Pemrograman Berbasis Fungsi – RA

TA Genap 2021/2022

Lecturer : Riksa Meidy Karim, S.Kom., M.Si., M.Sc.

NAMA : Muhammad Rivan Hasri

NIM : 120450079

Tugas Exercise

>> Exercise 1 >>

1. Output yang dihasilkan dari input password anakanakcerdas2020 adalah Sc+TV+Sc+TS-Sc+TV+Sc+TS-Se+Sg+TZ+Sf+Sc+T[+Qh+Qf+Qh+Qf+. Hasil tangkapan layar code

```
limitPassword = 100
while True:
    opsi = input('Do you want to encrypt a password ? [Y/N] ')
    if (opsi == 'Y') or (opsi == 'y'):
        password = input('Enter your password: ')
        cek_panjangPassword(password,limitPassword)
        print(password, '------->', encrypted(password))
    else:
        break
```

```
Do you want to encrypt a password ? [Y/N] y
Enter your password: anakanakcerdas2020
anakanakcerdas2020 -------> Sc+TV+Sc+TS-Sc+TV+Sc+TS-Se+Sg+TZ+Sf+Sc
+T[+Qh+Qf+Qh+Qf+
Do you want to encrypt a password ? [Y/N] n
```

2. Hasil tangkapan layar code:

```
while True:
    opsi = input('Do you want to decrypt a password ? [Y/N] ')
    if (opsi == 'Y') or (opsi == 'y'):
        password = input('Enter your password: ')
        print(password, '------->', decrypted(password))
    else:
        break
```

```
Do you want to decrypt a password ? [Y/N] y
Enter your password: Sc-TV-Sc-TS+T[-Sc-TQ+TV-T[-Sf-Sc-T\-Sc-Qh-Qf-
Qh-Qf-TS+Sg-Se-Sg-
Sc-TV-Sc-TS+T[-Sc-TQ+TV-T[-Sf-Sc-T\-Sc-Qh-Qf-Qh-Qf-TS+Sg-Se-Sg- --
-----> anaksainsdata2020kece
Do you want to decrypt a password ? [Y/N] n
```

>> Exercise 2 >>

1. Hasil tangkapan layar code:

```
import numpy as np

# Create txt file for first number and second number
with open('number.txt','w') as f:
    f.write('950256169485865215028174799410854594365152121509684199523704038449874080399346937660203134161958576 3')
with open('number1.txt','w') as f:
    f.write('211606864269616293496578908053099280539190056897895849620155585583389683337229550780393624318706109 2')
```

```
# Open file
f1 = open('number.txt').read()
f2 = open('number1.txt').read()

# Add the first number and the second number with anonymous function
add = lambda x,y: int(x) + int(y)
add(f1,f2)
```

```
116186303375548150852475370746395387490434217840758004914385962403326376373657648844059675848066468 55
```

>> Exercise 3 >>

1. Modul solver.py dapat digunakan untuk menyelesaikan persamaan differensial orde 2 selain kasus non linear pendulum, namun yang perlu diperhatikan ialah fungsi Func yang akan selalu berbeda sesuai kasus yang ingin kita solve.

2. Hasil tangkapan layar

3. Hasil screenshot code:

```
import math as m

def Func(g,L,a):
    return (-(g/L))*m.sin(a)

def euler(t,h,y,dy,Func):
    d2y = Func(t,y,dy)
    y_next = y + (h * dy)
    dy_next = dy + (h * d2y)
    return (y_next, dy_next)

def cauchy_euler(params, Func):
    # initial condition
    t0 = params['t0']
    t_akhir = params['t_akhir']
    h = params['h']
    y0 = params['y0']
```

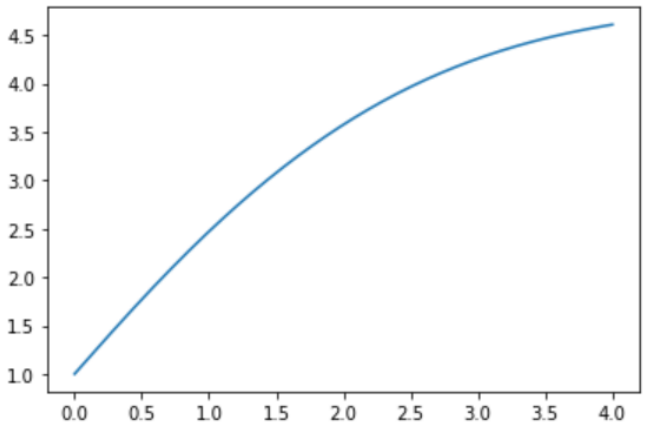
4. Hasil screenshot:

```
from solver import *
import matplotlib.pyplot as plt

params = {
    'g' : 9.8,
    'y0' : 1,
    't0' : 0,
    't_akhir' : 4,
    'h' : 0.001,
    'dy0' : 0.5 * 3.14
}

t, res_euler = cauchy_euler(params, Func)

plt.plot(t, res_euler)
plt.show()
```



5. Hasil screenshot

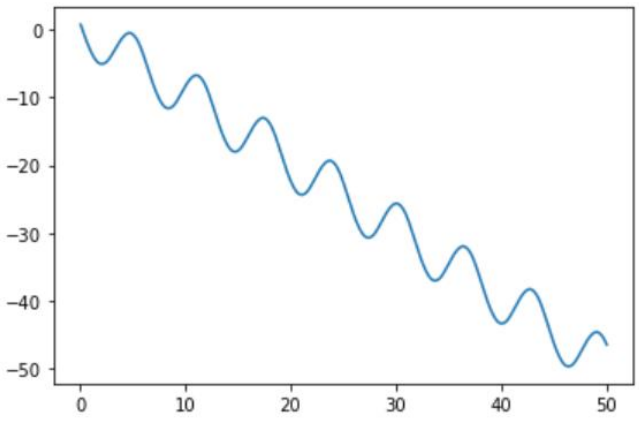
```
from solver import *
import math
from matplotlib import pyplot as plt

def pd(y,dy,x):
    return -y - dy + (0.5*(1 - math.cos(2*x)))

parameter = {
    't0' : 0,
    't_akhir' : 50,
    'h' : 0.05,
    'y0' : 1,
    'dy0' : -9/2
}

t2, res_euler2 = cauchy_euler(parameter, pd)

plt.plot(t2,res_euler2)
plt.show()
```



>> Exercise 4 >>

1.

>> Exercise 7 >>

1. Hasil screenshot

```
P = 'akulupa'
P = map(lambda x: ((x[0] * 2) + 1, x[1]), enumerate(P))

print(list(P))
```

```
[(1, 'a'), (3, 'k'), (5, 'u'), (7, 'l'), (9, 'u'), (11,
'p'), (13, 'a')]
```

2. Hasil screenshot

```
B = 24

fks = map(lambda b: b+1 if B % (b+1) == 0 else -1, range(B))

def filt(fks):
    x = []
    for f in fks:
        if f != -1:
            x.append(f)
    return x

print(filt(fks))
```

```
[1, 2, 3, 4, 6, 8, 12, 24]
```

3. Hasil screenshot

```
A = [[3,4], [5,6]]
B = [[1,2], [7,8]]

C = list(map(lambda ra,rb: list(map(lambda raa,rbb: raa+rbb, ra, rb)), A, B))

def dett(C):
    return (C[0][0] * C[1][1]) - (C[0][1] * C[1][0])

print(dett(C))
```

```
-16
```