Математические основы защиты информации и информационной безопасности

Отче по лабораторной работе № 2

Кейела Патачона - НПМмд-02-21

Содержание

List of Figures

1 Цель работы

Изучение алгоритмов маршрутной перестановки, решеток и Виженера

2 Теоретические сведения

2.1 Шифр маршрутной перестановки

Широкое распространение получили шифры перестановки, использующие некоторую геометрическую фигуру. Преобразования из этого шифра состоят в том, что в фигуру исходный текст вписывается по ходу одного “маршрута’ ’, а затем по ходу другого выписывается с нее. Такой шифр называют маршрутной перестановкой. Например, можно вписывать исходное сообщение в прямоугольную таблицу, выбрав такой маршрут: по горизонтали, начиная с левого верхнего угла поочередно слева направо и справа налево. Выписывать же сообщение будем по другому маршруту: по вертикали, начиная с верхнего правого угла и двигаясь поочередно сверху вниз и снизу вверх.

2.2 Шифр Кардано

Решётка Кардано — инструмент кодирования и декодирования, представляющий собой специальную прямоугольную (в частном случае — квадратную) таблицу-карточку, четверть ячеек которой вырезана.

Таблица накладывается на носитель, и в вырезанные ячейки вписываются буквы, составляющие сообщение. После переворачивания таблицы вдоль вертикальной оси, процесс вписывания букв повторяется. Затем то же самое происходит после переворачивания вдоль горизонтальной и снова вдоль вертикальной осей.

В частном случае квадратной таблицы, для получения новых позиций для

вписывания букв, можно поворачивать квадрат на четверть оборота.

Чтобы прочитать закодированное сообщение, необходимо наложить решётку Кардано нужное число раз на закодированный текст и прочитать буквы, расположенные в вырезанных ячейках.

Такой способ шифрования сообщения был предложен математиком Джероламо Кардано в 1550 году, за что и получил своё название.

2.3 Шифр Виженера

Шифр Виженера (фр. Chiffre de Vigenère) — метод полиалфавитного шифрования буквенного текста с использованием ключевого слова.

Этот метод является простой формой многоалфавитной замены. Шифр Виженера изобретался многократно. Впервые этот метод описал Джован Баттиста Беллазо (итал. Giovan Battista Bellaso) в книге La cifra del. Sig. Giovan Battista Bellaso в 1553 году, однако в XIX веке получил имя Блеза Виженера, французского дипломата. Метод прост для понимания и реализации, он является недоступным для простых методов криптоанализа.

В шифре Цезаря каждая буква алфавита сдвигается на несколько строк; например в шифре Цезаря при сдвиге +3, A стало бы D, B стало бы E и так далее. Шифр Виженера состоит из последовательности нескольких шифров Цезаря с различными значениями сдвига. Для зашифровывания может использоваться таблица алфавитов, называемая tabula recta или квадрат (таблица) Виженера. Применительно к латинскому алфавиту таблица Виженера составляется из строк по 26 символов, причём каждая следующая строка сдвигается на несколько позиций. Таким образом, в таблице получается 26 различных шифров Цезаря. На каждом этапе шифрования используются различные алфавиты, выбираемые в зависимости от символа ключевого слова.

3 Выполнение работы

3.1 Реализация шифра маршрутной перестановки на языке Python

```
# Мы принимаем пароль и текст на шрифрование
password = str(input('Введите пароль:'))
my_word = ''.join(str(input('Введите фразу:')).split())

# Длина прямоугольника
n = len(password)
# Дополняем если нужно
if len(my_word)%n != 0:
    my_word += 'a'*(n-len(my_word) % n)

# Отсортируем буквуы пароля
password_sort = ''.join(sorted(password))
index_list = []
for i in range (len(password)):
    f_index = password.find(password_sort[i])
    index_list.append(f_index)

# Шифруем наш текст
new_word = ''
```

```
for i in index_list:
    for j in range(len(my_word)//n):
        new_word += my_word[j*n+i]

# Выводим отшифрованный текст
print(new_word)
```

