МИНИСТЕРСТВО ПРОСВЕЩЕНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО».

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1
ОСНОВЫ ШИФРОВАНИЯ ДАННЫХ
ВАРИАНТ 11

выполнил	Студент 4 курса факультета Программной инженерии и компьютерной техники Группы P34121 Кустарев Иван Павлович
преподаватель	Старший преподаватель (квалификационная категория "старший преподаватель"), факультета программной инженерии и компьютерной техники Маркина Татьяна Анатольевна

Санкт-Петербург
2023 г.

Название и цель работы

Название: Основы шифрования данных

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

Вариант задания

Вариант = 11 %10 = 1. Реализовать в программе шифрование и дешифрацию содержимого файла по методу Цезаря с ключевым словом.

Листинг разработанной программы с комментариями

```
import string



# язык задаётся путём ввода русского/английского keyword и body



# загрузка данных (секретное слово,
# индекс смещения, режим (шифрование/дешифрование),
# текст для шифровки/дешифровки) из файла
def read_file(file_name):
    f = open(file_name, encoding='utf-8', mode='r')
    try:
        file_content = f.readlines()
        for i in range(0, len(file_content)):
            line = file_content[i].strip()
            # file_content[i] = line.split("=")[1]
        # чтение ключевого слова
        key_word = file_content[0].strip()
        # чтение индекса смещения
        key_index = file_content[1].strip()
        # чтение режима шифровака (расшифровка)
        mood = file_content[2].strip()
        # чтение текста, который нужно зашифровать (расшифровать)
        body = ''
        for i in range(3, len(file_content)):
            body += (file_content[i] + '\r')
```

```
        try:
            key_index = int(key_index)
            if mood.upper() == "TRUE":
                mood = True
            else:
                mood = False
        # валидация данных из файла на приведение типов и тп
        except TypeError:
            raise Exception("Invalid typing")
        if isinstance(key_word, str) and isinstance(body, str):
            return key_word, key_index, mood, body
        else:
            raise Exception("Invalid typing")
    except FileNotFoundError:
        raise Exception("File not found")
    except IndexError:
        raise Exception("Invalid file structure")
    finally:
        f.close()


# проверка на то, что символ принадлежит алфавиту (рус/англ).
# знаки препинания не являются частями алфавитов (https://clck.ru/35sqmi)
def is_alpha(word: str, canonical_alpha: str):
    word = word.upper()
    for char in word:
        if not (char in canonical_alpha):
            return False
    return True


# валидация ключевого слова
def is_correct_key_word(key_word: str, canonical_alpha: str):
    if not is_alpha(key_word, canonical_alpha):
        raise Exception("Keyword has not alpha symbols")
    if not key_word.isupper():
        raise Exception("Keyword is not in uppercase")
    for char in key_word:
        if key_word.count(char) > 1:
            raise Exception("The keyword contains repeated characters")
    if len(key_word) > len(canonical_alpha) - 2 or len(key_word) < 0:
        raise Exception("Keyword len should be in 0..." + str(len(canonical_alpha) - 2))
```

```
# валидация индекса сдвига
def is_correct_key_index(key_index: int, canonical_alpha: str):
    if key_index > len(canonical_alpha) - 2 or key_index < 0:
        raise Exception("Key_index len should be in 0..." + str(len(canonical_alpha) - 2))


# создание алфавита сдвигов
def create_shifts_alpha(key_word: str, canonical_alpha: str):
    alpha_list = canonical_alpha
    for i in key_word:
        alpha_list = alpha_list.replace(i, "")
    return key_word + alpha_list


# получение зашифрованного символа на основе входящего символа в верхнем регистре
def get_encrypted_char(alpha_list, upper_char, key_index: int, canonical_alpha: str):
    return alpha_list[(ord(upper_char) - ord(canonical_alpha[0]) - key_index)]


# получение расшифрованного символа на основе входящего символа в верхнем регистре
def get_decrypted_char(alpha_list: str, upper_char, key_index: int, canonical_alpha: str):
    return canonical_alpha[(alpha_list.rfind(upper_char) + key_index) % len(canonical_alpha)]


# функция для шифровки/дешифровки
def encryption_decryption(encryption: bool, key_word: str,
                          key_index: int, body: str, canonical_alpha: str):
    # валидация секретного слова и индекса смещения
    is_correct_key_word(key_word, canonical_alpha)
    is_correct_key_index(key_index, canonical_alpha)

    alpha_list = create_shifts_alpha(key_word, canonical_alpha)

    result = ""

    for char in body:
        # итерация по символам текста и
        # шифровка/дешифровка алфавитных символов + пропуск неалфавитных символов
        if is_alpha(char, canonical_alpha):
            if char.islower():
```

```
            # если символ - в lowercase
            char = char.upper()
            if encryption:
                char = get_encrypted_char(alpha_list, char, key_index, canonical_alpha)
            else:
                char = get_decrypted_char(alpha_list, char, key_index, canonical_alpha)
            char = char.lower()
        else:
            # если символ - в uppercase
            if encryption:
                char = get_encrypted_char(alpha_list, char, key_index, canonical_alpha)
            else:
                char = get_decrypted_char(alpha_list, char, key_index, canonical_alpha)

        result += char
    else:
        result += char
  return result


def cesar_encryption(key_word: str, key_index: int, text_body: str, canonical_alpha: str):
  return encryption_decryption(True, key_word, key_index, text_body, canonical_alpha)


def cesar_decryption(key_word: str, key_index: int, encrypted_body: str, canonical_alpha: str):
  return encryption_decryption(False, key_word, key_index, encrypted_body, canonical_alpha)


# основной процесс
def main_fun(path: str):
  # загрузка данных (секретное слово, индекс смещения, режим (шифрование/дешифрование),
  # текст для шифровки/дешифровки) из файла
  key_word, k, mood, body = read_file(path)
  # выбор алфавита (англ/русский)
  if body[0] in string.ascii_uppercase:
      canonical_alpha = string.ascii_uppercase
  else:
      a = ord('a')
      canonical_alpha = ''.join([chr(i).upper() for i in range(a, a + 32)])
  if key_word and k and body:
      # запуск шифровки/дешифровки
      if mood is True:
```

```
            response = cesar_encryption(key_word, k, body, canonical_alpha)
        else:
            response = cesar_decryption(key_word, k, body, canonical_alpha)
        print(response)


if __name__ == '__main__':
    file_path = input("Введите путь к файлу или Enter чтобы задать file.txt")
    if file_path == "":
        file_path = "file.txt"
    try:
        main_fun(file_path)
    except Exception as e:
        print(e)
```

