Дисциплина «Информационная безопасность**»**

Направление - 02.03.03 Математическое обеспечение и администрирование информационных систем

Профиль - Разработка и администрирование информационных систем

Лабораторная работа №3

Полиалфавитный шифр - многоалфавитный шифр

Шифры типа «Шифра Цезаря» (то есть моноалфавитные шифры, в которых каждой букве кодируемого текста ставится в соответствие однозначно какя-то шифрованная буква) довольно-таки легко поддаются криптоанализу. Возникла потребность в разработке таких шифров, ручная расшифровка которых может потребовать очень значительных усилий. И на смену моноалфавитным шифрам пришли полиалфавитные шифры. В европейских странах это произошло в эпоху Возрождения, когда развитие торговли потребовало надёжные способы защиты информации. Одним из первых предложил полиалфавитный шифр итальянский архитектор Батисте Альберти. В последствие данный шифр получил имя дипломата XVI века Блеза де Вижинера. Также вклад в развите полиалфавитных шифров внёс немецкий аббат XVI века Иоганн Трисемус. Простым, но стойким способом полиалфавитной замены является шифр Плейфера, открытый в начале XIX века Чарльзом Уитстоном. Этот шифр использовался вплоть до I мировой войны. Последним словом в развитие полиалфавитных шифров стали так называемые роторные машины, которые позволяли легко создавать устойчивые к криптоатакам полиалфавитные шифры. Примером такой машины является немецкая машина Enigma, разработанная в 1917 г. Эдвардом Хеберном.

Полиалфавитный шифр – это криптосистема, в которой используется несколько моноалфавитных шифров. Поэтому необходимо иметь как минимум 2 таблицы и шифрование текста происходит следующим образом. Первый символ шифруется с помощью первой таблицы, второй символ – с помощью второй таблицы и так далее.

Сильные стороны полиалфавитных шифров заключается в том, что атака по маске и атака частотным криптоанализом здесь не работает, потому что в таких шифрах две разные буквы могут быть зашифрованы одним и тем же символом.

Хорошее решение получается при использование алфавита, содержащего все символы клавиатуры компьютера и перемешенные. Второй алфавит можно составить из иероглифов КНР, третий – из иероглифов Японии и т.д.

