Лабораторная работа 2

Попов Дмитрий Павлович, НФИмд-01-23

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра математического моделирования и искусственного интеллекта

ПРЕЗЕНТАЦИЯ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2

дисциплина: Математические основы защиты информации и информационной безопасности

Преподователь: Кулябов Дмитрий Сергеевич

Студент: Попов Дмитрий Павлович

Группа: НФИмд-01-23

МОСКВА

2023 г.

Прагматика выполнения лабораторной работы

Прагматика выполнения лабораторной работы

- Требуется реализовать:

1. Маршрутное шифрование.
2. Шифрование с помощью решеток.
3. Таблица Виженера

Цель работы

Цель работы

Освоить на практике шифры перестановки.

Выполнение лабораторной работы

1. Реализовал программу для маршрутного шифрования (1/2)

1. Реализовал программу для маршрутного шифрования (1/2)

lab_2_marshrut.py

```
print("------------------------")
print(" Маршрутное шифрование")
print("------------------------")

alphabet = "а б в г д е 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"
alphabet = alphabet.split()
mess = str(input("Введите предложение, которое нужно зашифровать: ")).lower()
# mess = 'нельзя недооценивать противника'
mess = mess.replace(' ', '')
mess = mess.replace('\t', '')
orig_mess = mess
mess = list(mess)
# n_block = int(input("Введите длину блока N: "))
# n_block = 4
password = str(input("Введите пароль: ")).lower()
# password = 'пароль'
password = password.replace(' ', '')
password = password.replace('\t', '')

n_block = len(password)

print("\nВведенное сообщение без пробелов: ", orig_mess)
print("Длина блока N:                    ", n_block)
print("Введенный пароль без пробелов:    ", password)

# дополняем строку буквами "а" до кратного числу N
while len(mess) % n_block != 0:
    mess.append('а')
```

Figure 1: route1

1. Реализовал программу для маршрутного шифрования (2/2)

1. Реализовал программу для маршрутного шифрования (2/2)

```
# делим строку на блоки длины N
table = list()
tmp_lst = list()
for i in range(0, len(mess)):
    if i % n_block == 0 and i != 0:
        table.append(tmp_lst)
        tmp_lst = list()
    tmp_lst.append(mess[i])
    if i == len(mess) - 1:
        table.append(tmp_lst)

print("\nТаблица")
for t in table:
    print(t)

# сортируем пароль и формируем шифруем сообщение согласно столбцам таблицы по алфавиту пароля
sort_password = sorted(password)
print("_____" * n_block)
print(list(password))
cypher_mess = str()
for i in sort_password:
    col = password.index(i)
    for row in range(0, len(table)):
        cypher_mess += table[row][col].upper()

print("\nВведенное сообщение без пробелов: ", orig_mess)
print("Зашифрованное сообщение:          ", cypher_mess)
```

Figure 2: route2

2. Вывод работы первой программы

2. Вывод работы первой программы

```
-----------------------
 Маршрутное шифрование
-----------------------
Введите предложение, которое нужно зашифровать: нельзя недооценивать противника
Введите пароль: пароль

Введенное сообщение без пробелов:  нельзянедооцениватьпротивника
Длина блока N:                     6
Введенный пароль без пробелов:     пароль

Таблица
['н', 'е', 'л', 'ь', 'з', 'я']
['н', 'е', 'д', 'о', 'о', 'ц']
['е', 'н', 'и', 'в', 'а', 'т']
['ь', 'п', 'р', 'о', 'т', 'и']
['в', 'н', 'и', 'к', 'а', 'а']
------------------------------
['п', 'а', 'р', 'о', 'л', 'ь']

Введенное сообщение без пробелов:  нельзянедооцениватьпротивника
Зашифрованное сообщение:           ЕЕНПНЗОАТАЬОВОКННЕЬВЛДИРИЯЦТИА

Process finished with exit code 0
```

Figure 3: route_out

3. Реализовал программу для шифрования с помощью решеток (1/3)

