



Pemrograman Berbasis Fungsi - RA TA Genap 2021/2022

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

NAMA : Dimas Wahyu Saputro

NIM : 120450081

Tugas Exercise

>> Exercise 1 >>

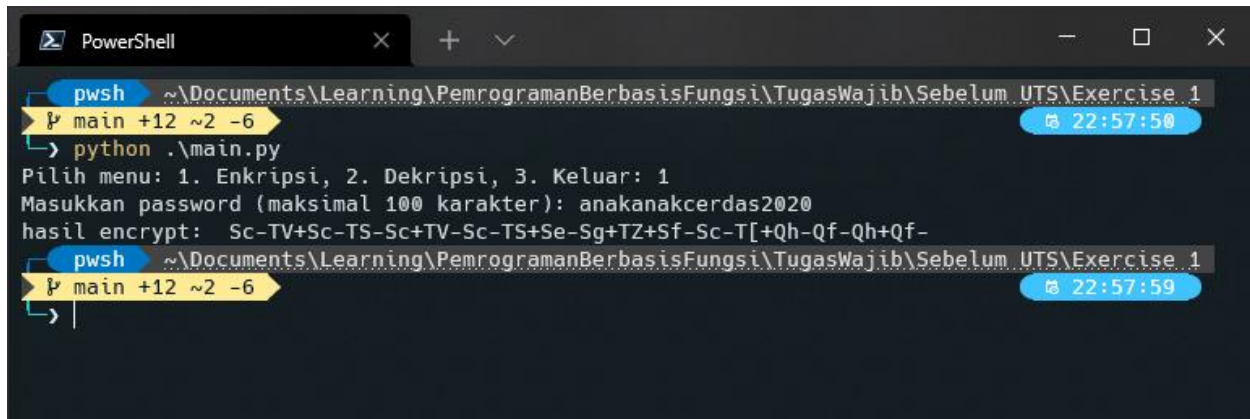
1.a. Bantulah user tersebut dengan membuatkan sebuah program yang secara otomatis mengubah password yang di input menjadi terenkripsi sesuai aturan tersebut!

```
main.py
1 import tools
2
3 pilihan = input('Pilih menu: 1. Enkripsi, 2. Dekripsi, 3. Keluar: ')
4 if pilihan == '1':
5     password = input('Masukkan password (maksimal 100 karakter): ')
6     password_final = tools.enc(password)
7     print('Hasil encrypt: ', password_final)
8 elif pilihan == '2':
9     password = input('Masukkan password yang ingin didekripsi: ')
10    password_final = tools.dec(password)
11    print('Hasil decrypt: ', password_final)
12 else:
13    print('Terima kasih')

tools.py
1 # author : Dimas Wahyu Saputro
2 # NIM : 120450081
3 # Affiliation : Sains Data ITERA
4 # Date : 04 March 2022
5 # Program Description : Program to solve simple encryption password problem
6
7 def enc(password):
8     password_list = list(password)
9     temp_password = []
10    for i in range(len(password_list)):
11        temp_password.append(chr(((ord(password_list[i]))//26) + 80))
12        temp_password.append(chr(((ord(password_list[i]))%26) + 80))
13        if temp_password[i] >= temp_password[i+1]:
14            temp_password.append('*')
15    else:
16        temp_password.append('.')
17    password_final = ''.join(temp_password)
18    return password_final
19
20 def dec(password):
21    # convert string to list
22    password_list = list(password)
23    splitpass = [password_list[i:i+3] for i in range(0, len(password_list), 3)]
24    temp_password = []
25    for word in splitpass:
26        a = ord(word[0]) - 80
27        b = ord(word[1]) - 80
28        desc = a * 26 + b
29        temp_password.append(chr(desc))
30    decrypt_final = ''.join(temp_password)
31    return decrypt_final
```

1.b. Apa output yang dihasilkan dari program tersebut jika input password adalah 'anakanakcerdas2020'

