Отчёт по лабораторной работе №2  
Шифры перестановки

Студент: Леонова Алина Дмитриевна, 1032212306

Группа: НФИмд-01-21

Преподаватель: Кулябов Дмитрий Сергеевич,

д-р.ф.-м.н., проф.

Москва 2021

Содержание

# 1 Цель работы

Целью данной работы является ознакомление с шифрами перестановки и их реализация на выбранном языке программирования.

# 2 Задание

Реализовать все рассмотренные шифры программно.

# 3 Теоретическое введение

Шифры перестановки преобразуют открытый текст в криптограмму путем перестановки его символов. Способ, каким при шифровании переставляются буквы открытого текста, и является ключем шифра.

## 3.1 1. Маршрутное шифрование (Маршрутная перестановка)

Маршрутная перестановка — это шифр вертикальной перестановки, метод симметричного шифрования, в котором элементы исходного открытого текста меняют местами [1].



Figure 1: Пример шифр перестановки (1)



Figure 2: Пример шифр перестановки (2)

В этом шифре также используется прямоугольная таблица, в которую сообщение записывается по строкам слева направо. Выписывается шифрограмма по вертикалям, при этом столбцы выбираются в порядке, определяемом ключом (см. рис. 1). Также возможна вариация, когда ключём служит пароль и есть договорённость как его использовать, например, в алфпвитном порядке (см. рис. 2).

## 3.2 2. Шифрование с помощью решеток

Выбирается натуральное число k > 1, и квадрат размерности k×k построчно заполняется числами 1, 2, …, k. Для примера возьмем k = 2.

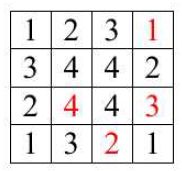


Figure 3: Пример шифрования с помощью решетки (1)

Квадрат поворачивается по часовой стрелке на 90° и размещается вплотную к предыдущему квадрату. Аналогичные действия совершаются еще два раза, так чтобы в результате из четырех малых квадратов образовался один большой с длиной стороны 2k (см. рис. 3).

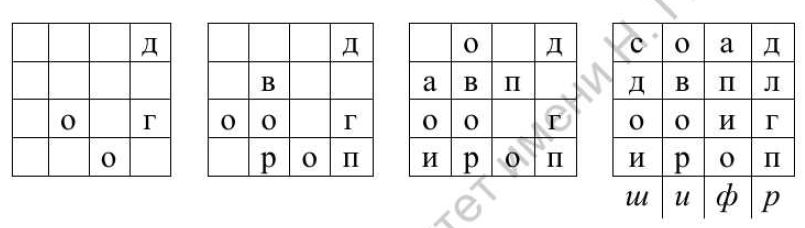


Figure 4: Пример шифрования с помощью решетки (2)

Далее из большого квадрата вырезаются клетки с числами от 1 до k2, для каждого числа одна клетка. Процесс шифрования происходит следующим образом. Сделанная решетка (квадрат с прорезями) накладывается на чистый квадрат 2k×2k и в прорези по строчкам (т.е. слева направо и сверху вниз) вписываются первые буквы открытого текста. Затем решетка поворачивается на 90° по часовой стрелке и накладывается на частично заполненный квадрат, вписывание продолжается.

После третьего поворота, наложения и вписывания все клетки квадрата будут заполнены. Например, с использованием изображенной выше решетки и пароля ш и ф р открытый текст договор подписали переводится в криптограмму за пять шагов, итоговая криптограмма: ОВОРДЛГПАПИОСДОИ (см. рис. 4) [2].

## 3.3 3. Таблица Виженера

Шифр Виженера — метод полиалфавитного шифрования буквенного текста с использованием ключевого слова. Этот метод является простой формой многоалфавитной замены [3].