3. Реализовал программу для шифрования с помощью решеток (1/3)

lab_2_reshetka.py

```
print("---------------------------------------")
print(" Шифрование с использованием решеток")
print("---------------------------------------")
import numpy as np
from random import randint
alphabet = "а б в г д е 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"
alphabet = alphabet.split()
def print_matrix(matrix):
    for row in matrix:
        print(row)
    print()
def sorted_matrix(a, N, M):
    k = M-1
    while k > 0:
        ind = 0
        for j in range(k+1):
            if a[N-1][j] > a[N-1][ind]:
                ind = j
        for i in range(N):
            m = a[i][ind]
            a[i][ind] = a[i][k]
            a[i][k] = m
        k -= 1
    return a
def encrypt_with_grid(k):
    encrypted = np.zeros((2*k, 2*k))
    for _ in range(4):
        for i in range(k):
            for j in range(k):
                encrypted[i][j] = i * k + j + 1
```

Figure 4: grid1

3. Реализовал программу для шифрования с помощью решеток (2/3)

3. Реализовал программу для шифрования с помощью решеток (2/3)

```
        encrypted = np.rot90(encrypted)
        print(encrypted)
        print()
    uniq = [num for num in range(1, k**2+1)]
    indexes = []
    while len(uniq) > 0:
        for i, lst in enumerate(encrypted):
            if len(uniq) == 0:
                break
            index = np.where(lst == uniq[0])
            index = index[0]
            if len(index) == 1:
                indexes.append((i, index[0]))
                uniq.pop(0)
            elif len(index) > 1:
                prob = randint(0, 1)
                if prob == 1:
                    indexes.append((i, index[-1]))
                else:
                    indexes.append((i, index[0]))
                uniq.pop(0)
    print("Индексы: ", indexes)
    print()
    encrypted_matrix = np.chararray((k * 2, k * 2), unicode=True)
    ind_text = 0
    for _ in range(k*2):
        for ind in indexes:
            encrypted_matrix[ind[0]][ind[1]] = text[ind_text]
            ind_text += 1
```

Figure 5: grid2

3. Реализовал программу для шифрования с помощью решеток (3/3)

3. Реализовал программу для шифрования с помощью решеток (3/3)

```
        encrypted_matrix = np.rot90(encrypted_matrix)
        print(encrypted_matrix)
        print()
    order_passw = []
    for letter in password:
        order_passw.append(alphabet.index(letter))
    matrix = [np.ndarray.tolist(row) for row in encrypted_matrix]
    matrix.append(order_passw)
    print_matrix(matrix)
    sorted_matrix(matrix, len(matrix), len(matrix[0]))
    print_matrix(matrix)
    shifr = ""
    for i in range(len(matrix) - 1):
        for j in range(len(matrix) - 1):
            shifr += matrix[j][i]
    print("\nВведенное сообщение    : ", text)
    print("Зашифрованное сообщение: ", shifr.upper())
def encrypt(text, password):
    k = int(len(text)**0.25)
    if k**4 != len(text):
        raise ValueError("Length of the text should be a perfect square.")

    if len(password) != k**2:
        raise ValueError(f"Length of the password should be {k}.")

    encrypt_with_grid(2)

text = "договор подписали".replace(" ", "")
password = "шифр"
encrypt(text, password)
```

Figure 6: grid3

4. Вывод работы второй программы (1/3)

4. Вывод работы второй программы (1/3)

```
---------------------------------------
 Шифрование с использованием решеток
---------------------------------------
[[0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [2. 4. 0. 0.]
 [1. 3. 0. 0.]]

[[0. 0. 0. 0.]
 [0. 0. 0. 0.]
 [2. 4. 4. 3.]
 [1. 3. 2. 1.]]

[[0. 0. 3. 1.]
 [0. 0. 4. 2.]
 [2. 4. 4. 3.]
 [1. 3. 2. 1.]]

[[1. 2. 3. 1.]
 [3. 4. 4. 2.]
 [2. 4. 4. 3.]
 [1. 3. 2. 1.]]

Индексы:  [(0, 0), (1, 3), (2, 3), (1, 2)]

[['' 'o' 'г' '']
 ['' 'o' '' '']
```

4. Вывод работы второй программы (2/3)