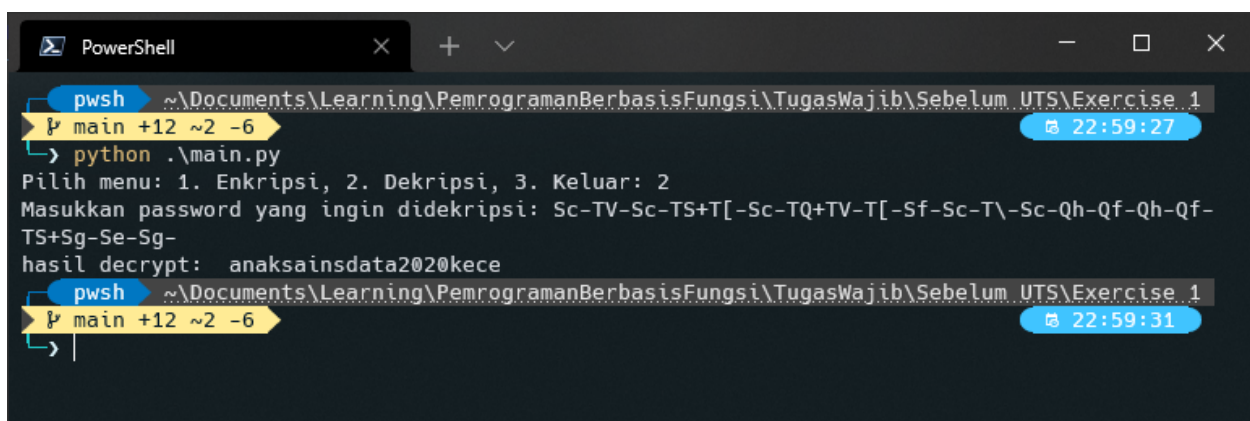
hasil encrypt: Sc-TV+Sc-TS-Sc+TV-Sc-TS+Se-Sg+TZ+Sf-Sc-T[+Qh-Qf-Qh+Qf-



```
PowerShell
pwsh ~\Documents\Learning\PemrogramanBerbasisFungsi\TugasWajib\Sebelum UTS\Exercise 1
P main +12 ~2 -6
python .\main.py
Pilih menu: 1. Enkripsi, 2. Dekripsi, 3. Keluar: 1
Masukkan password (maksimal 100 karakter): anakanakcerdas2020
hasil encrypt: Sc-TV+Sc-TS-Sc+TV-Sc-TS+Se-Sg+TZ+Sf-Sc-T[+Qh-Qf-Qh+Qf-
pwsh ~\Documents\Learning\PemrogramanBerbasisFungsi\TugasWajib\Sebelum UTS\Exercise 1
P main +12 ~2 -6
```

1.c. (Bonus) User tersebut lupa password asli yang dia inputkan ke dalam program tersebut, password setelah dienkripsi adalah 'Sc-TV-Sc-TS+T[-Sc-TQ+TV-T[-Sf-Sc-T\ -Sc-Qh-Qf-Qh-Qf-TS+Sg-Se-Sg-'. Bantulah user tersebut mendapatkan password asli nya!

hasil decrypt: anaksainsdata2020kece



```
PowerShell
pwsh ~\Documents\Learning\PemrogramanBerbasisFungsi\TugasWajib\Sebelum UTS\Exercise 1
P main +12 ~2 -6
python .\main.py
Pilih menu: 1. Enkripsi, 2. Dekripsi, 3. Keluar: 2
Masukkan password yang ingin didekripsi: Sc-TV-Sc-TS+T[-Sc-TQ+TV-T[-Sf-Sc-T\ -Sc-Qh-Qf-Qh-Qf-TS+Sg-Se-Sg-
hasil decrypt: anaksainsdata2020kece
pwsh ~\Documents\Learning\PemrogramanBerbasisFungsi\TugasWajib\Sebelum UTS\Exercise 1
P main +12 ~2 -6
```

>> Exercise 2 >>

Selesaikan penjumlahan 100 digit tersebut dengan membuat program dengan langkah sebagai berikut:

1. Buat file txt untuk bilangan pertama
2. Buat file txt untuk bilangan kedua
3. Input file txt bilangan pertama dan bilangan kedua
4. Buatlah program untuk menghitung penjumlahan kedua bilangan tersebut

Implementasikan konsep fungsi yang telah dipelajari ke dalam pembuatan program tersebut! Sertakan juga screenshot hasil program nya!

