МИНОБРНАУКИ РОССИИ

Изображение выглядит как текст

Автоматически созданное описаниеФедеральное государственное бюджетное образовательное учреждение высшего образования

НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ им. Р.Е.АЛЕКСЕЕВА

Институт радиоэлектроники и информационных технологий

Кафедра информатики и систем управления

Отчет

По лабораторной работе №1

«Шифры замены»

По дисциплине

«Защита информации»

РУКОВОДИТЕЛЬ:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Капранов С.Н.

(подпись) (фамилия, и.,о.)

СТУДЕНТ:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Игнаков К. М.

(подпись) (фамилия, и.,о.)

19-ВМ

(шифр группы)

Работа защищена «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_

С оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Нижний Новгород 2022

**Задание №1.5.**

Реализовать алгоритм шифрования данных «Шифр Трисемуса».

**Шифр «Гронсфельда».**

Шифрующие таблицы Трисемуса по принципу шифрования аналогичны полибианскому квадрату. Различия между ними заключаются лишь в принципе заполнения таблиц буквами алфавита.

Таблица Трисемуса заполняется с помощью ключевого слова, повторяющиеся буквы которого отбрасываются. Затем таблица дополняется не вошедшими в нее буквами алфавита по порядку как в системе Цезаря с ключевым словом. Таким образом, ключом в таблицах Трисемуса является ключевое слово и размер таблицы.

При шифровании буква открытого текста заменяется буквой, расположенной ниже нее в том же столбце.

**Алгоритм.**

1. Сначала вводим сообщение, которое будем кодировать/декодировать.

2. Вводится ключ.

3. Выводим ключ.

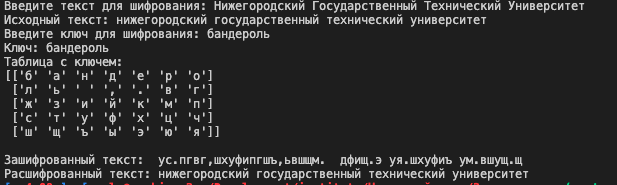
4. Далее происходит само кодирование по правилам кодиорвания Шифра Трисемуса.

5. Затем выводим зашифрованное сообщение.

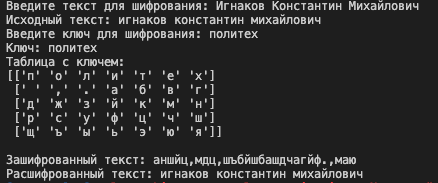
6. Для демонстрации работы декодирования сразу же расшифруем ранее зашифрованное сообщение.

**Примеры шифрования/расшифровывания.**

1. Нижегородский государственный технический университет:



2. Игнаков Константин Михайлович:



**Текст программы.**

import numpy as np

rows = 5

column = 7

russian\_alpabeth = (chr(i) for i in range(ord('а'), ord('я') + 1))

def create\_table(key: str):

start\_words = []

updated\_alpbeth = list(russian\_alpabeth)

updated\_alpbeth.append(' ')

updated\_alpbeth.append('.')

updated\_alpbeth.append(',')

for i in key:

if i not in start\_words:

start\_words.append(i)

updated\_alpbeth.remove(i)

updated\_alpbeth.sort()

updated\_alpbeth = start\_words + updated\_alpbeth

table = np.array(updated\_alpbeth).reshape((rows, column))

table.astype('str')

return table, updated\_alpbeth

def encrypt(text: str, table: list):

answer = ''

for i in text:

new\_index = table.index(i) + column

answer += table[new\_index % (rows \* column)]

return answer

def decrypt(text: str, table: list):

answer = ''

for i in text:

new\_index = table.index(i) - column

if new\_index >= 0:

answer += table[new\_index]

else:

answer += table[rows \* column - abs(new\_index)]

return answer

if \_\_name\_\_ == '\_\_main\_\_':

print('Исходный текст:', text := input("Введите текст для шифрования: ").strip().lower().replace('ё', 'е'))

print('Ключ:', key := input("Введите ключ для шифрования: ").strip().lower().replace('ё', 'е'))

# text = 'Игнаков К.М.'.lower()

# key = 'тест'.lower()

table, slice\_array = create\_table(key)

print(f'Таблица с ключем:\n{table}', end='\n\n')

text\_encrypt = encrypt(text, slice\_array)

print(f"Зашифрованный текст: {text\_encrypt}")

decrypt\_table = slice\_array.copy()

decrypt\_text = decrypt(text\_encrypt, decrypt\_table)

print(f'Расшифрованный текст: {decrypt\_text}')