

# **Отчет по лабораторной работе №3**

**Шифрование гаммированием**

Легиньких Галина Андреевна

# Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Вывод	8

## Список иллюстраций

2.1	Ключ . . . . .	6
2.2	Шифрование . . . . .	6
2.3	Дешифрование . . . . .	6
2.4	Фрагмент . . . . .	7
2.5	Вызов функций . . . . .	7
2.6	Вывод . . . . .	7

## Список таблиц

# **1 Цель работы**

Освоить на практике применение режима однократного гаммирования.

## 2 Выполнение лабораторной работы

1. Написала функцию для генерации рандомного ключа состоит из случайно последовательности символов. (рис. 2.1)

```
import random

def generate_key(word):
    key = ""
    for _ in range(len(word)):
        key += random.choice("0123456789abcdef") # Шестнадцатеричная система
    return key
```

Рис. 2.1: Ключ

2. Функция шифрования. В основе используется XOR. (рис. 2.2)

```
def encrypt(plaintext, key):
    ciphertext = ""
    for i in range(len(plaintext)):
        char = plaintext[i]
        key_char = key[i%len(key)]
        encrypted_char = chr(ord(char) ^ ord(key_char)) # XOR операция
        ciphertext += encrypted_char
    return ciphertext
```

Рис. 2.2: Шифрование

3. Аналогичный принцип для дешифрования. (рис. 2.3)

```
def decrypt(ciphertext, key):
    decryptedtext = ""
    for i in range(len(ciphertext)):
        char = ciphertext[i]
        key_char = key[i%len(key)]
        decrypted_char = chr(ord(char) ^ ord(key_char)) # XOR операция
        decryptedtext += decrypted_char
    return decryptedtext
```

Рис. 2.3: Дешифрование

4. Функция нахождения ключей для фрагмента. (рис. 2.4)

```
def find_possible_key(chiphertext, fragment):
    possible_keys = []
    for i in range(len(chiphertext) - len(fragment)+1):
        possible_key = ""
        for j in range(len(fragment)):
            char = ciphertext[i+j]
            fragment_char = fragment[j]
            key_char = chr(ord(char) ^ ord(fragment_char))
            possible_key += key_char
        possible_keys.append(possible_key)
    return possible_keys
```

Рис. 2.4: Фрагмент

5. Основной кусок кода, где задается строка и вызов всех функций. (рис. 2.5)

```
text = "С Новым Годом, друзья!"
key = generate_key(text)
encrypted_text = encrypt(text, key)
decrypted_text = decrypt(encrypted_text, key)
fragment = "С Новым"
possible_keys = find_possible_key(encrypted_text, fragment)
print("Ключ -", key)
print("Зашифрованный текст -", encrypted_text)
print("Дешифрованный текст -", decrypted_text)
print("Возможные ключи -", possible_keys)
```

Рис. 2.5: Вызов функций

6. После запуска программы мы получим следующее. (рис. 2.6)

```
Ключ - 336033fa5298c0bccdd6d1
Зашифрованный текст - Bn0E9uA90nпLBT0r0Fm-
Дешифрованный текст - С Новым Годом, друзья!
Возможные ключи - [' 336033f', ' n\x13?'\x110', ' \n0\x1cFm\x1a', ' /Codo00', ' jG@\x1461', ' YQ\x183F:', ' {a;2?Kc', ' 0I\x1134\x14P', ' \x07b\x108m10', ' -Э\x1bamm', ' ,\x1W\x1c\x1f', ' "wE0eh0", '~0i\x1180', ' n0J\x1dA\x18F', ' 8P>0e1\x17', ' vfm0f'b']
```

Рис. 2.6: Вывод

## **3 Вывод**

Освоила на практике применение режима однократного гаммирования.