The screenshot shows the Visual Studio Code interface with a workspace named 'bilangan_pertama.txt - PBF-TugasWajib (Workspace)'. The Explorer panel on the left shows the project structure, including files like 'bilangan_kedua.txt', 'bilangan_pertama.txt', and 'main.py'. The main editor displays the content of 'bilangan_kedua.txt' and 'bilangan_pertama.txt', both containing long strings of digits. Below these, the 'main.py' file is open, showing a Python script that reads the contents of both text files and prints their sum using a lambda function. The script is as follows:

```
1 # the big number addition
2 bilangan_pertama = open('bilangan_pertama.txt').read()
3 bilangan_kedua = open('bilangan_kedua.txt').read()
4
5 print((lambda x,y: int(x) + int(y))(bilangan_pertama, bilangan_kedua))
6
```

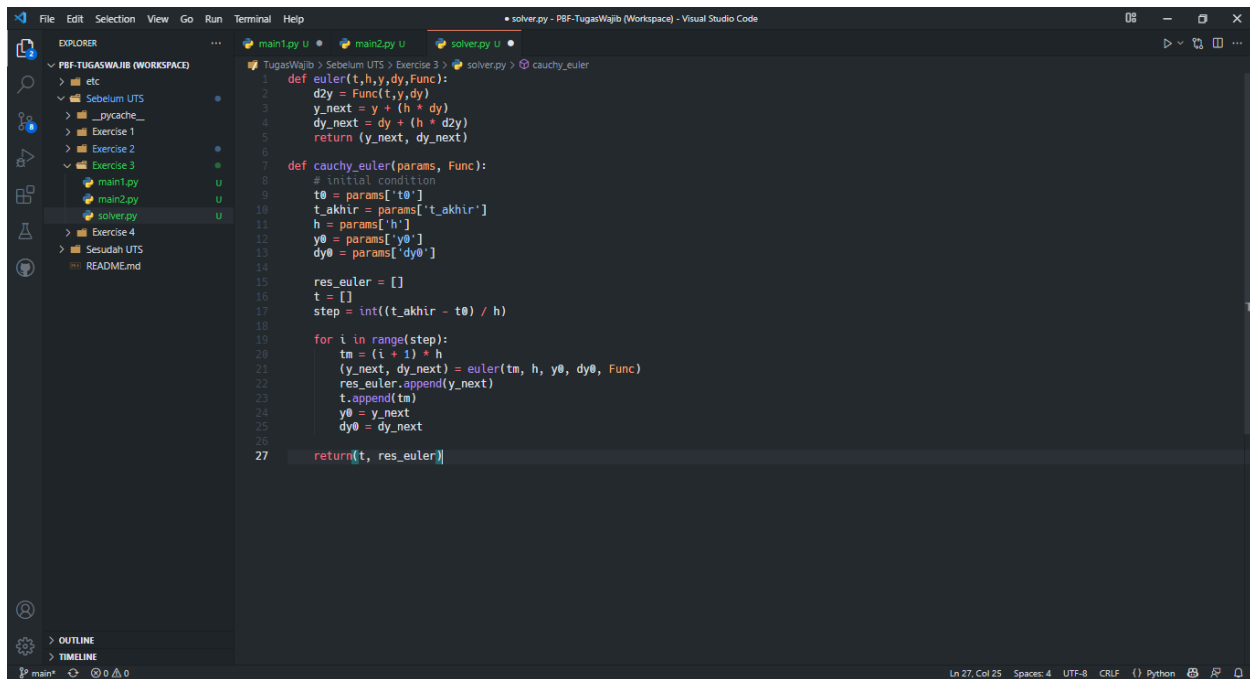
The TERMINAL panel at the bottom shows the execution of the program. It displays the command 'python .\main.py' and the resulting output, which is the sum of the two large numbers: 1161863833754815085247537074639538749043421784075800491438596240332637637365764884405967584806646855. The terminal also shows the command 'push' and the output of the program.

