```
**Презентация лабораторной работы 7**
```

ТЕМА **«** Элементы криптографии. Однократное гаммирование »

Выполнил:
Студент группы НПИбд-02-21
Студенческий билет № 1032205641
Сатлихана Петрити

Введение

Освоить на практике применение режима однократного гаммирования

Последовательность выполнения работы

Нужно подобрать ключ, чтобы получить сообщение «С Новым Годом, друзья!». Требуется разработать приложение, позволяющее шифровать и дешифровать данные в режиме однократного гаммирования.

Приложение должно:

1. Определить вид шифротекста при известном ключе и известном открытом тексте.

```
[1] import re
```

```
[24] russian_alphabet = [
        'А', 'Б', 'В', 'Г', 'Д', 'Е', 'Ё', 'Ж', 'З', 'И', 'Й', 'К', 'Л', 'М', 'Н', 'О', 'П', 'Р', 'С', 'Т',
        'У', 'Ф', 'Х', 'Ц', 'Ч', 'Ш', 'Щ', 'Ъ', 'Ы', 'Ь', 'Э', 'Ю', 'Я', ' ', '!', '?', '.', ',', '«', '»'
    ]
```

```
def decrypt(ciphertext, gama):
    textLen = len(ciphertext)
    gamaLen = len(gama)

    keytext = []
    for i in range(textLen):
        keytext.append(gama[i % gamaLen])

    decrypted_text = []
    for i in range(textLen):
        if ciphertext[i] in russian_alphabet:
            decrypted_index = (russian_alphabet.index(ciphertext[i]) - russian_alphabet.index(keytext[i])) % len(russian_alphabet)
            decrypted_text.append(russian_alphabet[decrypted_index])
        else:
            decrypted_text.append(ciphertext[i])

    return ''.join(decrypted_text)
```

```
[38] encrypted_message = encrypt('С Новым Годом, друзья!', 'ААЪАЙАААААААААААААААА')
     decrypted_message = decrypt(encrypted_message,'ААЪАЙАААААААААААААААА' )
```

```
[39] print(encrypted_message)
     print(decrypted_message)

С Бовым Годом, друзья!
С Новым Годом, друзья!
```

рис. 1 Определение вид шифротекста

2. Определить ключ, с помощью которого шифротекст может быть преобразован в некоторый фрагмент текста, представляющий собой один из возможных вариантов прочтения открытого текста .

```
def derive_key(ciphertext, plaintext):
    if len(ciphertext) != len(plaintext):
        raise ValueError("It is not in the same length")

    key = []
    for i in range(len(ciphertext)):
        if ciphertext[i] in russian_alphabet and plaintext[i] in russian_alphabet:
            key_index = (russian_alphabet.index(ciphertext[i]) - russian_alphabet.index(plaintext[i])) % len(russian_alphabet)
            key.append(russian_alphabet[key_index])
        else:
            key.append(ciphertext[i])

    return ''.join(key)

ciphertext = "С Бовым Годом, друзья!"
plaintext_fragment = "С Новым Годом, друзья!"
```

```
[43] derived_key = derive_key(ciphertext, plaintext_fragment)
     print(derived_key)

ААЪовымААодомААдрузьяА
```

рис. 2 Определение ключа

Вывод

В результате выполнения работы я освоила на практике применение режима однократного гаммирования