Результаты работы программы

1. Шифрование (Английский):

1.1 Входные данные:

keyword=ADZKIH

key_index=14

mod=true

body=Interdum dictum. Sed ex. Morbi faucibus. Nisi in vitae vitae mattis nec justo mattis sed leo, luctus et ultricies. Nulla sapien in molestie sed aene-an tempus ultricies. Ultricies. Sit habitasse ut. Adipiscing dictum quis, arcu cursus nulla pellentesque cursus libero, amet, cras orci, sit eleifend qu

1.2 Выходные данные:

Tyhpkobx otnhbx. Ipo pf. Xakmt qlbntmbi. Ytit ty cthlp cthlp xlhhti ypn ubiha xlhhti ipo wpa, wbnhbi ph bwhktntpi. Ybwwl ildtpy ty xawpihtp ipo lpyp-ly

hpxdbi bwhktntpi. Bwhktntpi. Ith slmthliip bh. Lotdtintyr otnhbx zbti, lknb nbkibi ybwwl dpwwpyhpizbp nbkibi wtmpka, lxph, nkli aknt, ith pwptqpyo zb

2. Расшифровка (Английский):

2.1 Входные данные:

keyword=ADZKIH

key_index=14

mod=false

body=Tyhpkobx otnhbx. Ipo pf. Xakmt qlbntmbi. Ytit ty cthlp cthlp xlhhti ypn ubiha xlhhti ipo wpa, wbnhbi ph bwhktntpi. Ybwwl ildtpy ty xawpihtp ipo lpyp-ly hpxdbi bwhktntpi. Bwhktntpi. Ith slmthliip bh. Lotdtintyr otnhbx zbti, lknb nbkibi ybwwl dpwwpyhpizbp nbkibi wtmpka, lxph, nkli aknt, ith pwptqpyo zb

2.2 Выходные данные:

Interdum dictum. Sed ex. Morbi faucibus. Nisi in vitae vitae mattis nec justo mattis sed leo, luctus et ultricies. Nulla sapien in molestie sed aene-an tempus

ultricies. Ultricies. Sit habitasse ut. Adipiscing dictum quis, arcu cursus nulla pellentesque cursus libero, amet, cras orci, sit eleifend qu

3. Шифрование (Русский):

3.1 Входные данные:

keyword=АБВЯГ

key_index=14

mod=true

body=Поставленных эксперимент что интересный образом играет соображения анализа занимаемых идейные и развития. Организационной рамки же важные по важную структура сфера анализа эксперимент модель способствует сфера в соображения разработке место особенности способствует задач. Количественный также же же

3.2 Выходные данные:

Баягсуьцююмж оыябцвщэцюг ига щюгцвцяюмъ атвсшаэ щфвсцг яаатвсчцющр сюсьщшс шсющэсцэмж щхцъюмц щ всшущгщр. Авфсющшсзщаююаъ всэыщ чц усчюмц ба усчюдп ягвдыгдвс яецвс сю-

сьщшс оыябцвщэцюг эахцьн ябаяатягудцг яецвс у яаатвсчцющр всшвстагыц эцяга аяатцююаягщ ябаяатягудцг шсхси. Ыаьщицягуцююмъ гсычц чщ чщ

4. Расшифровка (Русский):

4.1 Входные данные:

keyword=АБВЯГ

key_index=14

mod=false

body=Баягсуьцююмж оыябцвщэцюг ига щюгцвцяюмъ атвсшаэ щфвсцг яаатвсчцющр сюсьщшс шсющэсцэмж щхцъюмц щ всшущгщр. Авфсющшсзщаююаъ всэыщ чц усчюмц ба усчюдп ягвдыгдвс яецвс сюсьщшс оыябцвщэцюг эахцьн ябаяатягудцг яецвс у яаатвсчцющр всшвстагыц эцяга аяатцююаягщ ябаяатягудцг шсхси. Ыаьщицягуцююмъ гсычц чщ чщ

4.2 Выходные данные:

Поставленных эксперимент что интересный образом играет соображения анализа занимаемых идейные и развития. Организационной рамки же важные по важную структура сфера анализа эксперимент модель способствует сфера в соображения разработке место особенности способствует задач. Количественный также же же

Вывод

Во время выполнения данной лабораторной работы я:

1. Получил практические навыки написания программ для шифрования/расшифровки текста методом Цезаря с ключевым словом
2. Ознакомился с различными видами реализации данного метода шифрования
3. Выяснил какие ограничения налагаются на контрольное слово и индекс сдвига при реализации алгоритма