>> Exercise 3 >>

1. Apakah modul solver.py dapat digunakan untuk menyelesaikan persamaan differensial orde 2 selain kasus non linear pendulum? Jelaskan mengapa!

Jawab:

2. Implementasikan solver tersebut dengan cara membuat file solver.py



```
1 def euler(t,h,y,dy,Func):
2     d2y = Func(t,y,dy)
3     y_next = y + (h * dy)
4     dy_next = dy + (h * d2y)
5     return (y_next, dy_next)
6
7 def cauchy_euler(params, Func):
8     # initial condition
9     t0 = params['t0']
10    t_akhir = params['t_akhir']
11    h = params['h']
12    y0 = params['y0']
13    dy0 = params['dy0']
14
15    res_euler = []
16    t = []
17    step = int((t_akhir - t0) / h)
18
19    for i in range(step):
20        tm = (i + 1) * h
21        (y_next, dy_next) = euler(tm, h, y0, dy0, Func)
22        res_euler.append(y_next)
23        t.append(tm)
24        y0 = y_next
25        dy0 = dy_next
26
27    return(t, res_euler)
```

3. Untuk menyelesaikan persamaan dengan solver.py, bentuk fungsi harus diubah menjadi:

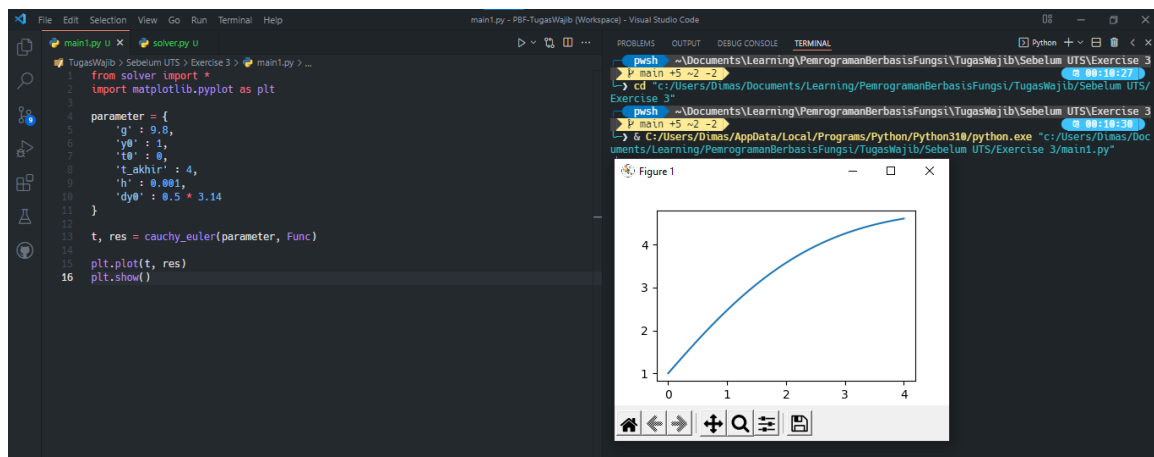
$$\frac{d^2\alpha}{dt^2} = -\frac{g}{L} * \sin(\alpha)$$

Definisikan fungsi Func sebagai fungsi yang me return nilai $-g/L * \sin(a)$!

```
3 def Func(g, l, a):  
4     return -(g/l) * math.sin(a)
```

4. Menggunakan Parameter Parameter yang ada dalam tabel diatas, buatlah program yang menggunakan solver.py untuk menemukan solusi persamaan diferensial non linear tersebut! Hint (Solusi Akhir berupa plot)