3.2 Реализация шифра решеткой на языке Python

```
import numpy as np

# We enter to word to encode
word_to_encode = list(''.join(input().split()).upper())
print(word_to_encode)

# wE DEFINE THE VALUE OF K: 4k**2 = N
if int(np.sqrt(len(word_to_encode)/4)) == np.sqrt(len(word_to_encode)/4):
  k = int(np.sqrt(len(word_to_encode)/4))
else:
  # wE FILL THE EMPTY POSITIONS WITH STARS
  k = int(np.sqrt(len(word_to_encode)/4)) + 1
  word_to_encode += ['*']*(4 * k**2 - len(word_to_encode))
print(word_to_encode)

# Simple matrix of k * k
k_matrix = np.reshape(np.arange(k**2),(k,k))
print(k_matrix,'\n')

# Complete matrix:
```

```
full_matrix = np.concatenate((np.concatenate((k_matrix,np.rot90(k_matrix,
1)),axis=1),
                              np.concatenate((np.rot90(k_matrix,-
3),np.rot90(k_matrix,-2)),axis=1)
                              ),axis=0)
print(full_matrix)

full_matrix_1d = full_matrix.flatten()
print(full_matrix_1d)
slovar = {}
for i in range(k**2):
  slovar[i] = np.where(full_matrix_1d==i)[0]
print('Dictionary: ',slovar)

# WE RANDOMLY CHOOSE OUR NUMBERS
positions = []
pos = 0
for i in range(k**2):
  pos = np.random.choice(slovar[i])
  positions.append((pos // (2*k) ,pos % (2*k)))
print(positions)

# IN THIS PART WE CREATE AND
# COMPUTE IT
encoding_matrix = np.zeros((2*k,2*k),dtype=int)
tmp1 = np.zeros((2*k,2*k),dtype=int)
tmp2 = np.zeros((2*k,2*k),dtype=int)
tmp3 = np.zeros((2*k,2*k),dtype=int)
i = 0
```

```
for x, y in positions:
  encoding_matrix[x,y] = i
  tmp1[x,y] = i + k**2
  tmp2[x,y] = i + 2*k**2
  tmp3[x,y] = i + 3*k**2
  i += 1
print(encoding_matrix)

# WE ADD ROTATED MATRIX
encoding_matrix += np.rot90(tmp1,-1)
print(encoding_matrix)
encoding_matrix += np.rot90(tmp2,-2)
print(encoding_matrix)
encoding_matrix += np.rot90(tmp3,-3)
print(encoding_matrix)

print(encoding_matrix)

# WE GET THE PASSWORD AND
# wE CHECK IF IT'S CORRECT
good_password = False
while not good_password:
  password = input(f'Enter a password of {2*k} characters: ')
  if len(password) == 2*k:
    good_password = True

# We get the order of the letters in the password
chars, index = np.unique(list(password),return_index=True)
print(chars,index)
```

```
# ENCODING OUR MESSAGE USING
# THE ORDER GOT FROM THE PASSWORD
encoded_text = ''
for col in index:
  for row in range(2*k):
    encoded_text += word_to_encode[2*k*row + col]

print(encoded_text)
```

3.3 Реализация шифра Виженера на языке Python

```
slovar = 'абвгдеёжзийклмнопрстуфхцчшщъыьэюя'
password = str(input('Введите пароль: ')).lower()
word = str(input('Введите фразу для шифрования: ')).lower()
k = (len(word) % len(password)) # Количество символов которые нужно допол
password_len = '' + password * (len(word) // len(password)) + password[:k
print(word, password_len, sep='\n')
slovar_visinera = []
slovar_i = 'абвгдеёжзийклмнопрстуфхцчшщъыьэюя'
for i in range(len(slovar)):
    slovar_visinera.append(slovar_i)
    new = slovar_i[1:] + slovar_i[0]
    slovar_i = new
print("Квадрат вижинера:", slovar_visinera)
message = ''
for i in range(len(word)):
    f_index1 = slovar.find(word[i])
    f_index2 = slovar.find(password_len[i])
```

```
    message += slovar_visinera[f_index1][f_index2]
print(f'Зашифрованное сообщение: {message}')
```

