МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«МИРЭА — Российский технологический университет»

РТУ МИРЭА

Институт кибербезопасности и цифровых технологий

(наименование института, филиала)

Кафедра КБ-14 «Цифровые технологии обработки данных»

(наименование кафедры)

Практическая работа

по дисциплине Криптографическая защита информации

(наименование дисциплины)

Выполнил:

БСБО-05-20

Верхотуров В. С.

Москва — 2024 г.

СОДЕРЖАНИЕ

1 ШИФР МНОГОАЛФАВИТНОЙ ЗАМЕНЫ ВИЖИНЕРА

1.1 Задание

Для повышения стойкости шифра используют многоалфавитные замены, в которых для замены символов исходного текста используются символы нескольких алфавитов.

Одной из разновидностей такого метода является схема шифрования Вижинера. Шифр очень устойчивый к вскрытию. Таблица Вижинера представляет собой квадратную матрицу с n2 элементами, где n – число символов используемого алфавита. Каждая строка получена циклическим сдвигом алфавита на один символ

При шифровании сообщения его выписывают в строку, а под ним буквенный ключ. Если ключ оказался короче сообщения, то его циклически повторяют. Шифровку получают, находя символ в колонке таблицы по букве текста и строке, соответствующей букве ключа.

Например:

Сообщение П Р И Е З Ж А Ю Ш Е С Т О Г О

Ключ А Г А В А А Г А В А А Г А В А

Шифровка П Н И Г З Ж Ю Ю ЮА Е О Т М

Предыдущие шифры называются монограммными, так как шифрование ведется по одной букве. Шифрование по 2 букве называются биграммными.

1.2 Выполнение практической

Результат практической: `https://crypto-tasks.vercel.app/task1`.

Репозиторий `https://github.com/ValeryVerkhoturov/crypto-tasks`.

Листинг 1 – Шифр многоалфавитной замены Вижинера

```
1 export class VigenereCipher {
2         private alphabet: string = АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ'';
3         private mod: number = this.alphabet.length;
4
5         encrypt(text: string, key: string): string {
6                 return this.processText(text, key, 'encrypt');
7         }
```

```
8
9           decrypt(text: string, key: string): string {
10                  return this.processText(text, key, 'decrypt');
11          }
12
13          private processText(text: string, key: string, mode: 'encrypt' | '
                decrypt'): string {
14                  let processedText = '';
15                  let keyIndex = 0;
16
17                  text = text.toUpperCase();
18                  key = key.toUpperCase();
19
20                  for (let i = 0; i < text.length; i++) {
21                          const char = text[i]
22                          if (this.alphabet.includes(char)) {
23                                  const charIndex = this.alphabet.indexOf(char);
24                                  const keyChar = key[keyIndex % key.length];
25                                  const keyCharIndex = this.alphabet.indexOf(
                                        keyChar);
26
27                                  if (mode === 'encrypt') {
28                                          processedText += this.alphabet[(
                                                charIndex + keyCharIndex) % this.
                                                mod];
29                                  } else {
30                                          let decodeIndex = (charIndex -
                                                keyCharIndex) % this.mod;
31                                          if (decodeIndex < 0) {
32                                                  decodeIndex += this.mod;
33                                          }
34                                          processedText += this.alphabet[
                                                decodeIndex];
35                                  }
36
37                                  keyIndex++;
38                          } else {
39                                  processedText += char;
40                          }
41                  }
42
43                  return processedText;
44          }
45  }
```

2 МАГИЧЕСКИЙ КВАДРАТ

2.1 Задание

Магический квадрат - это квадратная таблица с вписанными в клетки последовательными натуральными числами от 1, которые дают в сумме по каждому столбцу, каждой строке и каждой диагонали одно и тоже число.

Чтобы зашифровать открытый текст с помощью такого квадрата, нужно пронумеровать все буквы в фразе по порядку и вставить их в квадрат вместо соответствующих цифр.

2.2 Выполнение практической

Результат практической: `https://crypto-tasks.vercel.app/task2`.

Репозиторий `https://github.com/ValeryVerkhoturov/crypto-tasks`.

Листинг 2 – Шифр Магический квадрат

```
export class MagicSquareCipher {
        private magicSquare: number[][]
        private magicSquareDimensions: number

        constructor(magicSquare: number[][]) {
                this.magicSquare = magicSquare;
                this.magicSquareDimensions = magicSquare.length;
        }

        encrypt(text: string): string {
                let encryptedText = Array(text.length).fill(null);

                for (let i = 0; i < text.length; i++) {
                        const row = Math.floor(i / this.magicSquareDimensions)
                            ;
                        const col = i % this.magicSquareDimensions;
                        const newPos = this.magicSquare[row][col];
                        console.log(this.magicSquare, newPos)
                        if (newPos < text.length) {
                                encryptedText[newPos] = text[i];
                        }
                }

                return encryptedText.join("");
        }

```

```
26          decrypt(encryptedText: string): string {
27                  let decryptedText = Array(encryptedText.length).fill(null);
28
29                  for (let i = 0; i < encryptedText.length; i++) {
30                          const row = Math.floor(i / this.magicSquareDimensions)
                                ;
31                          const col = i % this.magicSquareDimensions;
32                          const originalPos = this.magicSquare[row][col];
33                          if (originalPos < encryptedText.length) {
34                                  decryptedText[i] = encryptedText[originalPos];
35                          }
36                  }
37
38                  return decryptedText.join("");
39          }
40  }
```

ЗАКЛЮЧЕНИЕ