4. Вывод работы второй программы (2/3)

```
[['' 'о' 'р' '']
 ['г' 'п' '' '']
 ['о' 'о' '' '']
 ['в' '' '' 'д']]

[['' 'д' 'п' 'д']
 ['р' 'и' '' '']
 ['о' 'п' 'о' '']
 ['о' 'г' 'о' 'в']]

[['д' 'а' 'л' 'в']
 ['п' 'и' 'о' 'о']
 ['д' 'и' 'п' 'г']
 ['с' 'р' 'о' 'о']]

['д', 'а', 'л', 'в']
['п', 'и', 'о', 'о']
['д', 'и', 'п', 'г']
['с', 'р', 'о', 'о']
[24, 8, 20, 16]

['а', 'в', 'л', 'д']
['и', 'о', 'о', 'п']
['и', 'г', 'п', 'д']
```

4. Вывод работы второй программы (3/3)

4. Вывод работы второй программы (3/3)

```
Введенное сообщение    :  договорподписали
Зашифрованное сообщение:  АИИРВОГОЛОПОДПДС

Process finished with exit code 0
```

Figure 9: grid_out3

5. Реализовал программу для шифрования таблицей Виженера (1/2)

5. Реализовал программу для шифрования таблицей Виженера (1/2)

lab_2_vizhenera.py

```
print("--------------------")
print(" Таблица Виженера")
print("--------------------")

alphabet = "а б в г д е 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"
alphabet = alphabet.split()
mess = str(input("Введите предложение, которое нужно зашифровать: ")).lower()
# mess = 'криптография серьезная наука'
mess = mess.replace(' ', '')
mess = mess.replace('\t', '')
orig_mess = mess
password = str(input("Введите пароль: ")).lower()
# password = 'математика'
password = password.replace(' ', '')
password = password.replace('\t', '')

print("Введенное сообщение без пробелов: ", orig_mess)
print("Введенный пароль без пробелов:    ", password)

# записываем пароль такой же длины, как и предложение
mess = list(mess)
passw = list()
for i in range(0, len(mess)):
    index = i % len(password)
    passw.append(password[index])

print("\nПароль над предложением:")
print(passw)
print(mess)
```

Figure 10: viginere1

5. Реализовал программу для шифрования таблицей Виженера (2/2)

5. Реализовал программу для шифрования таблицей Виженера (2/2)

```
31  # создаем таблицу для шифрования
32  table = list()
33  table.append(alphabet)
34  for i in range(1, len(alphabet)):
35      tmp_alph = alphabet[i:] + alphabet[:i]
36      table.append(tmp_alph)
37
38  print("\nТаблица шифрования")
39  for t in table:
40      print(t)
41
42  # шифруем сообщение по таблице
43  cypher_mess = str()
44  for i in range(0, len(passw)):
45      row = alphabet.index(passw[i])
46      col = alphabet.index(mess[i])
47      cypher_mess += table[row][col].upper()
48
49  print("\nВведенное сообщение без пробелов: ", orig_mess)
50  print("Зашифрованное сообщение:          ", cypher_mess)
```

Figure 11: viginere2

6. Вывод работы третьей программы (1/3)

6. Вывод работы третьей программы (1/3)