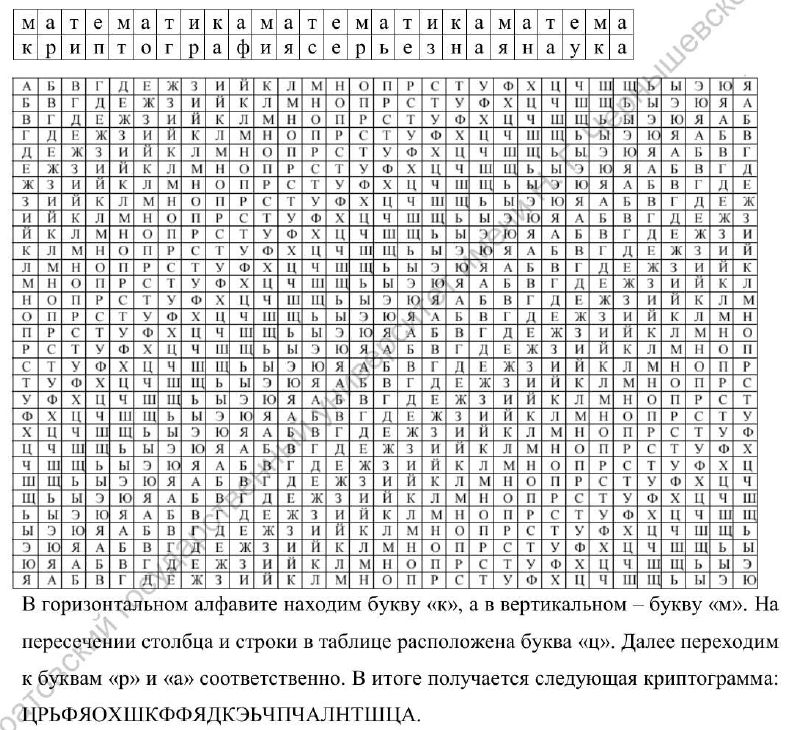


Figure 5: Пример использования таблицы Виженера

Пароль записывается с повторениями пока не поровняется по длине с текстом, который требуется зашифровать. Для шифрования и дешифрования используется таблица Виженера, она содержит все возможные смещения интересующего алфавита (см. рис. 5).

# 4 Выполнение лабораторной работы

## 4.1 0. Импортирование библиотек и промежуточные функции

Для первого и второго шифров в конце требцется делать одно и то же действие:

import numpy as np  
import math  
  
def grid\_reading(key, table):  
 \_key = sorted(key)  
 order = [key.index(i) for i in \_key]  
 print(list(key))  
 #print(\_key)  
 print('Порядок использования столбцов:', order)  
 m = table.shape[0]  
 res = ''  
 for j in order:  
 for i in range(m):  
 res += table[i][j]  
 return res  
  
ru = [chr(i) for i in range(ord('а'), ord('я')+1)]  
print('ru: ',ru)

## 4.2 1. Маршрутное шифрование (Маршрутная перестановка)

print('####################################################################')  
# 1. Маршрутное шифрование  
print('# 1.')  
def task1\_rout(text, key):  
 print('Текст: ', text)  
 text = text.replace(' ','').lower()  
   
 print('Ключ:', key)  
 key = key.lower()  
   
 n = len(key)  
 t\_len = len(text)  
 m = math.ceil(t\_len/n)  
 A = np.full((m,n),'')  
 print('Длина текста:', t\_len)  
 print('n = ', n, '\nm = ', m)  
   
 for k in range(n\*m - t\_len):  
 text += text[-1]  
 #print(text)  
   
 k = 0  
 for i in range(m):  
 for j in range(n):  
 A[i][j] = text[k]  
 k += 1  
 print(A)  
   
 res = grid\_reading(key, A)  
   
 print('Криптограмма: ', res)  
   
  
text = 'Нельзя недооценивать противника'  
key = 'пароль'  
task1\_rout(text, key)  
  
print('----------------------------------------------------')  
text = 'Live long and prosper'  
key = 'Spock'  
task1\_rout(text, key)

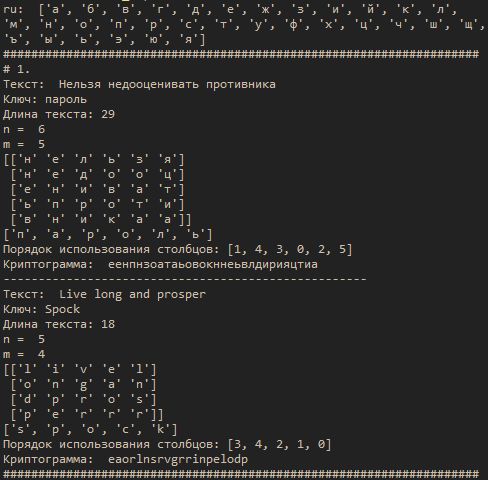


Figure 6: Результат выполнения (1)

Результат выполнения этого фрагмента кода, реализации маршрутного шифрования и проверкой (см. рис. 6).