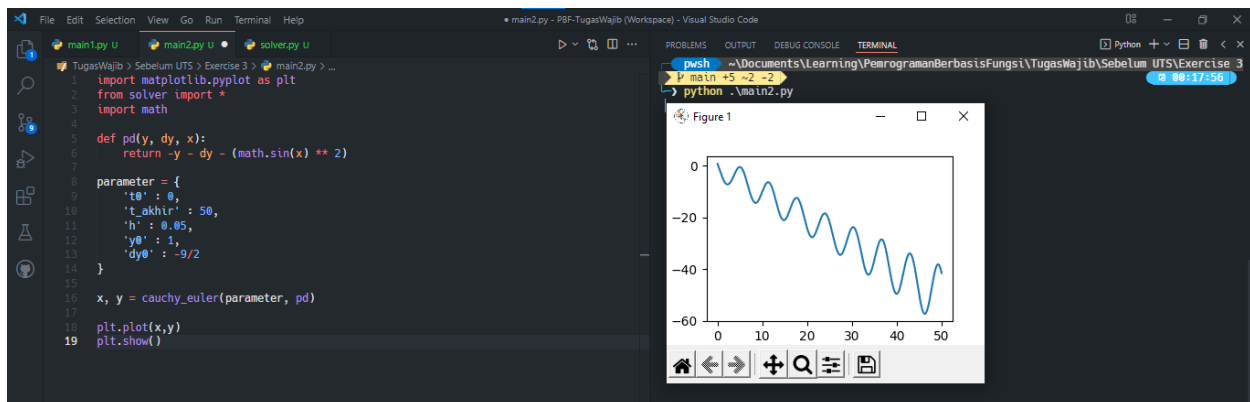
Parameter	Deskripsi	Value
g	Konstanta Gravitasi	9.8 m/s^2
L	Panjang Tali Pendulum	1 m
t0	Waktu Awal	0 detik
tn	Waktu Akhir	4 detik
h	Step Size	0.001
α_0	Nilai awal alpha	$0.5 * 3.14$



5. Perhatikan Persamaan Diferensial Diatas! Buatlah program untuk menyelesaikan PD tersebut dengan menggunakan solver.py sebagai modul dengan parameter berikut!

$$\frac{d^2y}{dx^2} = -y - \frac{dy}{dx} + \sin^2(x)$$

Parameter	Deskripsi	Value
X_0	X awal	0
X_n	X akhir	50
h	Step Size	0.05
y(X_0) = y_0	Nilai awal Y	1
y'(X_0) = y'_0	Nilai awal dy/dx	-9/2



>> Exercise 4 >>

1. Identifikasikan Variabel Global yang dibutuhkan!

Variabel Global yang dibutuhkan:

- Time/waktu waktu permainan (t)
- Skor dari Tim A (A_score)
- Skor dari Tim B (B_score)
- Status bola untuk mengetahui letak keberadaan bola (ball)
- Status permainan untuk menandakan permainan sedang berlangsung atau memulai pertandingan baru (status)

2. Identifikasikan Events yang terjadi!

- Permainan dimulai dari midfield. Bola akan dipegang Tim A.
- Event pertandingan terjadi sesuai dengan letak bola
- Saat Bola berada di tengah lapangan, maka MDA dan MDB akan beradu skill. MD yang memegang bola akan menggunakan skill Dribble(D) dan MD yang tidak memegang bola akan menggunakan skill Tackle(D). Jika $D > T$, maka bola akan berpindah ke Pemain ATK, Jika sebaliknya, maka bola akan berganti ke ATK lawan.
- Saat Bola berada di Area Penyerangan dan dikuasai oleh pemain ATK. Pemain ATK akan berhadapan dengan Pemain DF lawan. Jika Skill Dribble Pemain ATK > Skill Tackle DF lawan. Maka pemain ATK akan melakukan Shoot. Jika sebaliknya, maka bola dipegang oleh DF lawan
- Saat Pemain ATK akan melakukan shoot, maka dia akan berhadapan dengan GK Lawan. Jika skill shoot(H) > skill Save (S). Maka akan terjadi gol. Jika sebaliknya maka kiper akan menyelamatkan bola dan bola akan di oper ke DF.
- Saat Terjadi Gol, maka skor bertambah dan bola kembali ke MD tim yang kebobolan.
- Saat Bola berada di Area Pertahanan oleh pemain DF. Pemain DF akan beradu dengan Pemain ATK Lawan. Pemain DF akan melakukan passing ke pemain MD. Jika

skill Passing(P) > skill Intercept(I) lawan, maka bola akan berpindah ke MD. Jika sebaliknya maka, bola akan diambil oleh ATK lawan dan ATK lawan akan langsung melakukan Shooting.

