Tugas 3 Praktikum
Kriptografi
Kelas A Semester Ganjil 2021/2022

Oleh:
WAFI FAHRUZZAMAN – 140810200009

PROGRAM STUDI S-1 TEKNIK INFORMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS PADJADJARAN
2022

helper.py

Berisi fungsi-fungsi untuk membantu :

- Konversi karakter ke angka
- Konversi angka ke karakter
- Modulus inverse
- Input text
- Input key hill cipher (supaya dari list dapat menjadi matrix)
- Mencari determinan dan divalidasi supaya Determinan key harus ganjil dan selain 13
- Fungsi Inverse matrix

```
import enum
import numpy as np

# helper general #

class PROCESS(enum.Enum):
    encrypt = 'encrypt'
    decrypt = 'decrypt'

def char_to_num(x):
    x = ord(x)-65
    return x

def num_to_char(x):
    x = chr(x+65)
    return x

def mod_inverse(A, M):
    for X in range(1, M):
        if (((A % M) * (X % M)) % M == 1):
            return X
    return -1

def input_text(string):
    text = input("Input " + string + ": ")
    text = text.replace(' ', '').upper()

    return text
```

```
# helper hill cipher #
def input_key(n):
    key = list(map(int, input("Masukkan nilai key matrix (dipisahkan
koma): ").split(",")))
    key = np.array(key).reshape(n, n) % 26

    print("Matrix Key: ")
    print(key)

    return key

def vector(text, m):
    text_in_number = list(map(char_to_num, list(text)))
    return np.array(text_in_number).reshape(int(len(text)/m), m)

def determinan_with_validation(matrix):
    determinan = int(np.linalg.det(matrix))
    if determinan % 2 == 0 or determinan == 13 :
        print("Determinan key harus ganjil dan selain 13!")
        exit()

    return determinan

def inverse(determinan, matrix):
    return (mod_inverse(determinan % 26, 26) * np.round(determinan *
np.linalg.inv(matrix)).astype(int) % 26)
```

hill_cipher.py

Berisi :

- Menu untuk pilihan enkiripsi, dekripsi, dan cari key
- Kemudian input ukuran matrix key (n x n), input key, dan input text
- Apabila pilih 1 atau 2 maka akan masuk ke fungsi hill_cipher
    - Di dalam fungsi hill cipher:
        - Mencari determinan dan divalidasi: Determinan key harus ganjil dan selain 13
        - Mengonversi plaintext menjadi number
        - Menyusun matrix text
        - Apabila prosesnya dekripsi maka key akan dicari modular inverse
        - Kemudian dilakukan proses perkalian matrix text dengan matrix key sehingga didapatkan hasil enkripsi atau dekripsi
- Apabila pilih 1 maka akan masuk ke fungsi solve_key

- Mengonversi plaintext dan ciphertext ke bentuk matrix
- Kemudian mencari key nya dengan mengalikan matrix ciphertext dengan modular invers matrix plaintext

```
'''
Nama         : Wafi Fahruzzaman
NPM          : 140810200009
Deskirosi    : Tugas 3 - Program Hill Cipher
'''

import helper as helper
import numpy as np

def hill_cipher(process, text, key, n):
    determinan_key = helper.determinan_with_validation(key)

    if(len(text) % n != 0) :
        last_char = text[-1]
        text = last_char*(n - len(text) % n)

    text_vector = helper.vector(text, n)
    result = np.array([], dtype=int)

    if process == helper.PROCESS.decrypt:
        key = helper.inverse(determinan_key, key)

    for i in range(len(text_vector)):
            temp = np.matmul(key, text_vector[i].reshape(n, 1)) % 26
            result = np.append(result, temp)

    result = list(map(helper.num_to_char, result))

    output = ''.join(result)

    return output

def solve_key(plaintext, ciphertext, m):
    p_vector = helper.vector(plaintext, m)
    p_matrix = np.array([], dtype=int)

    c_vector = helper.vector(ciphertext, m)
    c_matrix = np.array([], dtype=int)

    for i in range(m):
        c_matrix = np.append(c_matrix, c_vector[i])
```

```
        p_matrix = np.append(p_matrix, p_vector[i])

    c_matrix = np.transpose(c_matrix.reshape(m,m))
    p_matrix = np.transpose(p_matrix.reshape(m,m))

    p_det = helper.determinan_with_validation(p_matrix)
    p_inverse = helper.inverse(p_det, p_matrix)

    result = np.matmul(c_matrix, p_inverse) % 26
    return result


while True :
    print("\n|--- Menu ---|")
    print("1) Enkripsi\n2) Dekripsi\n3) Cari Key\n4) Exit")
    menu = input("\nPilihan\t: ")

    if menu == '1' or menu == '2':
        ukuran_matrix = int(input("\nInput ukuran (n) matrix key [n x
n]: "))
        key = helper.input_key(ukuran_matrix)

        text = helper.input_text("text")
        if(len(text) < ukuran_matrix):
            print("terjadi ketidaksesuaian antara ukuran matrix key dan
jumlah karakter")
            exit()

        if(menu == '1'):
            print("\n== Enkripsi ==")
            print("Plaintext\t: " + text)
            print("Ciphertext\t: " + hill_cipher(helper.PROCESS.encrypt,
text, key, ukuran_matrix))
        elif menu == '2':
            print("\n== Dekripsi ==")
            print("Ciphertext\t: " + text)
            print("Plaintext\t: " + hill_cipher(helper.PROCESS.decrypt,
text, key, ukuran_matrix))

    elif menu == '3':
        print("\n")
        plaintext = helper.input_text("plaintext")
        ciphertext = helper.input_text("ciphertext")
        m = int(input("\nInput nilai m: "))
```

```
        print("\nPlaintext: " + plaintext + "\nCiphertext: " +
ciphertext)
        key = solve_key(plaintext, ciphertext, m)

        print("key:")
        print(key)

    elif menu == '4':
        exit()

    else :
        print("\nError Input.\n")
```

```
|--- Menu ---|
1) Enkripsi
2) Dekripsi
3) Cari Key
4) Exit

Pilihan : 2

Input ukuran (n) matrix key [n x n]: 3
Masukkan nilai key matrix (dipisahkan koma): 3, 4, 6, 21, 15, 14, 20, 23, 5
Matrix Key:
[[ 3  4  6]
 [21 15 14]
 [20 23  5]]
Input text: SMXSJPBGHTQJKST

== Dekripsi ==
Ciphertext      : SMXSJPBGHTQJKST
Plaintext       : WAFIFAHRUZZAMAN
```

```
|--- Menu ---|
1) Enkripsi
2) Dekripsi
3) Cari Key
4) Exit

Pilihan : 1

Input ukuran (n) matrix key [n x n]: 3
Masukkan nilai key matrix (dipisahkan koma): 3, 4, 6, 21, 15, 14, 20, 23, 5
Matrix Key:
[[ 3  4  6]
 [21 15 14]
 [20 23  5]]
Input text: WAFIFAHRUZZAMAN

== Enkripsi ==
Plaintext       : WAFIFAHRUZZAMAN
Ciphertext      : SMXSJPBGHTQJKST
```

```
|--- Menu ---|
1) Enkripsi
2) Dekripsi
3) Cari Key
4) Exit

Pilihan : 3


Input plaintext: WAFIFAHRUZZAMAN
Input ciphertext: SMXSJPBGHTQJKST

Input nilai m: 3

Plaintext: WAFIFAHRUZZAMAN
Ciphertext: SMXSJPBGHTQJKST
key:
[[ 3  4  6]
 [21 15 14]
 [20 23  5]]
```