Национальный исследовательский университет ИТМО

Факультет программной инженерии и компьютерной техники

Направление программная инженерия
Образовательная программа системное и прикладное программное обеспечение
Специализация системное программное обеспечение

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1.1

курса «Информационная безопасность»

по теме: «Криптографические системы с секретным ключом»

Вариант № 4

Выполнил студент:
Тюрин Иван Николаевич
группа: P33102

Преподаватель:
Маркина Т.А.,
Рыбаков С.Д.

Санкт-Петербург, 2024 г.

Содержание

Лабораторная работа № 1.1. Криптографические системы с секретным ключом **2**
1. Описание 2
2. Выполнение задания 2
3. Вывод 6

Лабораторная работа № 1.1 Криптографические системы с секретным ключом

1. Описание

Лабораторная работа № 1 «Основы шифрования данных».

Цель работы. Изучение основных принципов шифрования информации, знакомство с широко известными алгоритмами шифрования, приобретение навыков их программной реализации.

Задание варианта № 4:

4. Реализовать в программе шифрование и дешифрацию файла с использованием квадрата Полибия, обеспечив его случайное заполнение.

2. Выполнение задания

В соответствии с условием задания разработан скрипт на языке Python выполняющий шифрование текста из файла, имя которого подается на ввод, по методу «Квадрата Полибия». Исходный код скрипта можно видеть на листинге 1.1.

Утилита принимает на вход 4 аргумента:

1. тип операции (encrypt/decrypt),
2. входной файл,
3. выходной файл,
4. название файла с содержанием «ключа» (значения квадрата Полибия).

При шифровании создается случайный ключ и записывается в файл указанный 3 аргументом.

Пример работы утилиты по шифрованию и дешифрованию фрагмента текста из трактата «Lorem ipsum» можно видеть на изображении 1.1.