3.4 Контрольные примеры

3.4.1 Контрольный пример 1

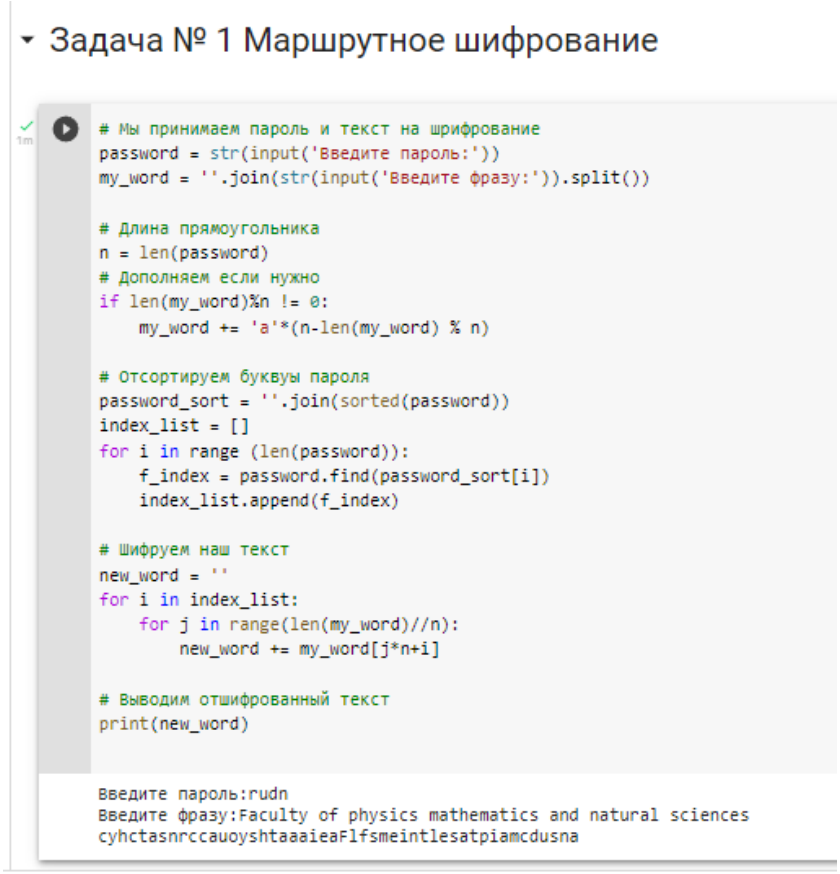

Figure 3.1: Работа алгоритма маршрутной перестановки

3.4.2 Контрольный пример 2

Задача № 2 Шифрование с помощью решеток

```
[ ] import numpy as np
```

```
# We enter to word to encode
word_to_encode = list(''.join(input().split()).upper())
print(word_to_encode)

# wE DEFINE THE VALUE OF K: 4k**2 = N
if int(np.sqrt(len(word_to_encode)/4)) == np.sqrt(len(word_to_encode)/4):
  k = int(np.sqrt(len(word_to_encode)/4))
else:
  # wE FILL THE EMPTY POSITIONS WITH STARS
  k = int(np.sqrt(len(word_to_encode)/4)) + 1
  word_to_encode += ['*']*(4 * k**2 - len(word_to_encode))
print(word_to_encode)
```

```
KEYELA PATATCHONA
['K', 'E', 'Y', 'E', 'L', 'A', 'P', 'A', 'T', 'A', 'T', 'C', 'H', 'O', 'N', 'A']
['K', 'E', 'Y', 'E', 'L', 'A', 'P', 'A', 'T', 'A', 'T', 'C', 'H', 'O', 'N', 'A']
```

```
[ ] # Simple matrix of k * k
    k_matrix = np.reshape(np.arange(k**2),(k,k))
    print(k_matrix,'\n')

    # Complete matrix:
    full_matrix = np.concatenate((np.concatenate((k_matrix,np.rot9
                                  np.concatenate((np.rot90(k_matri
                                  ),axis=0)
    print(full_matrix)
```

```
[[0 1]
 [2 3]]

[[0 1 2 0]
 [2 3 3 1]
 [1 3 3 2]
 [0 2 1 0]]
```

```
full_matrix_1d = full_matrix.flatten()
print(full_matrix_1d)
slovar = {}
for i in range(k**2):
  slovar[i] = np.where(full_matrix_1d==i)[0]
print('Dictionary: ',slovar)
```

```
[0 1 2 0 2 3 3 1 1 3 3 2 0 2 1 0]
Dictionary:  {0: array([ 0,  3, 12, 15]), 1: array([ 1,  7,  8, 14]), 2: array([ 2,  4, 11, 13]), 3: array([ 5,  6,  9, 10])}
```

```
[ ] # WE RANDOMLY CHOOSE OUR NUMBERS
    positions = []
    pos = 0
    for i in range(k**2):
      pos = np.random.choice(slovar[i])
      positions.append((pos // (2*k) ,pos % (2*k)))
    print(positions)
```

```
[(3, 3), (3, 2), (3, 1), (2, 2)]
```

```
# IN THIS PART WE CREATE AND
# COMPUTE IT
encoding_matrix = np.zeros((2*k,2*k),dtype=int)
tmp1 = np.zeros((2*k,2*k),dtype=int)
tmp2 = np.zeros((2*k,2*k),dtype=int)
tmp3 = np.zeros((2*k,2*k),dtype=int)
i = 0
for x, y in positions:
  encoding_matrix[x,y] = i
  tmp1[x,y] = i + k**2
  tmp2[x,y] = i + 2*k**2
  tmp3[x,y] = i + 3*k**2
  i += 1
print(encoding_matrix)

# WE ADD ROTATED MATRIX
encoding_matrix += np.rot90(tmp1,-1)