3. Buatlah simulasi program tersebut menggunakan konsep paradigma fungsional dengan kondisi sebagai berikut:

a. Jumlah Supporter Tim A = 100.000 orang

b. Jumlah Supporter Tim A = 115.000 orang

Team A					Team B			
Role	Mentality	Skill	Value		Role	Mentality	Skill	Value
GK	80	Save	81		GK	77	Save	86
DF	79	Tackle	79		DF	78	Tackle	80
		Passing	78				Passing	81
MD	78	Tackle	60		MD	79	Tackle	70
		Dribble	76				Dribble	70
ATK	77	Dribble	80		ATK	80	Dribble	81
		Intercept	85				Intercept	86
		Shoot	92				Shoot	90

Jawab:

```

from numpy import random
supporter_A = 100000
supporter_B = 100000
ability_default = {
    'GK': 81, 'GK_E': 80, 'DF': 79, 'DF_P': 78, 'DF_E': 79, 'MD': 76, 'MD_D': 76, 'MD_E': 78, 'ATK_D': 80,
    'ATK_P': 85, 'ATK_E': 92
}

def main_game():
    t = 0
    while t < 90:
        print('Menit -', t)
        MDvsMD(t, t_A, t_B)
    t = 90

main_game()

```

Output exceeds the size limit. Open the full output data in a text editor

```

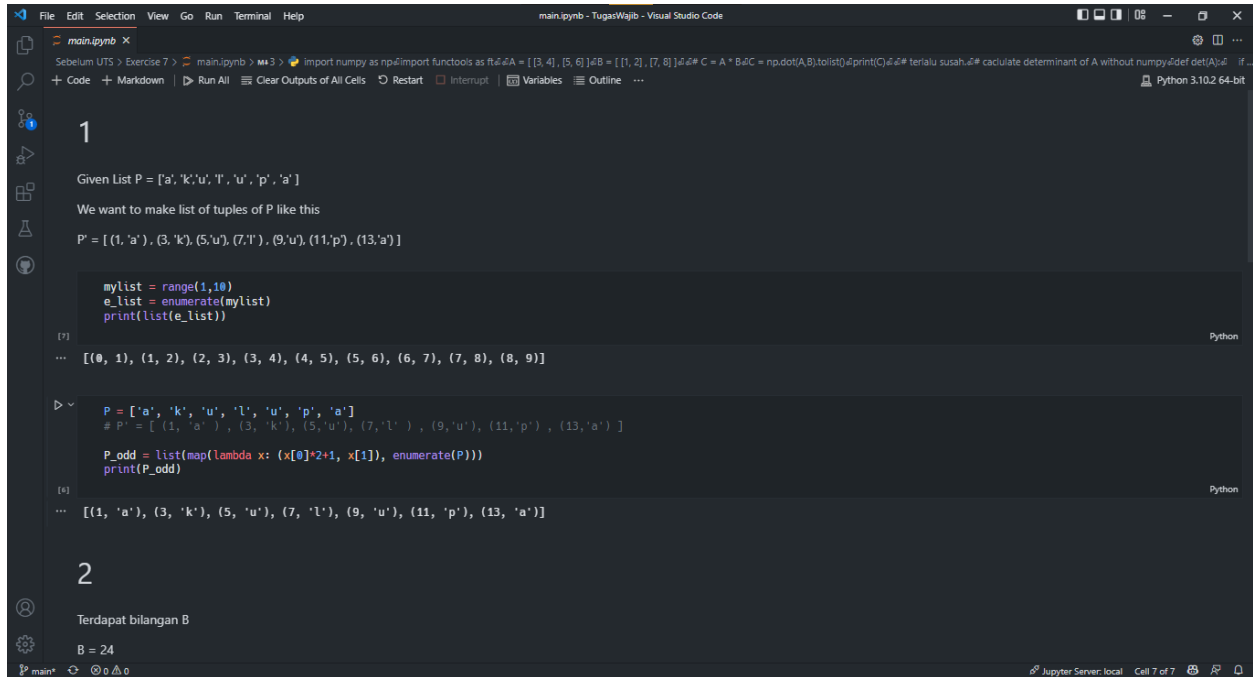
Menit - 0
MD berhasil Menghadang Giringan bola Lawan. Bersiap mengoper ke ATK.
ATK Menggiring Bola Melewati DF. Bersiap menendang.
ATK Menembak Bola Ke arah gawang. Gol!
1

Menit - 5
MD berhasil Menggiring Bola Melewati Musuh. MD mengoper bola ke ATK.
ATK Menggiring Bola Melewati DF. Bersiap menendang.
ATK Menembak Bola Ke arah gawang. Gol!
2

Menit - 9
MD berhasil Menggiring Bola Melewati Musuh. MD mengoper bola ke ATK.
DF berhasil merebut bola dari ATK.
DF berhasil Mengoper Bola Ke MD.
MD berhasil Menggiring Bola Melewati Musuh. MD mengoper bola ke ATK.
DF berhasil merebut bola dari ATK.
ATK berhasil Menotong Operan DF. Bersiap menendang.
ATK Menembak Bola Ke arah gawang. Gol!
3