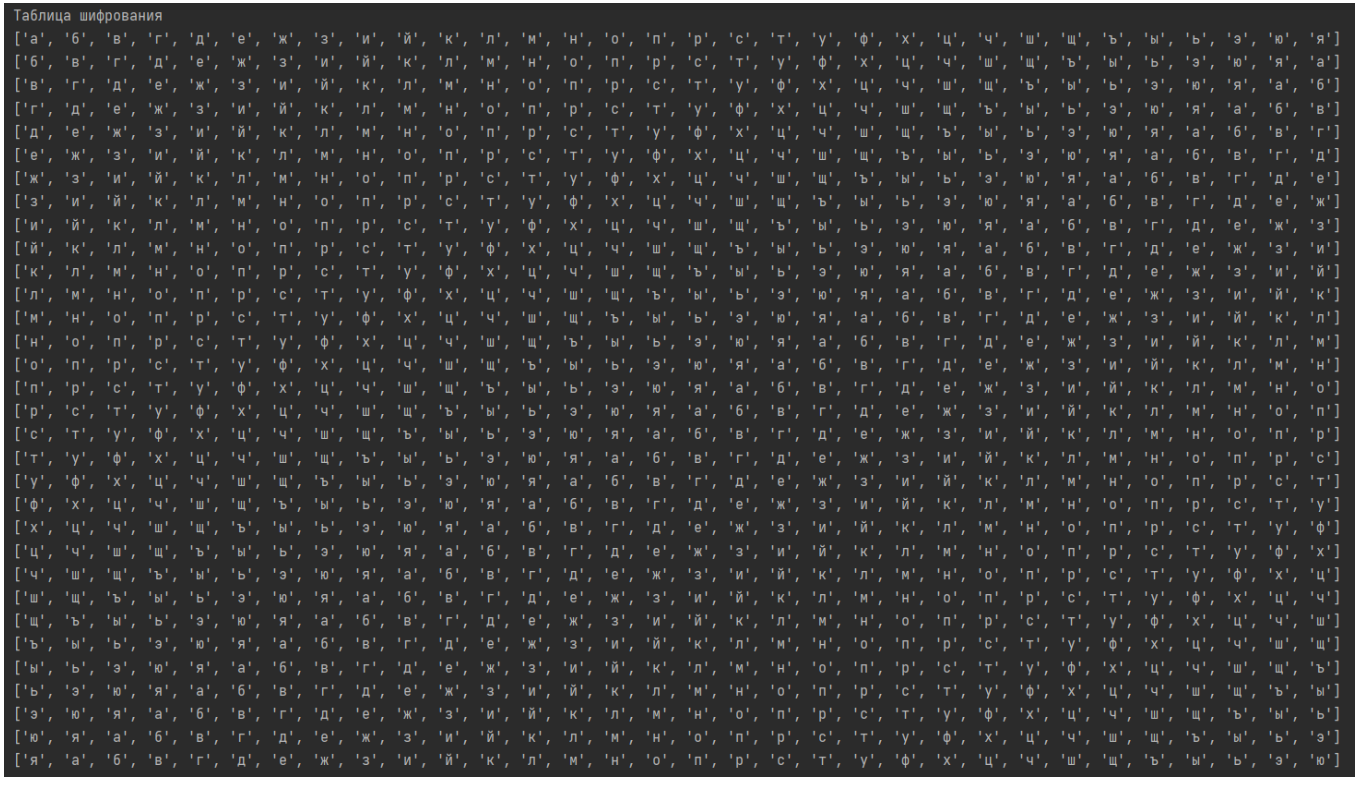
```
------------------
 Таблица Виженера
------------------
Введите предложение, которое нужно зашифровать: криптография серьезная наука
Введите пароль: математика
Введенное сообщение без пробелов:  криптографиясерьезнаянаука
Введенный пароль без пробелов:     математика

Пароль над предложением:
['м', 'а', 'т', 'е', 'м', 'а', 'т', 'и', 'к', 'а', 'м', 'а', 'т', 'е', 'м', 'а', 'т', 'и', 'к', 'а', 'м', 'а', 'т', 'е', 'м', 'а']
['к', 'р', 'и', 'п', 'т', 'о', 'г', 'р', 'а', 'ф', 'и', 'я', 'с', 'е', 'р', 'ь', 'е', 'з', 'н', 'а', 'я', 'н', 'а', 'у', 'к', 'а']
```

Figure 12: viginere_out1

6. Вывод работы третьей программы (2/3)