print(encoding_matrix)
encoding_matrix += np.rot90(tmp2,-2)
print(encoding_matrix)
encoding_matrix += np.rot90(tmp3,-3)
print(encoding_matrix)

print(encoding_matrix)
```

```
[[0 0 0 0]
 [0 0 0 0]
 [0 0 3 0]
 [0 2 1 0]]
[[ 8  9 10 12]
 [ 6 11 15 13]
 [ 5  7  3 14]
 [ 4  2  1  0]]
```

```
# WE GET THE PASSWORD AND
# WE CHECK IF IT'S CORRECT
good_password = False
while not good_password:
  password = input(f'Enter a password of {2*k} characters: ')
  if len(password) == 2*k:
    good_password = True

# We get the order of the letters in the password
chars, index = np.unique(list(password),return_index=True)
print(chars,index)
```

```
Enter a password of 4 characters: keyp
['e' 'k' 'p' 'y'] [1 0 3 2]
```

```
[ ] # ENCODING OUR MESSAGE USING
    # THE ORDER GOT FROM THE PASSWORD
    encoded_text = ''
    for col in index:
      for row in range(2*k):
        encoded_text += word_to_encode[2*k*row + col]

    print(encoded_text)
```

```
EAAOKLTHEACAYPTN
```

3.4.3 Контрольный пример 3

Задача № 3 Таблица Виженера

```
slovar = 'абвгдеёжзийклмнопрстуфхцчшщъыьэюя'
password = str(input('Введите пароль: ')).lower()
word = str(input('Введите фразу для шифрования: ')).lower()
k = (len(word) % len(password)) # Количество символов которые нужно дополнить
password_len = '' + password * (len(word) // len(password)) + password[:k]
print(word, password_len, sep='\n')
slovar_visinera = []
slovar_i = 'абвгдеёжзийклмнопрстуфхцчшщъыьэюя'
for i in range(len(slovar)):
    slovar_visinera.append(slovar_i)
    new = slovar_i[1:] + slovar_i[0]
    slovar_i = new
print("Квадрат вижинера:", slovar_visinera)
message = ''
for i in range(len(word)):
    f_index1 = slovar.find(word[i])
    f_index2 = slovar.find(password_len[i])
    message += slovar_visinera[f_index1][f_index2]
print(f'Зашифрованное сообщение: {message}')
```

```
Введите пароль: рудн
Введите фразу для шифрования: Кейела Патачона
кейела патачона
руднруднруднруд
Квадрат вижинера: ['абвгдеёжзийклмнопрстуфхцчшщъыьэюя', 'бвгдеёжзийклмнопрстуфхцчшщъыьэюяа', 'вгдеёжзийклмноп
Зашифрованное сообщение: ышнтьугэрёдеябд
```

Figure 3.2: Работа алгоритма Виженера

4 Выводы

Изучили алгоритмы шифрования с помощью перестановок

Список литературы

1. Шифр маршрутной перестановки
2. Шифр Кардано
3. Шифр Виженера