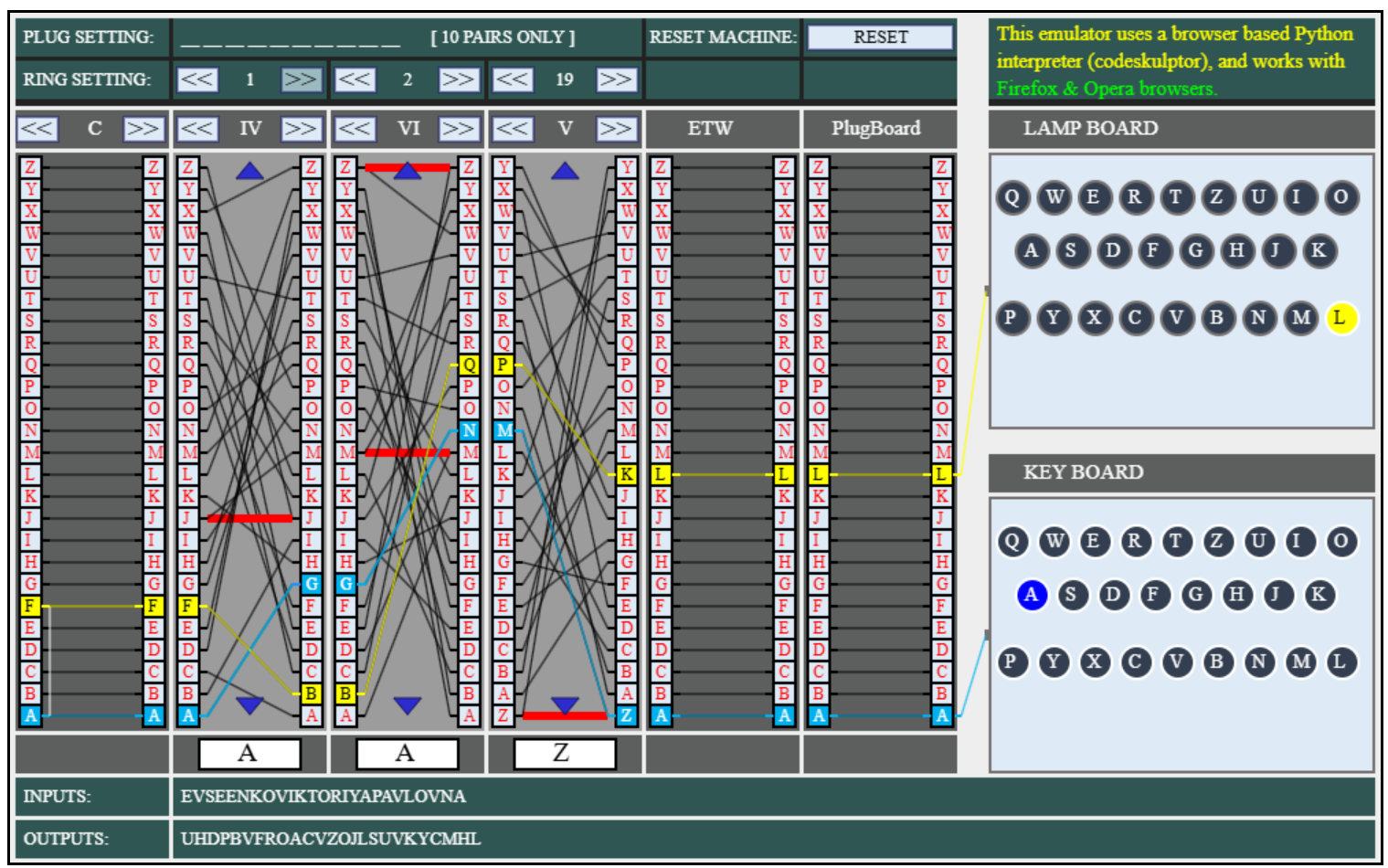
В ходе выполнения лабораторной работы были достигнуты поставленные цели и решены задачи, связанные с изучением и практическим освоением методов реализации перестановочно-подстановочных шифров на примере машины «Энигма». Мы закрепили теоретические знания по алгебраическому описанию криптографических операций, что позволило лучше понять принцип работы сложных схем шифрования и их криптостойкость. Практическая часть, выполненная на симуляторах «Энигмы», дала возможность не только реализовать процедуры зашифрования и расшифрования сообщений, но и оценить устойчивость подобных систем к криптоаналитическим атакам.

