Лабораторная работа №2

Шифры перестановки

Доборщук Владимир Владимирович, НФИмд-02-22

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

Цель данной работы — изучить и программно реализовать шифры перестановки.

2 Задание

Заданием является:

- Реализовать все описанные в лабораторной работе шифры.

3 Теоретическое введение

Шифры перестановки преобразуют открытый текст в криптограмму путем перестановки его символов.

4 Выполнение лабораторной работы

Для реализации шифров мы будем использовать Python, так как его синтаксис позволяет быстро реализовать необходимые нам алгоритмы.

Использовали библиотеки, представленные на рисунке 4.1.

```
In [1]: import numpy as np

In [2]: def get_alphabet(option="english"):
            if option == "english":
                return list(map(chr, range(ord("a"), ord("z")+1)))
            elif option == "russian":
                return list(map(chr, range(ord("а"), ord("я")+1)))
```

Рис. 4.1: Библиотеки и дополнительные функции

Также реализовали функции получения алфавитов (английского и русского).

4.1 Маршрутное шифрование

Маршрутное шифрование реализовали в соответствии с описанной в лабораторной работе процедурой. Успешно протестировали на приведенном в работе отрывке. Результаты и программный код представлены на рисунке 4.2.

In [3]:
```python
def marchroute_cipher(message: str, key: str):
    alphabet_russian = get_alphabet("russian")
    alphabet_english = get_alphabet()
    columns_size = len(key)

    message_cleared = list(filter(lambda s: s.lower() in alphabet_russian or s in alphabet_english, message))

    message_matrix = [
        [letter for letter in message_cleared[i:i+columns_size]]
        for i in range(0, len(message_cleared), columns_size)
    ]

    if len(message_matrix[-1]) < columns_size:
        message_matrix[-1] = message_matrix[-1] +
            [message_matrix[-1][-1]]*(columns_size-len(message_matrix[-1]))

    message_password_dict = { value : np.array(message_matrix)[:,k] for k, value in enumerate(list(key)) }

    ciphered_message = ''.join(['']).join(message_password_dict[k]).upper()
                                for k in sorted(message_password_dict.keys())])

    return ciphered_message
```

In [4]:
```python
m_test = "нельзя недооценивать противника"
k_test = "пароль"
```

In [5]:
```python
result = marchroute_cipher(m_test, k_test)
print(f'Результат шифрования: \
        \n{m_test} * [{k_test}]\n-> {result, len(result)}')
```

```
Результат шифрования:
нельзя недооценивать противника * [пароль]
-> ('ЕЕНПНЗОАТАЬОВОКННЕЬВЛДИРИЯЦТИА', 30)
```

Рис. 4.2: Маршрутное шифрование и его тестирование

4.2 Шифрование с помощью решеток

Данный вид шифрования не удалось реализовать.

4.3 Таблица Виженера

Маршрутное шифрование реализовали в соответствии с описанной в лабораторной работе процедурой. Успешно протестировали на приведенном в работе отрывке (с учетом, что русский алфавит немного изменен). Результаты и программный код представлены на рисунке 4.3.

In [9]:
```python
def vigenere_table(message: str, key: str, differ_alphabet=False):
    alphabet_russian = get_alphabet("russian")
    if differ_alphabet:
        alphabet_russian.remove('ь')
        alphabet_russian[alphabet_russian.index('ъ')] = 'ь'
    alphabet_english = get_alphabet()

    def find_letter_for_pair(letters_pair: tuple):
        if letters_pair[0].lower() in alphabet_russian:
            orig_letter_index = alphabet_russian.index(letters_pair[1].lower())
            key_letter_index = alphabet_russian.index(letters_pair[0].lower())

            shift = orig_letter_index + key_letter_index

            if shift > len(alphabet_russian):
                return alphabet_russian[shift - len(alphabet_russian)]

            return alphabet_russian[shift]

    message_cleared = list(filter(lambda s: s.lower() in alphabet_russian or s in alphabet_english, message))
    row_length = len(message_cleared)
    full_key = (list(key) * row_length)[:row_length]

    message_key_zip = list(zip(full_key, message_cleared))

    return ''.join(list(map(find_letter_for_pair, message_key_zip))).upper()
```

In [10]:
```python
m_test = "криптография - серьезная наука"
k_test = "математика"
```

In [11]:
```python
result = vigenere_table(m_test, k_test, True)

print(f'Результат шифрования: \
       \n({m_test}) * [{k_test}]\n-> {result, len(result)}')
```

```
Результат шифрования:
(криптография - серьезная наука) * [математика]
-> ('ЦРЬФЯОХШКФФЯДКЭЬЧПЧАЛНТШЦА', 26)
```

Рис. 4.3: Таблица Виженера и ее тестирование

5 Выводы

В рамках выполненной лабораторной работы мы изучили и реализовали следующие шифры перестановки: маршрутное шифрование и таблицу Виженера. Реализовать шифрование с помощью решеток не удалось.

Список литературы