6. Вывод работы третьей программы (2/3)



```
Таблица шифрования
['а', 'б', 'в', 'г', 'д', 'е', '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']
['б', 'в', 'г', 'д', 'е', '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', 'а']
['в', 'г', 'д', 'е', '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', 'а', 'б']
['г', 'д', 'е', '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', 'а', 'б', 'в']
['д', 'е', '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', 'а', 'б', 'в', 'г']
['е', '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', 'а', 'б', 'в', 'г', 'д']
['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', 'а', 'б', 'в', 'г', 'д', 'е']
['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', 'а', 'б', 'в', 'г', 'д', 'е', 'ж']
['и', 'й', 'к', 'л', 'м', 'н', 'о', 'п', 'р', 'с', 'т', 'у', 'ф', 'х', 'ц', 'ч', 'ш', 'щ', 'ъ', 'ы', 'ь', 'э', 'ю', 'я', 'а', 'б', 'в', 'г', 'д', 'е', 'ж', 'з']
['й', 'к', 'л', 'м', 'н', 'о', 'п', 'р', 'с', 'т', 'у', 'ф', 'х', 'ц', 'ч', 'ш', 'щ', 'ъ', 'ы', 'ь', 'э', 'ю', 'я', 'а', 'б', 'в', 'г', 'д', 'е', 'ж', 'з', 'и']
['к', 'л', 'м', 'н', 'о', 'п', 'р', 'с', 'т', 'у', 'ф', 'х', 'ц', 'ч', 'ш', 'щ', 'ъ', 'ы', 'ь', 'э', 'ю', 'я', 'а', 'б', 'в', 'г', 'д', 'е', 'ж', 'з', 'и', 'й']
['л', 'м', 'н', 'о', 'п', 'р', 'с', 'т', 'у', 'ф', 'х', 'ц', 'ч', 'ш', 'щ', 'ъ', 'ы', 'ь', 'э', 'ю', 'я', 'а', 'б', 'в', 'г', 'д', 'е', 'ж', 'з', 'и', 'й', 'к']
['м', 'н', 'о', 'п', 'р', 'с', 'т', 'у', 'ф', 'х', 'ц', 'ч', 'ш', 'щ', 'ъ', 'ы', 'ь', 'э', 'ю', 'я', 'а', 'б', 'в', 'г', 'д', 'е', 'ж', 'з', 'и', 'й', 'к', 'л']
['н', 'о', 'п', 'р', 'с', 'т', 'у', 'ф', 'х', 'ц', 'ч', 'ш', 'щ', 'ъ', 'ы', 'ь', 'э', 'ю', 'я', 'а', 'б', 'в', 'г', 'д', 'е', 'ж', 'з', 'и', 'й', 'к', 'л', 'м']
['о', 'п', 'р', 'с', 'т', 'у', 'ф', 'х', 'ц', 'ч', 'ш', 'щ', 'ъ', 'ы', 'ь', 'э', 'ю', 'я', 'а', 'б', 'в', 'г', 'д', 'е', 'ж', 'з', 'и', 'й', 'к', 'л', 'м', 'н']
['п', 'р', 'с', 'т', 'у', 'ф', 'х', 'ц', 'ч', 'ш', 'щ', 'ъ', 'ы', 'ь', 'э', 'ю', 'я', 'а', 'б', 'в', 'г', 'д', 'е', 'ж', 'з', 'и', 'й', 'к', 'л', 'м', 'н', 'о']
['р', 'с', 'т', 'у', 'ф', 'х', 'ц', 'ч', 'ш', 'щ', 'ъ', 'ы', 'ь', 'э', 'ю', 'я', 'а', 'б', 'в', 'г', 'д', 'е', 'ж', 'з', 'и', 'й', 'к', 'л', 'м', 'н', 'о', 'п']
['с', 'т', 'у', 'ф', 'х', 'ц', 'ч', 'ш', 'щ', 'ъ', 'ы', 'ь', 'э', 'ю', 'я', 'а', 'б', 'в', 'г', 'д', 'е', 'ж', 'з', 'и', 'й', 'к', 'л', 'м', 'н', 'о', 'п', 'р']
['т', 'у', 'ф', 'х', 'ц', 'ч', 'ш', 'щ', 'ъ', 'ы', 'ь', 'э', 'ю', 'я', 'а', 'б', 'в', 'г', 'д', 'е', 'ж', 'з', 'и', 'й', 'к', 'л', 'м', 'н', 'о', 'п', 'р', 'с']