```


>> Exercise 7 >>



```
main.ipynb - TugasWajib - Visual Studio Code
Sebelum UTS > Exercise 7 > main.ipynb > 1:3 > import numpy as np; A = [[3, 4], [5, 6]]; B = [[1, 2], [7, 8]]; C = A * B; C = np.dot(A, B); print(C); # terlalu susah, # calculate determinant of A without numpy; def det(A): if ...
Python 3.10.2 64-bit

1

Given List P = ['a', 'k', 'u', 'l', 'u', 'p', 'a']
We want to make list of tuples of P like this
P' = [(1, 'a'), (3, 'k'), (5, 'u'), (7, 'l'), (9, 'u'), (11, 'p'), (13, 'a')]

mylist = range(1,10)
e_list = enumerate(mylist)
print(list(e_list))

[7]
... [(0, 1), (1, 2), (2, 3), (3, 4), (4, 5), (5, 6), (6, 7), (7, 8), (8, 9)]

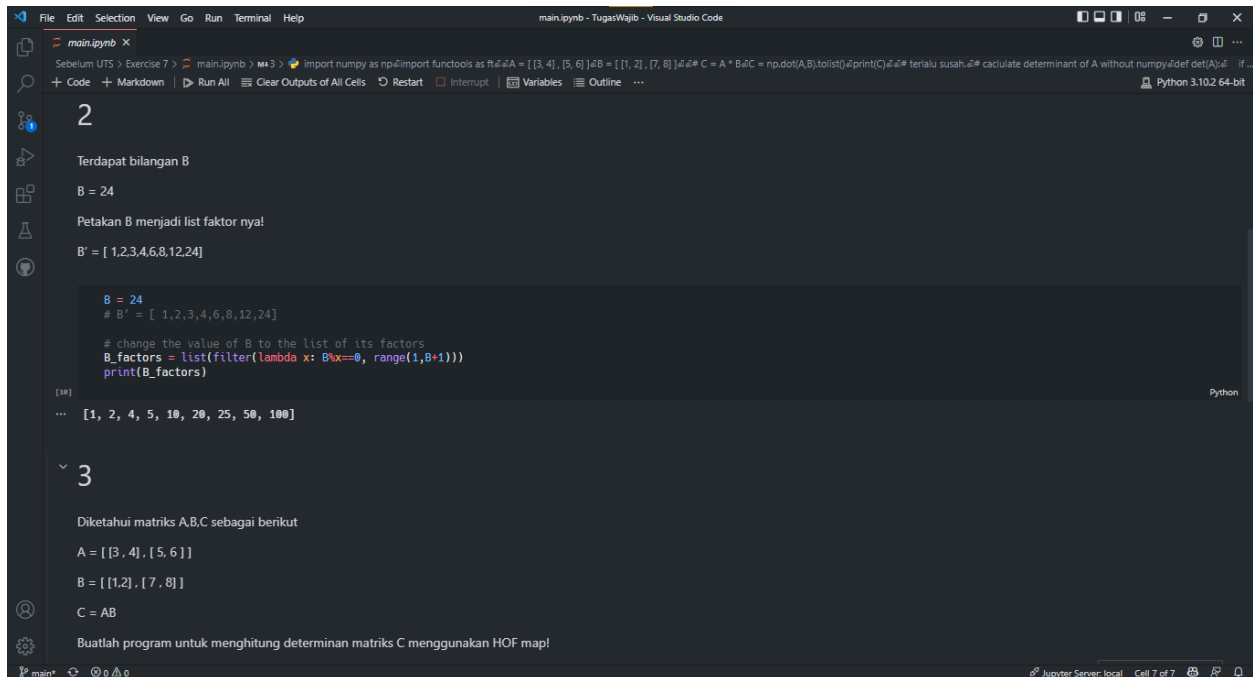
P = ['a', 'k', 'u', 'l', 'u', 'p', 'a']
# P' = [ (1, 'a') , (3, 'k') , (5, 'u') , (7, 'l') , (9, 'u') , (11, 'p') , (13, 'a') ]

P_odd = list(map(lambda x: (x[0]*2+1, x[1]), enumerate(P)))
print(P_odd)

[6]
... [(1, 'a'), (3, 'k'), (5, 'u'), (7, 'l'), (9, 'u'), (11, 'p'), (13, 'a')]

2

Terdapat bilangan B
B = 24
```



```
main.ipynb - TugasWajib - Visual Studio Code
Sebelum UTS > Exercise 7 > main.ipynb > 1:3 > import numpy as np; A = [[3, 4], [5, 6]]; B = [[1, 2], [7, 8]]; C = A * B; C = np.dot(A, B); print(C); # terlalu susah, # calculate determinant of A without numpy; def det(A): if ...
Python 3.10.2 64-bit

2

Terdapat bilangan B
B = 24

Petakan B menjadi list faktor nya!
B' = [ 1,2,3,4,6,8,12,24]

B = 24
# B' = [ 1,2,3,4,6,8,12,24]

# change the value of B to the list of its factors
B_factors = list(filter(lambda x: B%x==0, range(1,B+1)))
print(B_factors)

[18]
... [1, 2, 4, 5, 10, 20, 25, 50, 100]

3

Diketahui matriks A,B,C sebagai berikut
A = [[3, 4], [5, 6]]
B = [[1,2], [7, 8]]
C = AB

Buatlah program untuk menghitung determinan matriks C menggunakan HOF map!
```

