Прагматика выполнения лабораторной работы

В целях освоения программы предмета "Математические основы безопасности" студенты должны разбираться в основных принципах шифрования и дешифрования текста. На примере Шифров перестановки можно понять логику шифрования важной информации в электронных устройствах и принципы защиты информации. Все это необходимо для повышения безопасности в системе при работе с персональными или корпоративными компьютерами.

Цель выполнения лабораторной работы

Освоить на практике написание шифров простой замены.

Задачи выполнения лабораторной работы

1. Реализовать маршрутное шифрование.
2. Реализовать шифрование решеток.
3. Реализовать Таблицу Вижинера.

Результаты выполнения лабораторной работы

1. Реализовала Маршрутное шифрование.

   (рис. -@fig:001)

   ▾ Маршрутное шифрование

```
import math

from typing_extensions import Text
text=input("Введите текст, который хотите зашифровать:")
key=input("Введите слово ключ:")
len_blok=math.ceil(len(text)/len(key))

print(text, key, len_blok)

matrix=[]
j=0

for c in key:
  new=""
  j=key.index(c)
  for i in range(len_blok):
    if j<=len(text)-1:
      new=new+text[j]
      j+=len(key)
    else:
      new=new+"ь"
  new=c+new
  print(new)
  matrix.append(new)

print(matrix)
```

   У нас есть текст: нельзя недооценивать противника, и ключ- "пароль". Текст мы используем без пробелов.

Результаты выполнения лабораторной работы

2. Зашифровала текст с помощью маршрутного шифрования.

   (рис. -@fig:002)

```
print(sort)

Введите текст, который хотите зашифровать:нельзянедооцениватьпротивника
Введите слово ключ:пароль
нельзянедооцениватьпротивника пароль 5
пннеьв
аеенпн
рлдири
оьовок
лзоата
ьяцтиь
['пннеьв', 'аеенпн', 'рлдири', 'оьовок', 'лзоата', 'ьяцтиь']
['аеенпн', 'лзоата', 'оьовок', 'пннеьв', 'рлдири', 'ьяцтиь']
```

Текст разбивается на количество блоков = длине слова пароля. Чтобы все блоки были одинаковой длинны, мы заполняем оставшееся место мягким знаком. Далее, в первый блок мы записываем 0 символ текста и + каждый 6 символ. К началу каждого блока приписываем одну букву слова пароля. И в конце мы сортируем блоки по алфавиту, использую для этого первую букву блока - букву слова пароля.

Результаты выполнения лабораторной работы

3. Реализовала Шифр Виженера.

   (рис. -@fig:003)

Таблица Вижинера

```
def form_dict():
    d = {}
    iter = 0
    for i in range(0,127):
        d[iter] = chr(i)
        iter = iter +1
    return d

def encode_val(word):
    list_code = []
    lent = len(word)
    d = form_dict()

    for w in range(lent):
        for value in d:
            if word[w] == d[value]:
                list_code.append(value)
    return list_code

def comparator(value, key):
    len_key = len(key)
    dic = {}
```

Нам нужно создать словарь, для того чтобы шифровать символы. На скрине изображены 2 первые функции, с помощью которых можно представить тестовый символ английской раскладки в виде цифр (юникод раскладка поэтому 127 символов).

Результаты выполнения лабораторной работы

4. Зашифровала символы с помощью шифра Виженера.

(рис. -@fig:003)

```
def comparator(value, key):
    len_key = len(key)
    dic = {}
    iter = 0
    full = 0

    for i in value:
        dic[full] = [i,key[iter]]
        full = full + 1
        iter = iter +1
        if (iter >= len_key):
            iter = 0
    return dic

def full_encode(value, key):
    dic = comparator(value, key)
    print ('Compare full encode', dic)
    lis = []
    d = form_dict()

    for v in dic:
        go = (dic[v][0]+dic[v][1]) % len(d)
        lis.append(go)
    return lis
```

Далее мы к каждому символу и текста и слово-пароля подставляем цифру из нашего словаря. Комбинируем в новый словарь цифры шифра и цифры текста, повторяя заново цифры слово-пароля. В конце суммируем цифры и получаем новое значение, индекс которого и ищем в новом словаре - так получается шифротекст.

Результаты выполнения лабораторной работы

5. Дешифровала символы с помощью таблицы Виженера.

(рис. -@fig:001)(рис. -@fig:005)

```
    return lis

def decode_val(list_in):
    list_code = []
    lent = len(list_in)
    d = form_dict()

    for i in range(lent):
        for value in d:
            if list_in[i] == value:
               list_code.append(d[value])
    return list_code

def full_decode(value, key):
    dic = comparator(value, key)
    print ('Deshifre=', dic)
    d = form_dict()
    lis =[]

    for v in dic:
        go = (dic[v][0]-dic[v][1]+len(d)) % len(d)
        lis.append(go)
    return lis
```

Используя примерно те же действия для дешифровки символов, только цифры шифротекста - цифры слова пароля +127. Тем самым получаем первоначальный текст.

(рис. -@fig:006)

```
key_encoded = encode_val(key)
value_encoded = encode_val(word)

print ('Value= ',value_encoded)
print ('Key= ', key_encoded)

shifre = full_encode(value_encoded, key_encoded)
print ('Шифр=', ''.join(decode_val(shifre)))

decoded = full_decode(shifre, key_encoded)
print ('Decode list=', decoded)
decode_word_list = decode_val(decoded)
print ('Word=',''.join(decode_word_list))
```

```
Слово: Hello world
Ключ: paralel
Value=  [72, 101, 108, 108, 111, 32, 119, 111, 114, 108, 100]
Key=  [112, 97, 114, 97, 108, 101, 108]
Compare full encode {0: [72, 112], 1: [101, 97], 2: [108, 114], 3: [108, 97], 4: [111, 108], 5: [32, 101], 6: [119, 108], 7: [111, 112], 8:
Шифр= 9G_N\Bd`T_F
Deshifre= {0: [57, 112], 1: [71, 97], 2: [95, 114], 3: [78, 97], 4: [92, 108], 5: [6, 101], 6: [100, 108], 7: [96, 112], 8: [84, 97], 9: [95
Decode list= [72, 101, 108, 108, 111, 32, 119, 111, 114, 108, 100]
Word= Hello world
```

Выводы

В ходе данной лабораторной работы, написала программы для шифров перестановки. Поняла принцип шифрования и освоила написание шифров на языке Python.

{.standout}

Спасибо за внимание