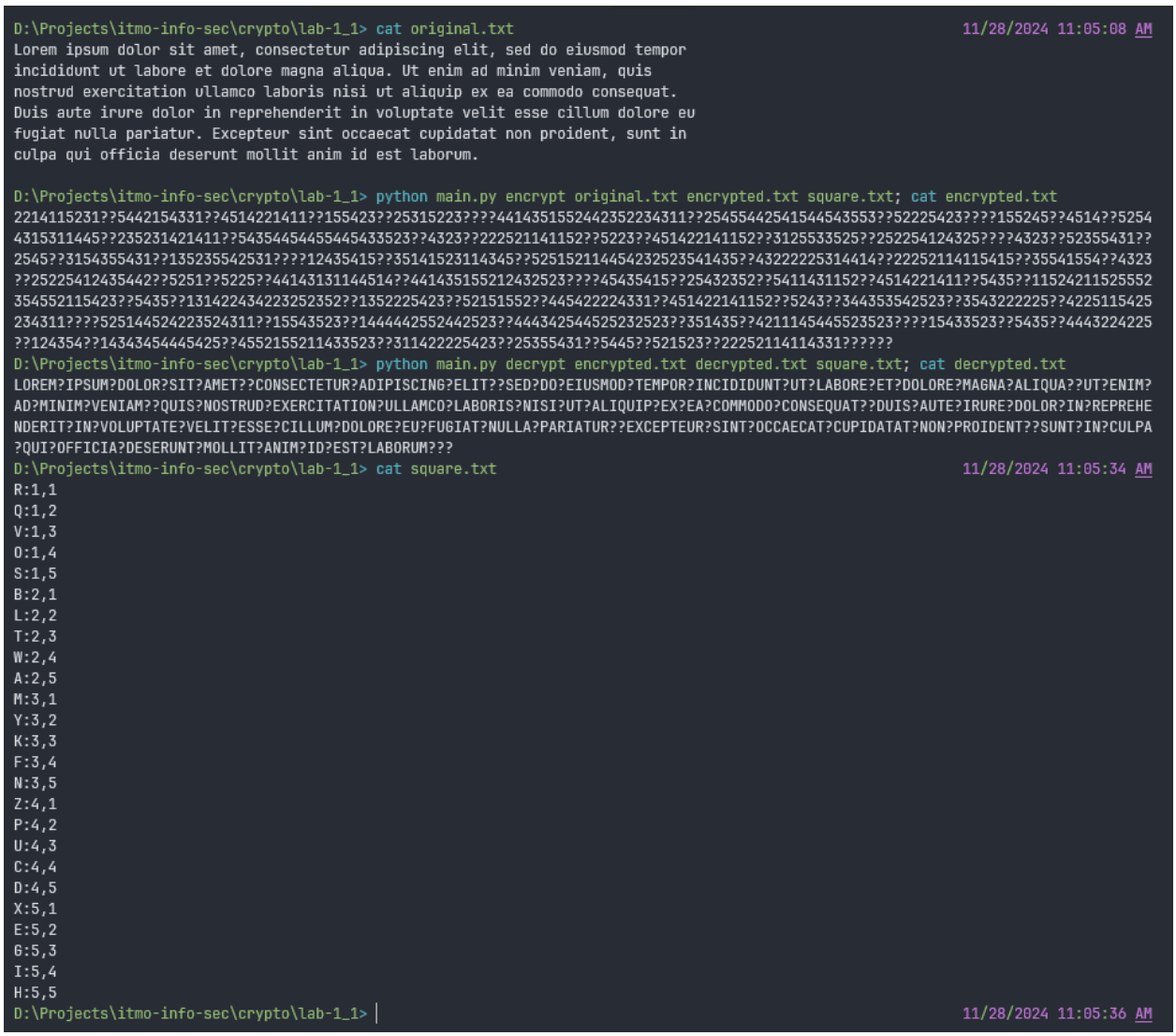

Рис. 1.1: Пример работы утилиты для шифрования фрагмента трактата Lorem ipsum.

```python
#!/usr/bin/env python

# 4. Реализовать в программе шифрование и дешифрацию файла с использованием
#    квадрата Полибия, обеспечив его случайное заполнение.

import random
import string
from typing import Literal
from sys import argv

type PolybiySquare = dict[str, tuple[int, int]]
type Mode = Literal["encrypt", "decrypt"]

```

```
14
15 def generate_polibiy_square() -> PolybiySquare:
16     alphabet = string.ascii_uppercase.replace("J", ""
   )
17     assert len(alphabet) == 25, "alphabet length
   shoud be square number"
18
19     permutation = random.sample(alphabet, len(
   alphabet))
20     square = {permutation[i]: (i // 5 + 1, i % 5 + 1)
    for i in range(len(permutation))}
21     return square
22
23
24 def encrypt(text: str, square: PolybiySquare) -> str:
25     text = text.upper().replace("J", "I")
26     encrypted_text = []
27     for char in text:
28         if char in square:
29             row, col = square[char]
30             encrypted_text.append(f"{row}{col}")
31         else:
32             encrypted_text.append("??")
33     return "".join(encrypted_text)
34
35
36 def decrypt(encrypted_text: str, square:
   PolybiySquare) -> str:
37     reverse_square = {v: k for k, v in square.items()
   }
38     decrypted_text = []
39     for i in range(0, len(encrypted_text), 2):
40         try:
41             row, col = int(encrypted_text[i]), int(
   encrypted_text[i + 1])
42             decrypted_text.append(reverse_square[(row
   , col)])
43         except ValueError:
44             decrypted_text.append("?")
45     return "".join(decrypted_text)
46
47
48 def process_file(
```

```
    input_filename, output_filename, square: PolybiySquare, mode: Mode = "encrypt"
):
    with open(input_filename, "r") as file:
        text = file.read()

    if mode == "encrypt":
        result = encrypt(text, square)
    elif mode == "decrypt":
        result = decrypt(text, square)

    with open(output_filename, "w") as file:
        file.write(result)


def main(
    input_filename: str, output_filename: str, square_data_filename: str, mode: Mode
):
    if mode == "encrypt":
        square = generate_polibiy_square()
        with open(square_data_filename, "w") as file:
            data = "\n".join([f"{k}:{square[k][0]},{square[k][1]}" for k in square])
            file.write(data)
    else:
        with open(square_data_filename, "r") as file:
            square = {}
            for row in file.readlines():
                [char, pos] = row.split(":")
                [i, j] = map(int, pos.split(","))
                square[char] = (i, j)

    process_file(input_filename, output_filename, square, mode=mode)


if __name__ == "__main__":
    if argv[1] == "encrypt":
        mode = "encrypt"
    elif argv[1] == "decrypt":
        mode = "decrypt"
    else:
```

```
88          raise KeyError("first argument specify wether
    'encrypt' or 'decrypt' text")
89
90      main(
91          input_filename=argv[2],
92          output_filename=argv[3],
93          square_data_filename=argv[4],
94          mode=mode,
95      )
```

Листинг 1.1: Код скрипта выполняющего кодирование и декодирование текста методом «квадрата Полибия»

3. Вывод

В результате выполнения работы были изучены методы шифрования с секретным ключом, в частности с помощью квадрата Полибия.