['у', 'ф', 'х', 'ц', 'ч', 'ш', 'щ', 'ъ', 'ы', 'ь', 'э', 'ю', 'я', 'а', 'б', 'в', 'г', 'д', 'е', 'ж', 'з', 'и', 'й', 'к', 'л', 'м', 'н', 'о', 'п', 'р', 'с', 'т']
['ф', 'х', 'ц', 'ч', 'ш', 'щ', 'ъ', 'ы', 'ь', 'э', 'ю', 'я', 'а', 'б', 'в', 'г', 'д', 'е', 'ж', 'з', 'и', 'й', 'к', 'л', 'м', 'н', 'о', 'п', 'р', 'с', 'т', 'у']
['х', 'ц', 'ч', 'ш', 'щ', 'ъ', 'ы', 'ь', 'э', 'ю', 'я', 'а', 'б', 'в', 'г', 'д', 'е', 'ж', 'з', 'и', 'й', 'к', 'л', 'м', 'н', 'о', 'п', 'р', 'с', 'т', 'у', 'ф']
['ц', 'ч', 'ш', 'щ', 'ъ', 'ы', 'ь', 'э', 'ю', 'я', 'а', 'б', 'в', 'г', 'д', 'е', 'ж', 'з', 'и', 'й', 'к', 'л', 'м', 'н', 'о', 'п', 'р', 'с', 'т', 'у', 'ф', 'х']
['ч', 'ш', 'щ', 'ъ', 'ы', 'ь', 'э', 'ю', 'я', 'а', 'б', 'в', 'г', 'д', 'е', 'ж', 'з', 'и', 'й', 'к', 'л', 'м', 'н', 'о', 'п', 'р', 'с', 'т', 'у', 'ф', 'х', 'ц']
['ш', 'щ', 'ъ', 'ы', 'ь', 'э', 'ю', 'я', 'а', 'б', 'в', 'г', 'д', 'е', 'ж', 'з', 'и', 'й', 'к', 'л', 'м', 'н', 'о', 'п', 'р', 'с', 'т', 'у', 'ф', 'х', 'ц', 'ч']
['щ', 'ъ', 'ы', 'ь', 'э', 'ю', 'я', 'а', 'б', 'в', 'г', 'д', 'е', 'ж', 'з', 'и', 'й', 'к', 'л', 'м', 'н', 'о', 'п', 'р', 'с', 'т', 'у', 'ф', 'х', 'ц', 'ч', 'ш']
['ъ', 'ы', 'ь', 'э', 'ю', 'я', 'а', 'б', 'в', 'г', 'д', 'е', 'ж', 'з', 'и', 'й', 'к', 'л', 'м', 'н', 'о', 'п', 'р', 'с', 'т', 'у', 'ф', 'х', 'ц', 'ч', 'ш', 'щ']
['ы', 'ь', 'э', 'ю', 'я', 'а', 'б', 'в', 'г', 'д', 'е', 'ж', 'з', 'и', 'й', 'к', 'л', 'м', 'н', 'о', 'п', 'р', 'с', 'т', 'у', 'ф', 'х', 'ц', 'ч', 'ш', 'щ', 'ъ']
['ь', 'э', 'ю', 'я', 'а', 'б', 'в', 'г', 'д', 'е', 'ж', 'з', 'и', 'й', 'к', 'л', 'м', 'н', 'о', 'п', 'р', 'с', 'т', 'у', 'ф', 'х', 'ц', 'ч', 'ш', 'щ', 'ъ', 'ы']
['э', 'ю', 'я', 'а', 'б', 'в', 'г', 'д', 'е', 'ж', 'з', 'и', 'й', 'к', 'л', 'м', 'н', 'о', 'п', 'р', 'с', 'т', 'у', 'ф', 'х', 'ц', 'ч', 'ш', 'щ', 'ъ', 'ы', 'ь']
['ю', 'я', 'а', 'б', 'в', 'г', 'д', 'е', 'ж', 'з', 'и', 'й', 'к', 'л', 'м', 'н', 'о', 'п', 'р', 'с', 'т', 'у', 'ф', 'х', 'ц', 'ч', 'ш', 'щ', 'ъ', 'ы', 'ь', 'э']
['я', 'а', 'б', 'в', 'г', 'д', 'е', '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']
```

Figure 13: viginere_out2

6. Вывод работы третьей программы (3/3)

6. Вывод работы третьей программы (3/3)

```
Введенное сообщение без пробелов:  криптографиясерьезнаянаука
Зашифрованное сообщение:           ЦРЪФЮОХШКФФЯГКЬЬЧПЧАЛНТШЦА

Process finished with exit code 0
```

Figure 14: viginere_out3

Вывод

Вывод

В результате выполнения работы я освоил на практике применение шифров перестановки.