Полученные результаты продемонстрировали, что несмотря на кажущуюся сложность алгоритмов шифрования «Энигмы», они основаны на систематическом механизме подстановки и перестановки, что делает их одновременно гибкими и уязвимыми при наличии определённых ошибок в настройках и эксплуатации. Освоение работы с симуляторами позволило приобрести практические навыки, необходимые для дальнейшего изучения и разработки криптографических приложений.

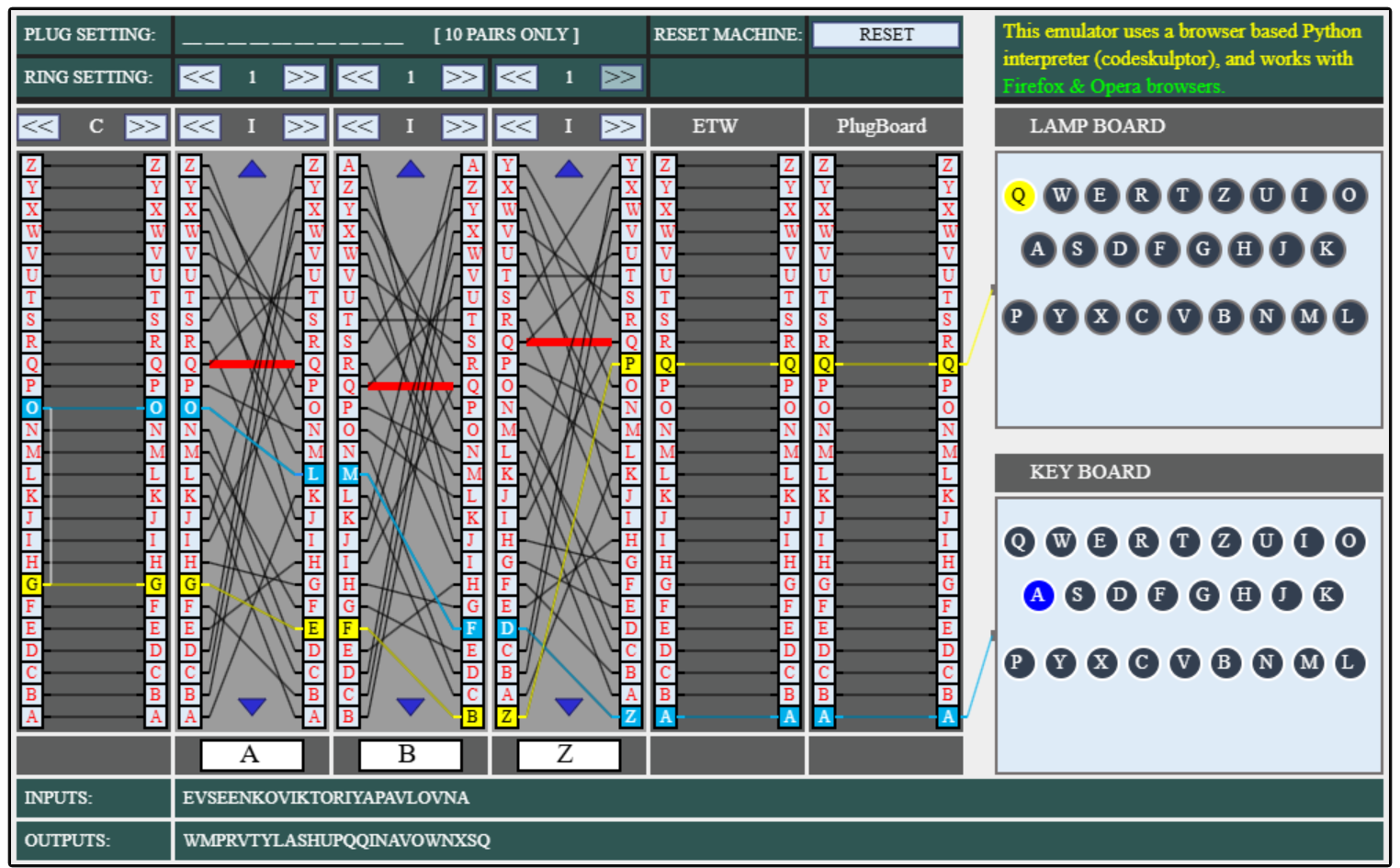
Таким образом, проведённая работа способствовала как углублению теоретических знаний, так и развитию практических навыков в области криптографии, что является важным этапом в изучении информационной безопасности и разработки современных систем защиты данных.