File Edit Selection View Go Run Terminal Help main.ipynb - TugasWajib - Visual Studio Code

main.ipynb x

Sebelum UTS > Exercise 7 > main.ipynb > 3 > import numpy as npimport functools as ftA = [[3,4],[5,6]]B = [[1,2],[7,8]]C = A * BdotC = np.dot(A,B).tolist()print(C)# terlalu susah.# caculate determinant of A without numpydef det(A): if ...

3

Diketahui matriks A,B,C sebagai berikut

A = $\begin{bmatrix} 3 & 4 \\ 5 & 6 \end{bmatrix}$

B = $\begin{bmatrix} 1 & 2 \\ 7 & 8 \end{bmatrix}$

C = AB

Buatlah program untuk menghitung determinan matriks C menggunakan HOF map!

```
import numpy as np
import functools as ft

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

# C = A * B
C = np.dot(A,B).tolist()
print(C)

# terlalu susah.
# caculate determinant of A without numpy
def det(A):
    if len(A) == 1:
        return A[0][0]
    else:
        return sum( [ A[0][i]*(-1)**i*det( [ [ A[j][k] for k in range(len(A)) if k != i ] for j in range(1,len(A)) ] ) for i in range(len(A[0])) ] )

print(det(C))
```

Python

... $\begin{bmatrix} 31 & 38 \\ 47 & 58 \end{bmatrix}$

12

main* Jupyter Server: local Cell 7 of 7