В работе рекомендую использовать модуль **random** языка программирования ***python***. random реализует генератор псевдослучайных чисел для различных распределений, включая целые и вещественные числа с плавающей запятой.

```
# Полиалфавитный шифр
import random
from colorama import init
init(autoreset=True)
# создадим класс с нужными цветами
class Bcolors:
        GN = '\033[32m'  # green
        Y    = '\033[93m'  # yellow
        R    = '\033[91m'  # red
# определим модуль действий при выполнении программы
print('')
print(Bcolors.GN + ' ПОЛИАЛФАВИТНЫЙ ШРИФТ')
print('')
print(Bcolors.Y + ' Номер действия')
print('')
print(Bcolors.Y + ' 1) - кодировать')
print(Bcolors.Y + ' 2) - декодировать')
print(Bcolors.Y + ' 3)  - создать новый ключ')
print(Bcolors.Y + ' 4) - справка'+'\n')

# определим 4 алфавита, перемешанные по-разному, получилась длина нашего ключа    до 162
# символов
# encoder
list_encoder = '''%Ёеуъ? 2v#;<\9}иЧЯfA№e4УАЖn|=ёюя`ИLBHM_ЬC0~ZwIHбпr'cbdqdgзB"$Dэlхй,C-
XF{Os(:БТжГ6zшЬJьkppKЕШЮX]uGнЩ[УДQРт)W5N8U.7фхФ/ЛV!чiS1tET3mjo3mг^авЭщРцскКПО&hMыа@улЙоЦ+
>*Ы'''
# encoder2
list_encoder2 = '''%Ёеуъ? =ёюя`ИLBHM_ЬC0~ZwIHбпr'cbdqdgзB"$Dэlхй,C-
XF{Os(:БТжГ6zшЬJьkppKЕШЮX]uGнЩ[УДQРт)W5N8U.7фхФ/ЛV!чiS1tET3mjo3mг^авЭщРцскКПО&hMыа@улЙоЦ+
>*Ы2v#;<\9}иЧЯfA№e4УАЖn|'''
# list_encoder3
list_encoder3 =  '''3mг^авЭщРцскКПО&hMыа@улЙоЦ+>*Ы%Ёеуъ?
2v#;<\9}иЧЯfA№e4УАЖn|=ёюя`ИLBHM_ЬC0~ZwIHбпr'cbdqdgзB"$Dэlхй,C-
XF{Os(:БТжГ6zшЬJьkppKЕШЮX]uGнЩ[УДQРт)W5N8U.7фхФ/ЛV!чiS1tET3mjo'''
# encoder4
list_encoder4 =   '''C-
XF{Os(:БТжГ6zшЬJьkppKЕШЮX]uGнЩ[УДQРт)W5N8U.7фхФ/ЛV!чiS1tET3mjo3mг^авЭщРцскКПО&hMыа@улЙоЦ+
>*Ы%Ёеуъ? 2v#;<\9}иЧЯfA№e4УАЖn|=ёюя`ИLBHM_ЬC0~ZwIHбпr'cbdqdgзB"$Dэlхй,'''
# модуль комментариев к программе, текст комментариев необходимо   поместить в текстовый
# файл readme.txt
def readme():
    try:
        handle = open("readme.txt", "r")
        data = handle.read()
        print(Bcolors.GN + data)
        handle.close()
    except IOError:
        print(Bcolors.R + 'Файл справки не обнаружен!')
# модуль генерации ключа – алгоритм кодирования      исходного  текста следубщий,   если
# порядковый номер символа делится на 3 без остатка, то берём   символ из 4 списка, если
# делится на 2 без остатка, то из 3 списка, иначе   со   2 списка.   И таким  образом мы
# добились, что повторяющихся символов будет очень мало.

# new key
def key():
    w = list_encoder
    q = ''.join(random.sample(w, len(w)))
    print('')
    print(Bcolors.GN + q)
# модуль кодирования
def enc():
```

```
    while True:
        b = input('\n' + 'Введите текст для encoder: '+'\n')
        if b == '':
            print(Bcolors.R + "Ничего не введено!")
            continue

        h = b + list_encoder
        if len(list_encoder) != len(set(h)):
            print(Bcolors.R + 'Символы в ключе не найдены!')
            continue
        print(Bcolors.GN + '\n' + 'Зашифрованный текст:' + '\n')

        i = 1
        for c in b:
            encoder = ''
            y = list_encoder.index(c)
            if i % 3 == 0:
                encoder += list_encoder4[y]
            elif i % 2 == 0:
                encoder += list_encoder3[y]
            else:
                encoder += list_encoder2[y]
            i += 1
            print(encoder, end="")
        break

    # модуль декодирования
def dec():
    while True:
        m = input('\n' + 'Введите текст для decoder: '+'\n')
        if m == '':
            print(Bcolors.R + "Ничего не введено!")
            continue

        t = m + list_encoder2
        if len(list_encoder2) != len(set(t)):
            print(Bcolors.R + 'Символы в ключе не найдены!')
            continue
        print(Bcolors.GN + '\n' + 'Расшифрованный текст:' + '\n')

        i = 1
        for x in m:
            decoder = ''
            if i % 3 == 0:
                y = list_encoder4.index(x)
                decoder += list_encoder[y]
            elif i % 2 == 0:
                y = list_encoder3.index(x)
                decoder += list_encoder[y]
            else:
                y = list_encoder2.index(x)
                decoder += list_encoder[y]
            i += 1
            print(decoder, end="")
               break
# модуль начала работы программы
while True:
    text_vybor = input('Выберите действие:'+'\n')
    if text_vybor == "1":
        enc()
        break
    elif text_vybor == "2":
        dec()
        break
```

```
    elif text_vybor == "3":
        key()
        break
    elif text_vybor == "4":
        readme()
        break
    else:
        print(Bcolors.R + "Введите 1,2,3 или 4!")

print(Bcolors.Y + "\n\nДля выхода нажмите Enter")
input()
```

ЗАДАНИЯ РАБОТЫ

1. Создать проект в среде Visual Studio 2019 с использование языка программирования Python.
2. Сформировать необходимое окружения языка Python из библиотек, необходимых для выполнения лабораторной работы.
3. Создать два файла-программы в языке python для шифровки и дешифровки с разными алфавитами, во втором модуле применить иероглифы.
4. Сформировать тексты программ, в соответствие с методическими указаниями, для шифровки и дешифровки.
5. Подготовить 4 примера – 2 на русском языке и 2 на английском для подготовки шифротекста. Примеры должны содержать правильные тексты (символы алфавита) и ошибочные.
6. Сформировать шифротексты.
7. Выполнить дешифровку шифротекстов.
8. Оформить отчет по работе с указанием описания алгоритма кодирования, описания программ шифровки и дешифровки, описание примеров и указания недостатков и достоинств алгоритма.