# **Приложение А**

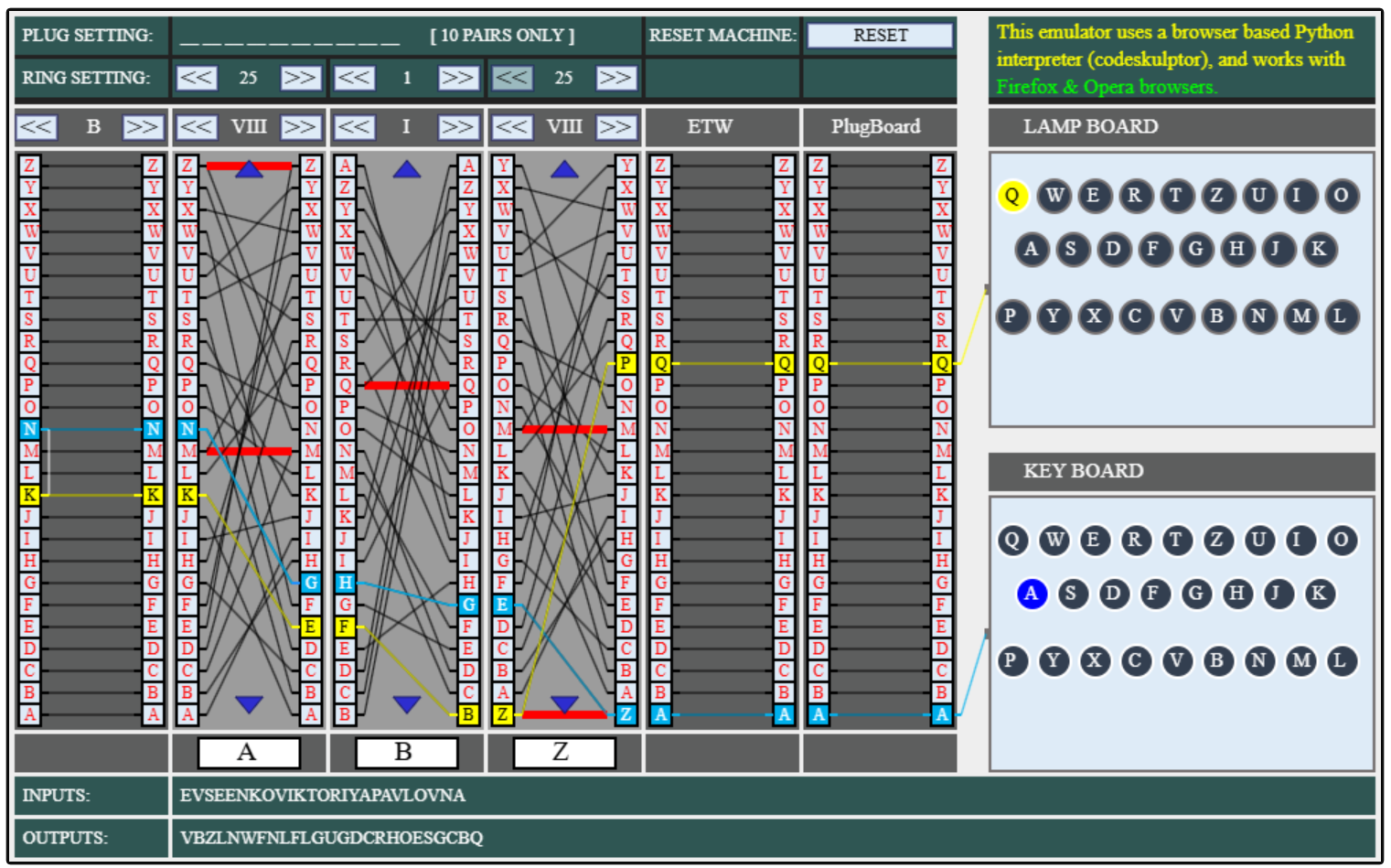
1. Результат зашифрования первого варианта настройки машины