## 4.3 2. Шифрование с помощью решеток

print('####################################################################')  
# 2. Шифрование с помощью решеток  
  
def rotate\_90r(A):  
 x = A.shape[0]  
 y = A.shape[1]  
 res = np.empty((y,x))  
 for i in range(x):  
 for j in range(y):  
 res[j, x-1-i] = A[i,j]  
 return res  
   
def generate\_key(lenth):  
 key = ''  
 while (len(key) != lenth):  
 \_key = np.random.randint(len(ru))  
 if (key.count(ru[\_key]) == 0):  
 key += ru[\_key]  
 return key  
  
  
print('# 2.')  
def task2\_grid(text):  
 print('Текст: ', text)  
 text = text.replace(' ','').lower()  
   
 t\_len = len(text)  
 print('Длина текста:', t\_len)  
   
 size = math.ceil(np.sqrt(t\_len)) # количество чисел в маленькой таблице  
 while size % np.sqrt(size) != 0:  
 size += 1  
   
 for k in range(size\*size - t\_len):  
 text += text[-1]  
   
 t\_s = int(np.sqrt(size)) # размер малькой таблицы  
 t\_s2 = t\_s \* 2 # размер большой таблицы  
 t\_el = np.arange(1,size+1) # массив значений size  
   
 key = generate\_key(t\_s2)   
 #key = 'шифр'  
 print('Ключ:', key)  
 key = key.lower()  
   
   
 A0 = np.empty((t\_s,t\_s))   
 n = 0  
 for i in range(t\_s):  
 for j in range(t\_s):  
 A0[i][j] = t\_el[n]  
 n += 1  
   
 A1 = np.concatenate((A0,rotate\_90r(A0)),axis=1)  
 A2 = np.concatenate((A1,rotate\_90r(rotate\_90r(A1))),axis=0)   
 #print(A0)  
 #print(A1)  
 print('Квадрат цифр:\n', A2)  
   
 R = np.zeros((t\_s2,t\_s2))  
 tmp = t\_el.copy()  
 for k in range (size):  
 r = np.random.randint(4)  
 tmp\_count = 0  
 for i in range (t\_s2):  
 for j in range (t\_s2):  
 if (A2[i][j] == k + 1):  
 if(tmp\_count == r):  
 R[i][j] = A2[i][j]  
 tmp\_count = -100  
 tmp = tmp[tmp != k+1]  
 else:  
 tmp\_count += 1  
 print('Сгенерированная схема:\n', R)  
   
 n = 0  
 answer = np.full((t\_s2, t\_s2),'')  
   
 for k in range(4):  
 for i in range(t\_s2):  
 for j in range(t\_s2):  
 if(R[i][j] != 0):  
 answer[i][j] = text[n]  
 n += 1  
 #print(k)  
 #print(R)  
 #print(answer)  
 R = rotate\_90r(R)  
   
 print('Схема из букв:\n', answer)  
   
 res = grid\_reading(key, answer)  
 print('Криптограмма: ', res)  
  
  
text = 'договор подписали'  
task2\_grid(text)  
  
print('----------------------------------------------------')  
  
text = 'за что мне все эти страдания'  
task2\_grid(text)

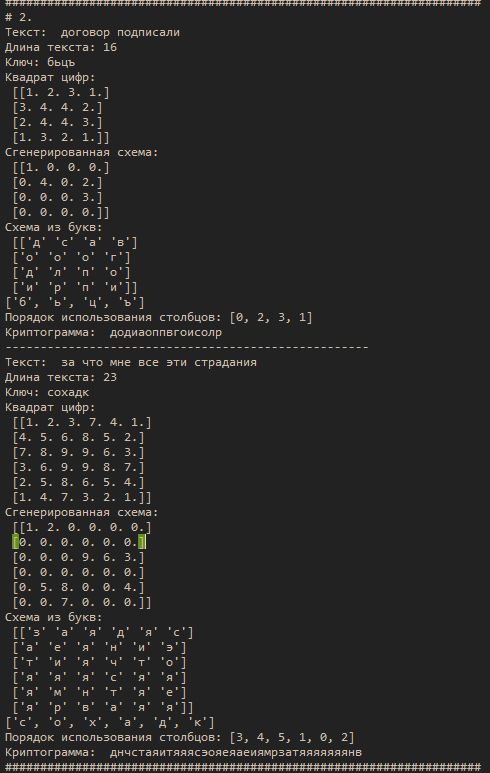


Figure 7: Результат выполнения (2)

