# Элементы криптографии. Шифрование (кодирование) различных исходных текстов одним ключом

Косолапов Степан [1]

28 октября, 2023, Москва, Россия

[1] Российский Университет Дружбы Народов

# Цели и задачи работы

# Цель лабораторной работы

Освоить на практике применение режима однократного гаммирования на примере кодирования различных исходных текстов одним ключом.

# Процесс выполнения лабораторной работы

# Вспомогательные функции

```
const stringToHex = (str) => {
    return str.split('').map(c => c.charCodeAt(0).toString(16)).join(' ');
}

const hexToString = (hexStr) => {
    return hexStr.split(' ').map(c => String.fromCharCode(Number.parseInt(c, 16)
}
```

# Функция для генерации ключа

```
const generateKey = (length) => {
    const result = [];

    for (let i = 0; i < length; i++) {
        const asciiCode = Math.floor(Math.random() * 1048);
        result.push(asciiCode.toString(16));
    }

    return result.join(' ');
}
```

# Функция шифрования

```
const gammingCipher = (hexText, hexKey) => {
    const textSplit = hexText.split(' ');
    const keySplit = hexKey.split(' ');

    if (textSplit.length !== keySplit.length) {
        throw new Error('Key and message must have equal lengths.');
    }

    return textSplit.map((textCharHex, i) => {
        const keyCharHex = keySplit[i];

        const xorResult = Number.parseInt(textCharHex, 16) ^ Number.parseInt(ke

        return xorResult.toString(16);
    }).join(' ')
```

# Генерация ключа

```
const initialTextLength = 'Штирлиц – Вы Герой!!!!'.length;
console.log('initialTextLength:', initialTextLength);
console.log('generatedKey:', generateKey(initialTextLength));
```

```
node index.js
```

```
initialTextLength: 22
generatedKey: 203 3e7 2ea ec 2dc 29 3b4 10b 7f 23 33b 185 1ac 121 26f 97 1d5
```

# Шифрование сообщений

```
const message = 'Штирлиц – Вы Герой!!!!';
const message2 = 'Привет, Штирлиц, ура!!';

const hexMessage = stringToHex(message);
const hexMessage2 = stringToHex(message2);

const hexKey = '203 3e7 2ea ec 2dc 29 3b4 10b 7f 23 33b 185 1ac 121 26f 97 1c

const enc1 = gammingCipher(hexMessage, hexKey);
const enc2 = gammingCipher(hexMessage2, hexKey);
```

# Известные данные

```
let known1 = 'Штирлиц**********!!!!';
let hexKnown1 = stringToHex(known1);

let known2 = '******, Штирлиц, *****';
let hexKnown2 = stringToHex(known2);
```

# Разгадывание через перекрытие шаг 1

```
console.log('known1 xor enc1 xor enc2', hexToString(gammingCipher(enc2, gam
known2 = 'Привет, Штирлиц, *ра!!';
console.log('known2 xor enc2 xor enc1', hexToString(gammingCipher(enc1, gam
known1 = 'Штирлиц – Вы Герой!!!!';
console.log('known1 xor enc1 xor enc2', hexToString(gammingCipher(enc2, gam
known1 = 'Штирлиц – Вы Герой!!!!';
known2 = 'Привет, Штирлиц, ура!!';
```

# Вывод программы

node index.js

known1 Штирлиц**********!!!!
known2 ******, Штирлиц, *****

known1 xor enc1 xor enc2 Привет,*□ш!бYцдPpa!!
known2: Привет, Штирлиц, *ра!!

known2 xor enc2 xor enc1 Штирлиц – Вы ГероР!!!!
known1: Штирлиц – Вы Герой!!!!

known1 xor enc1 xor enc2 Привет, Штирлиц, ура!!
known1: Штирлиц – Вы Герой!!!!
known2: Привет, Штирлиц, ура!!

# Выводы по проделанной работе

# Вывод

В данной работе мы освоили на практике применение режима однократного гаммирования на примере кодирования различных исходных текстов одним ключом.