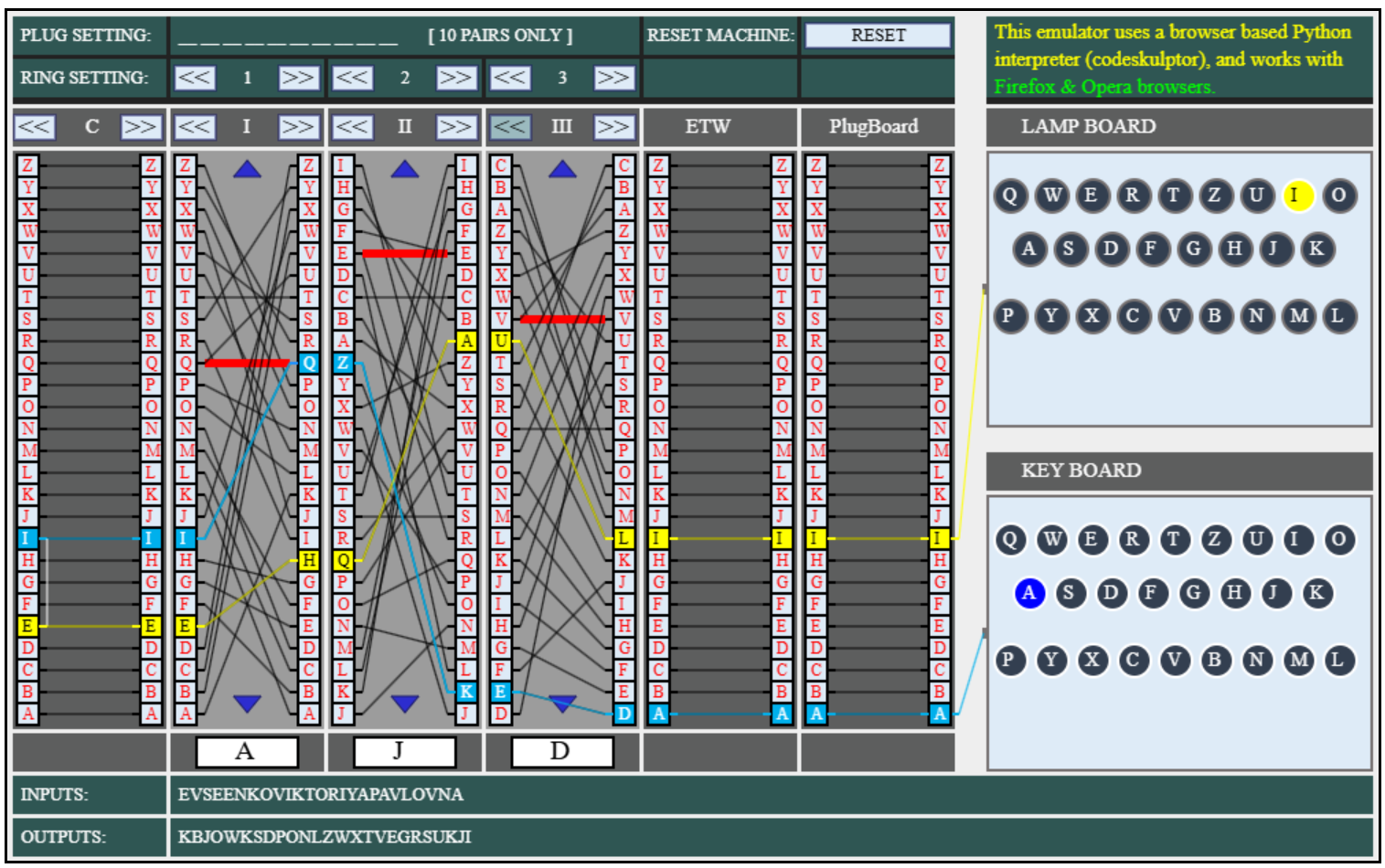
1. Результат зашифрования второго варианта настройки машины



