

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ**

**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**(ДГТУ)**

Факультет Информатика и вычислительная техника

Кафедра Кибербезопасность информационных систем

**Лабораторная работа № 3**

по дисциплине «Криптографические методы защиты информации»

на тему «Реализация алгоритма шифрования Плейфейра»

Выполнил обучающийся гр. ВКБ41

Баскаков П.Е.

(Фамилия, Имя, Отчество)

Проверила:

Доцент Рощина Е.В.

(Должность, Фамилия, Имя, Отчество)

Ростов-на-Дону

2019

**Краткие теоретические сведения**

**1. Как формируется шифрующая таблица для реализации алгоритма Плейфейра?**

В основе алгоритма Плейфейра – использование шифрующей таблицы, формируемой аналогично таблице подстановок Трисемуса. Составной ключ шифрования также включает ключевое слово и размер шифрующей таблицы.

**2. Какие ограничения накладываются на шифруемый текст?**

Шифруемый текст должен иметь четное количество букв, и в нем не должно быть биграмм, содержащих две одинаковые буквы.

**3. Что такое биграмма?**

Одна пара букв открытого текста.

**4. В чем заключается процедура шифрования с помощью алгоритма Плейфейра?**

Последовательность биграмм открытого текста преобразуется с помощью шифрующей таблицы в последовательность биграмм шифртекста по следующим правилам:

а) если обе буквы биграммы открытого текста не попадают на одну строку или столбец, тогда находят буквы в углах прямоугольника, определяемого данной парой букв. Последовательность букв в биграмме шифртекста должна быть зеркально расположенной по отношению к последовательности букв в биграмме открытого текста;

б) если обе буквы биграммы открытого текста принадлежат одному столбцу таблицы, то буквами шифртекста считаются 2 буквы, которые лежат под ними. Если при этом буква открытого текста находится в нижней строке, то для шифртекста берется соответствующая буква из верхней строки того же столбца;

в) если обе буквы биграммы открытого текста принадлежат одной строке таблицы, то буквами шифртекста считаются буквы, которые лежат справа от них. Если при этом буква открытого текста находится в крайнем правом столбце, то для шифра берут соответствующую букву из левого столбца в той же строке.

**Цель работы:** формирование умений шифрования с использованием алгоритма шифрования Плейфейра.

**Ход работы (вариант 2)**

**Задание 1.** Зашифруйте сообщение, используя алгоритм Плейфейра согласно своему варианту. Размер шифрующей таблицы 4 × 8.

Сообщение: Смысл жизни подобен карабканию по канату, который мы же сами подкинули в воздух

Ключевое слово: УСПЕХ

Программное средство было реализовано на языке Python 3.6. Результаты работы показаны на рисунках 1, 2.

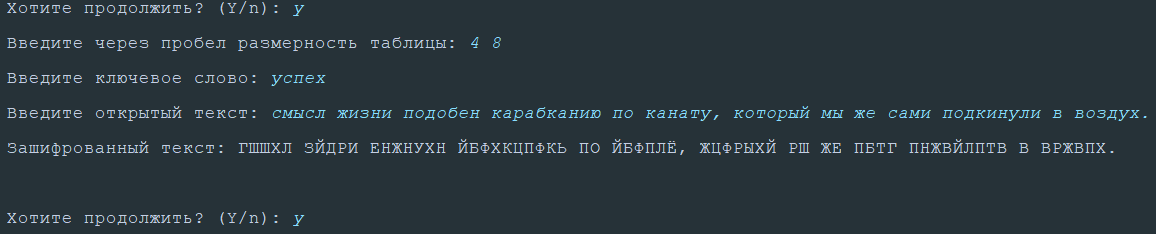


Рисунок 1 – Результат шифрования с таблицей 4x8

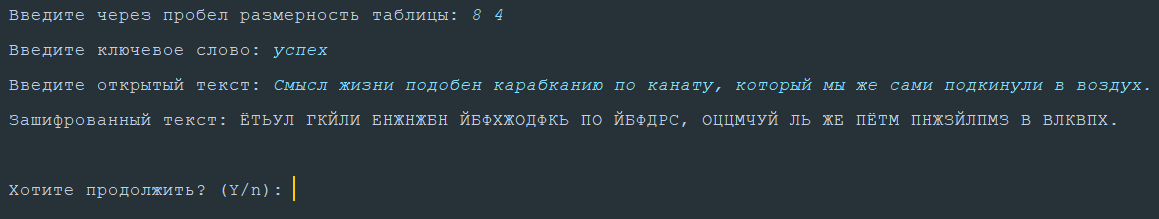


Рисунок 2 – Результат шифрования с таблицей 8x4

Листинг программного средтва:

import locale

from typing import Tuple, List, NoReturn

from collections import OrderedDict

from functools import cmp\_to\_key

from string import digits

from config import ALPHABET

class ШОError(Exception):

