ФГБОУ ВО ‘‘Чувашский государственный университет им. И. Н. Ульянова’’.

Факультет: Информатики и вычислительной техники.

Кафедра: Вычислительной техники.

Средства защиты информации

Лабораторная работа №2

Выполнил: студент группы ИВТ-41-18

Сергеев Давид Евгеньевич

Проверил: доцент Ковалев Сергей Васильевич

Чебоксары 2021 г.

Задание:

Целью работы является знакомство с классическим криптографическим алгоритмом - алгоритмом шифрования данных при помощи перестановки.

Решение:

Каждое преобразование шифра перестановки, предназначенное для зашифрования сообщения длиной n символов, можно задать с помощью следующей таблицы:

$$\begin{pmatrix} 1 & 2 & \dots & n \\ i_1 & i_2 & \dots & i_n \end{pmatrix}$$

где in - номер места шифртекста, на которое перемещается буква под номером n исходного сообщения при выбранном преобразовании. В верхней строке таблицы выписаны по порядку числа от 1 до n, а в нижней - те же числа, но в произвольном порядке. Такая таблица называется подстановкой степени n. Зная подстановку, задающую преобразование, можно осуществить как шифрование, так и дешифрование текста.

Примеры шифров перестановки:

1. Шифр Сцитало
2. Шифр маршрутной перестановки
3. Шифр вертикальной перестановки

Предлагаю в качестве примера разобрать работу на 31 символьное сообщение. Для начала необходимо рассмотреть условия моего варианта.
V = {A..Z, a..z} (алфавит)
m = 18 (длина блока шифрограммы)
Для наглядности я составил таблицу, разворачивающую шифрограмму. Таким образом таблца выглядит следующим образом:

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
i_n	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1

При сообщении: «Luke Skywalker I am your Father». Таблица выше будет выглядеть следующим образом:

Сообщение №1: «Luke_Skywalker_I_a» -> « a_I_reklawykS_ekuL»

n	L	u	k	e	_	S	k	y	w	a	l	k	e	r	_	I	_	a
i_n	a	_	I	_	r	e	k	l	a	w	y	k	S	_	e	k	u	L

Сообщение №2: «m_your_Father____» -> «____rehtaF_ruoy_m»

n	m	_	y	o	u	r	_	F	a	t	h	e	r	_	_	_	_	_
i_n	_	_	_	_	_	r	e	h	t	a	F	_	r	u	o	y	_	m

Текст программы:

```
ALPHABET = [
    'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N',
    'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z', 'a', 'b',
    'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p',
    'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z'
]

ALPHABET_STRAIGHT = {
    0: 17, 1: 16, 2: 15, 3: 14, 4: 13, 5: 12, 6: 11, 7: 10, 8: 9, 9: 8,
    10: 7, 11: 6, 12: 5, 13: 4, 14: 3, 15: 2, 16: 1, 17: 0
}

ALPHABET_REVERSE = {
    17: 0, 16: 1, 15: 2, 14: 3, 13: 4, 12: 5, 11: 6, 10: 7, 9: 8, 8: 9,
    7: 10, 6: 11, 5: 12, 4: 13, 3: 14, 2: 15, 1: 16, 0: 17
}


CIPHER_MSG_BLOCK_SIZE = 18
SPACE_REPLACEMENT = '_'


def checker(msg: str) -> None or ValueError:
    if not set(msg).issubset(set(ALPHABET)):
        difference = set(msg) - set(ALPHABET)
        raise ValueError(f'Symbol not in alphabet: {difference}')


def input_data_handling() -> list:
    input_data_handling.message = input("Введите сообщение: ")
    message = input_data_handling.message

    global ALPHABET, ALPHABET_REVERSE, ALPHABET_STRAIGHT
    ALPHABET.append(SPACE_REPLACEMENT)  # space -> '_'
    message.replace(' ', SPACE_REPLACEMENT)

    msg_blocks = []
    word = ''
    msg_len = 0
    for idx in range(len(message)):
        if message[idx] != ' ':
            word += message[idx]
        else:
            word += SPACE_REPLACEMENT
        msg_len += 1

        if idx == len(message) - 1 and len(word) != CIPHER_MSG_BLOCK_SIZE or
len(word) == CIPHER_MSG_BLOCK_SIZE:
            len_delta = CIPHER_MSG_BLOCK_SIZE - len(word)
            for x in range(len_delta):
                word += SPACE_REPLACEMENT
            checker(word)
            msg_blocks.append(word)
            word = ''
            msg_len = 0
    return msg_blocks
```

```
def cipher(msg: str, encrypted=True):
    if encrypted:
        return encrypt(msg)
    else:
        return decrypt(msg)


def encrypt(msg: str):
    encrypted = '*' * CIPHER_MSG_BLOCK_SIZE

    for idx in range(len(msg)):
        encryption_idx: int = ALPHABET_STRAIGHT[idx]
        encrypted = encrypted[:encryption_idx] + msg[idx] +
encrypted[encryption_idx+1:]
    return encrypted


def decrypt(msg: str):
    decrypted = '*' * CIPHER_MSG_BLOCK_SIZE

    for idx in range(len(msg)):
        decryption_idx = ALPHABET_REVERSE[idx]
        decrypted = decrypted[:decryption_idx] + msg[idx] +
decrypted[decryption_idx+1:]
    return decrypted


def main():
    msg_blocks = input_data_handling()
    encrypted = [cipher(el, encrypted=True) for el in msg_blocks]
    decrypted = [cipher(el, encrypted=False) for el in encrypted]
    print(f'encrypted: {encrypted}')
    print(f'decrypted: {decrypted}')


if __name__ == '__main__':
    main()
```

Пример работы программы:

```
Введите сообщение: Luke Skywalker I am your Father
encrypted: ['a_I_reklawykS_ekuL', '_____rehtaF_ruoy_m']
decrypted: ['Luke Skywalker I a', 'm your Father     ']

Process finished with exit code 0
```