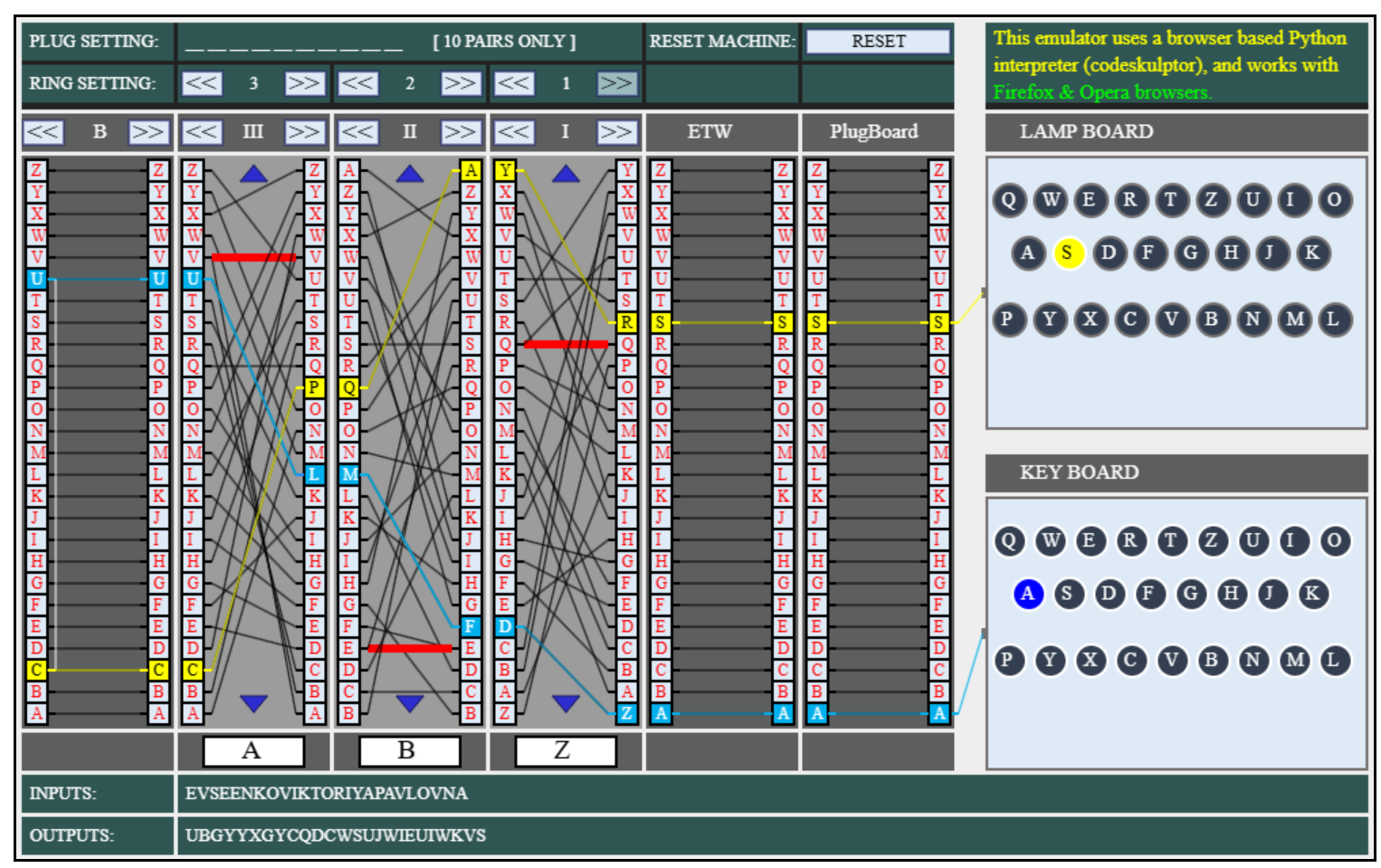
1. Результат зашифрования третьего варианта настройки машины