pass

class Pos:

def \_\_init\_\_(self, row: int, column: int):

self.row = row

self.column = column

def encrypt(open\_text: str, keyword: str, table\_size: Tuple[int, int]) -> str:

cipher\_text = str()

curr\_pos = 0

cipher\_table = get\_cipher\_table(keyword, table\_size)

while curr\_pos < len(open\_text):

digram = open\_text[curr\_pos:curr\_pos + 2]

if all(map(lambda symb: symb in ALPHABET, digram)):

cipher\_text += encrypt\_one\_digram(digram, cipher\_table)

else:

cipher\_text += digram

curr\_pos += 2

return cipher\_text

def encrypt\_one\_digram(digram: str, cipher\_table: List[List[str]]) -> str:

pos1 = find\_symbol\_in\_table(digram[0], cipher\_table)

pos2 = find\_symbol\_in\_table(digram[1], cipher\_table)

encrypted1 = cipher\_table[pos1.row][pos2.column]

encrypted2 = cipher\_table[pos2.row][pos1.column]

if pos1.row == pos2.row:

encrypted1 = cipher\_table[pos1.row][(pos1.column + 1) % len(cipher\_table[pos1.row])]

encrypted2 = cipher\_table[pos2.row][(pos2.column + 1) % len(cipher\_table[pos2.row])]

if pos1.column == pos2.column:

encrypted1 = cipher\_table[(pos1.row + 1) % len(cipher\_table)][pos1.column]

encrypted2 = cipher\_table[(pos2.row + 1) % len(cipher\_table)][pos2.column]

return encrypted1 + encrypted2

def find\_symbol\_in\_table(symbol: str, cipher\_table: List[List[str]]) -> Pos:

for index\_row, row in enumerate(cipher\_table):

if symbol in row:

return Pos(index\_row, row.index(symbol))

def get\_cipher\_table(keyword: str, table\_size: Tuple[int, int]) -> List[List[str]]:

unique\_key\_sequence = list(OrderedDict.fromkeys(keyword).keys())

other\_alphabet = sorted(set(ALPHABET).difference(set(unique\_key\_sequence)),

key=cmp\_to\_key(locale.strcoll))

new\_alphabet = unique\_key\_sequence + other\_alphabet

truncate\_alphabet\_if\_needed(table\_size, new\_alphabet)

columns = table\_size[1]

curr\_index = 0

cipher\_table = []

while curr\_index < len(new\_alphabet):

cipher\_table.append(new\_alphabet[curr\_index:curr\_index + columns])

curr\_index += columns

return cipher\_table

def truncate\_alphabet\_if\_needed(table\_size: Tuple[int, int], sequence: List[str]) -> NoReturn:

table\_len = table\_size[0] \* table\_size[1]

while len(sequence) > table\_len:

sequence.pop()

def validate\_keyword(keyword: str) -> (NoReturn, str):

if not keyword.isalpha():

raise ШОError("Ключевое слово должно состоять из букв русского алфавита!")

check\_correct\_language(keyword)

return keyword

def validate\_table\_size(user\_input: str) -> (Tuple[int, int], NoReturn):

try:

rows, columns = map(lambda string: int(string), user\_input.split())

except ValueError:

raise ШОError("Некорректный ввод!")

else:

if rows \* columns > len(ALPHABET):

raise ШОError("Размер таблицы не может быть больше длины алфавита!")

return rows, columns

def validate\_open\_text(open\_text: str) -> (str, NoReturn):

if len(open\_text) % 2 != 0:

raise ШОError("Длина открытого текста нечетная!")

if set(open\_text).intersection(set(digits)):

raise ШОError("Открытый текст не должен содержать цифр!")

check\_correct\_language(open\_text)

return open\_text

def check\_correct\_language(text: str) -> NoReturn:

for symbol in text:

if symbol.isalpha() and symbol not in ALPHABET:

raise ШОError("Содержатся символы на постороннем языке. Повторите ввод")

if \_\_name\_\_ == "\_\_main\_\_":

while True:

try:

rows, columns = validate\_table\_size(input("Введите через пробел размерность таблицы: "))

keyword = validate\_keyword(input("Введите ключевое слово: ").upper())

open\_text = validate\_open\_text(input("Введите открытый текст: ").upper())

except ШОError as msg:

print(msg, "\n")

continue

else:

cipher\_text = encrypt(open\_text, keyword, (rows, columns))

print("Зашифрованный текст:", cipher\_text, "\n")

choose = input("Хотите продолжить? (Y/n): ").upper()

if choose == "Y" or choose == "":

continue

break

**Заключение.** В ходе выполнения лабораторной работы были получены навыки шифрования с использованием с использованием алгоритма шифрования Плейфейра. Таким образом, цель работы была достигнута.