Результат выполнения этого фрагмента кода, реализации шифрования с помощью решеток с проверкой (см. рис. 7).

## 4.4 3. Таблица Виженера

print('####################################################################')  
letters = {ru[i]:i for i in range(len(ru))}  
print('Словарь букв ru: ', letters)  
  
# 3. Таблица Виженера  
print('# 3.')  
  
vigenere\_table = np.array(ru)  
for i in range(1, len(ru)):  
 row = np.roll(ru, -i)  
 vigenere\_table = np.vstack((vigenere\_table, row))  
print('Таблица Виженера: \n',vigenere\_table)  
  
def task3\_vigenere(text, key):  
 print('Текст: ', text)  
 text = text.replace(' ','').lower()  
   
 t\_len = len(text)  
 print('Длина текста:', t\_len)  
   
 print('Ключ:', key)  
 key = key.lower()  
   
 n = len(key)  
 \_key = key  
   
 while len(\_key) < t\_len:  
 \_key += \_key[len(\_key) - n]  
 print('-----')  
 print(text)  
 print(\_key)  
 print('-----')  
   
 res = ''  
 for i in range(t\_len):  
 x = letters[\_key[i]] # номера букв ключа  
 y = letters[text[i]] # номера букв текста  
   
 res += vigenere\_table[x][y]  
 print('Криптограмма: ', res)  
  
  
text = 'криптография серьезная наука'  
key = 'математика'  
task3\_vigenere(text, key)  
  
print('----------------------------------------------------')  
text = 'ты не пройдешь'  
key = 'Гендальф'  
task3\_vigenere(text, key)

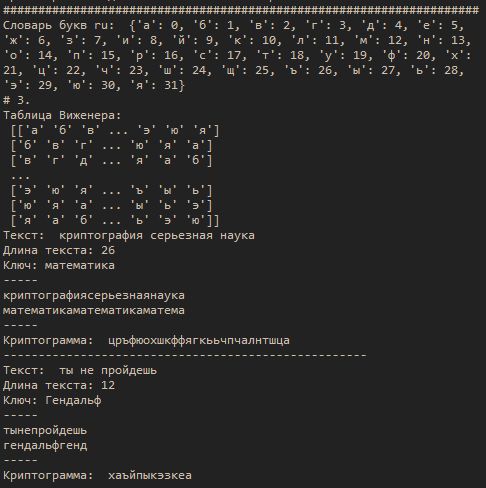


Figure 8: Результат выполнения (3)

Результат выполнения этого фрагмента кода, реализации использования таблицы Виженера с проверкой (см. рис. 8).

# 5 Выводы

Цель лабораторной работы была достигнута, три данные шифра перестановки были изучены и реализованы на языке программирования Python.

# Список литературы

1. NeverWalkAloner. Перестановочный шифр [Электронный ресурс]. Википедия, 2021. URL: <https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D0%B5%D1%81%D1%82%D0%B0%D0%BD%D0%BE%D0%B2%D0%BE%D1%87%D0%BD%D1%8B%D0%B9_%D1%88%D0%B8%D1%84%D1%80>.

2. Перестановочные шифры. [Электронный ресурс]. IT1406: Информационная безопасность, 2021. URL: <https://it.rfei.ru/course/~k017/~7mdCpor7/~c5kOtaHY>.

3. Шифр Виженера [Электронный ресурс]. Википедия, 2021. URL: <https://ru.wikipedia.org/wiki/%D0%A8%D0%B8%D1%84%D1%80_%D0%92%D0%B8%D0%B6%D0%B5%D0%BD%D0%B5%D1%80%D0%B0>.