

Отчёт по лабораторной работе №7

Режим Однократного Гарммирования

Вакутайпа Милдред

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Выводы	9
	Список литературы	10

Список иллюстраций

3.1	Функция для генерации ключа	7
3.2	Функция для шифрования и дешифрования	7
3.3	функция для нахождения возможных ключей	7
3.4	Проверка	8

Список таблиц

1 Цель работы

Научиться применять режим однократного гарммирования.

2 Задание

Подобрать ключ, чтобы получить сообщение “С Новым Годом”. Требуется разработать приложение, позволяющее шифровать и дешифровать данные в режиме однократного гаммирования.

3 Выполнение лабораторной работы

На языке программирования python, создала функцию для генерации случайного ключа.

```
import string
import random

def generate_key (text):
    return ''.join(random.choice(string.ascii_letters + string.digits) for _ in text)
```

Рис. 3.1: Функция для генерации ключа

Делала одну функцию для шифрования и дешифрования текста.

```
def crypt (text, key):
    return ''.join(chr(ord(c)^ord(k)) for c, k in zip(text, key))
```

Рис. 3.2: Функция для шифрования и дешифрования

Нужно определить ключ, с помощью которого шифротекст может быть преобразован в некоторый фрагмент текста. Для этого создала функцию для нахождения возможных ключей для фрагмента текста.

```
def find_keys(cipher, fragment):
    return [''.join(chr(ord(cipher[i+j])^ord(f)))
            for i in range(len(cipher)-len(fragment)+1)
            for j, f in enumerate(fragment)]
```

Рис. 3.3: функция для нахождения возможных ключей

Далее проверила работы программы. Шифрование и дешифрование происходит верно, как и нахождение ключей, с помощью которых можно расшифровать верно кусок текста.

```

bland = 'C Hоmеn fоdоm'
key = generate_key(bland)
ciphered = crypt(bland, key)
decrypted = crypt(ciphered, key)
fragment = 'C Hоmеn'

possible_keys = find_keys(ciphered, fragment)

print(f"orig: {bland}, \nkey: {key}, \n ciphered: {ciphered}, decrypt: {decrypted}")
print(f"possible keys: {possible_keys}, decrypt: {crypt(ciphered, possible_keys[0])}")

orig: C Hоmеn fоdоm, key: Y65x1PofuajDr, ciphered: H0uayHfFwajOe, decrypt: C Hоmеn fоdоm
possible keys: ['\n', '6', '5', 'x', 'l', 'p', 'o', 'a', 'j', '[', 'i', ')', '\x18', '0', '\t', '8', 'C', 'X', 'a', 'й', 'z', 'e', '0', '\x06', 'm', 'V',
'n', 'c', '\x2f', 'n', 'N', 'h', 't', '\x14', 'b', ':', 'o', 'h', 'x', 'm', '\x15', 'f', 'r', 'f', '[', 'a', 'l', 'l', 'r'], decrypt: C

```

Рис. 3.4: Проверка

4 Выводы

При выполнении данной работы я научилась применять режим однократного гарммирования.

Список литературы