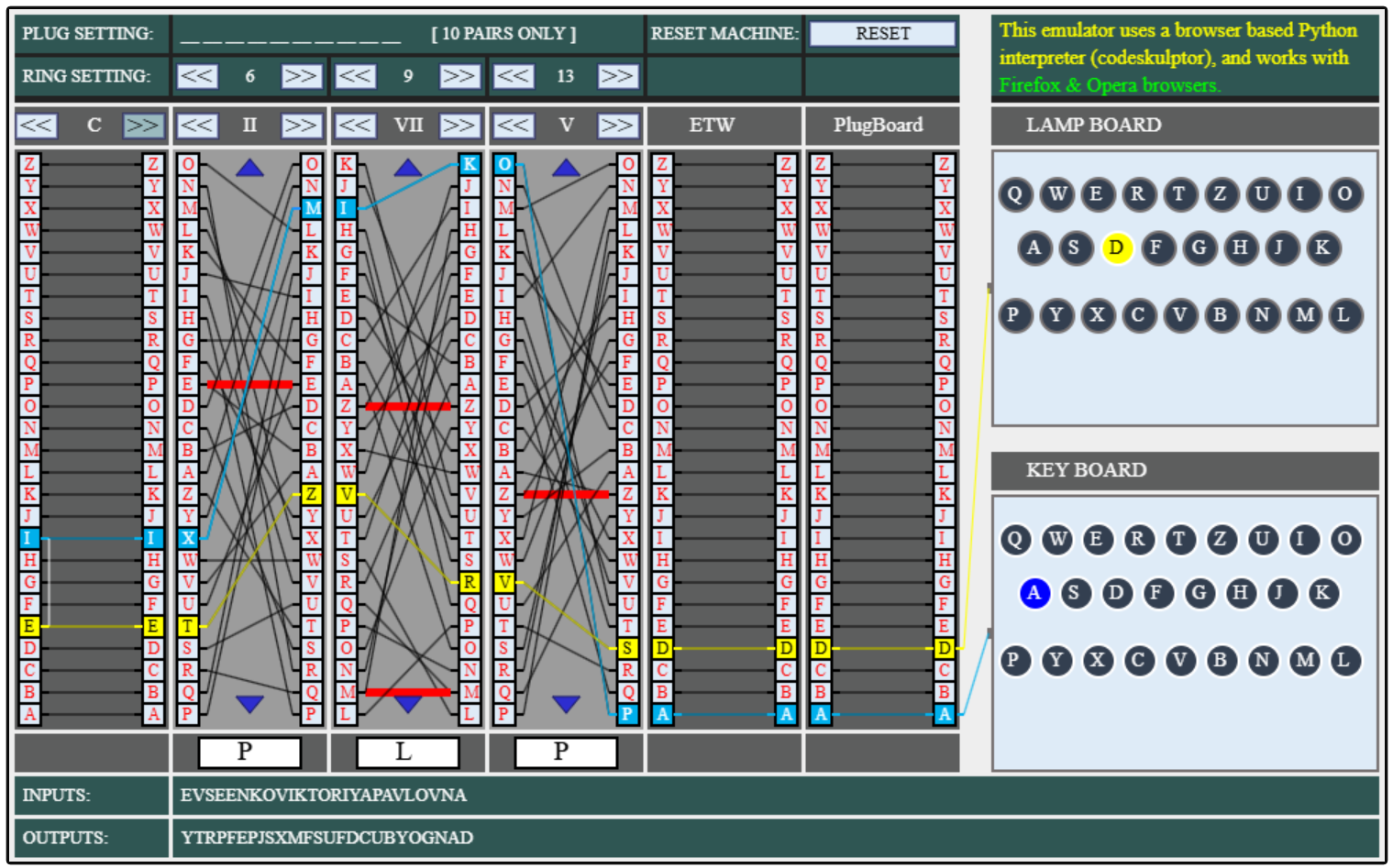
1. Результат зашифрования четвертого варианта настройки машины



1. Результат зашифрования пятого варианта настройки машины



1. Результат зашифрования шестого варианта настройки машины



1. Результат зашифрования седьмого варианта настройки машины



1. Результат зашифрования восьмого варианта настройки машины

