

Nama : Danang Adityo Nugroho

Kelas : TIF 3A4

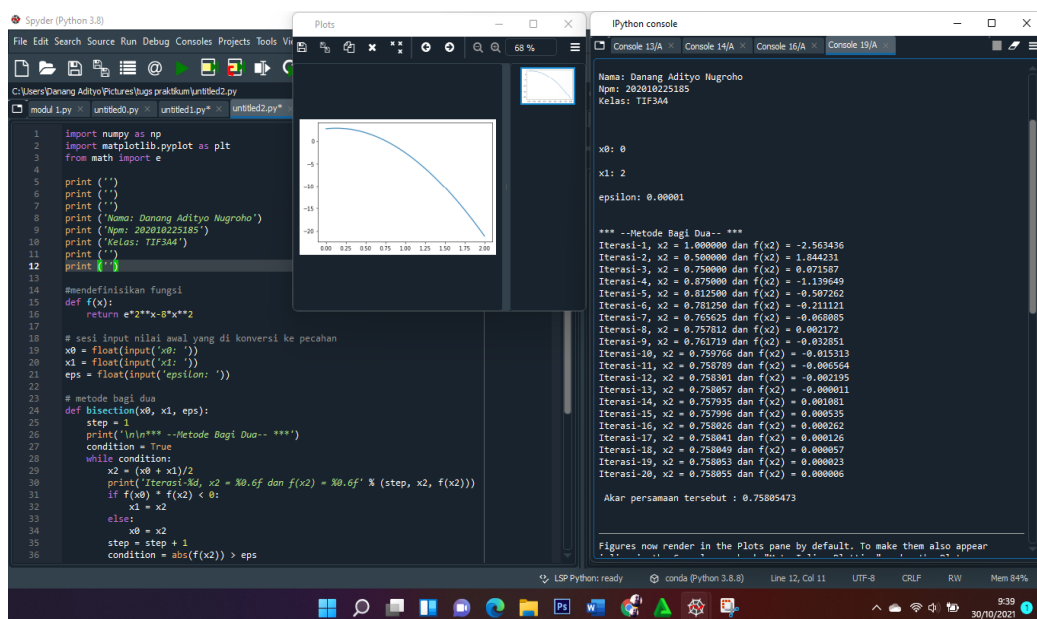
Npm : 202010225185

Tugas

Solusi Persamaan Non Linear

Pertanyaan :

- Bisection (bagi dua)



Coding Bisection (bagi dua)

```
import numpy as np
```

```
import matplotlib.pyplot as plt
```

```
from math import e
```

```
print ("
```

```
print ("
```

```
print ("
```

```
print ('Nama: Danang Adityo Nugroho')
```

```
print ('Npm: 202010225185')
```

```
print ('Kelas: TIF3A4')
```

```
print ("
```

```
print ("
```

```
#mendefinisikan fungsi
```

```
def f(x):
```

```
    return e*2**x-8*x**2
```

```
# sesi input nilai awal yang di konversi ke pecahan
```

```
x0 = float(input('x0: '))
```

```
x1 = float(input('x1: '))
```

```
eps = float(input('epsilon: '))
```

```
# metode bagi dua
```

```
def bisection(x0, x1, eps):
```

```
    step = 1
```

```
    print('\n\n*** --Metode Bagi Dua-- ***')
```

```
    condition = True
```

```
    while condition:
```

```
        x2 = (x0 + x1)/2
```

```
        print('Iterasi-%d, x2 = %0.6f dan f(x2) = %0.6f' % (step, x2, f(x2)))
```

```

    if f(x0) * f(x2) < 0:

        x1 = x2

    else:

        x0 = x2

    step = step + 1

    condition = abs(f(x2)) > eps

    print('\n Akar persamaan tersebut : %0.8f' % x2)

# pengecekan nilai awal

if f(x0) * f(x1) > 0.0:

    print('Nilai yang di prediksi tidak mengurung akar')

    print('Silahkan mencoba ulang prediksi nilai baru')

    print()

else:

    bisection(x0, x1, eps)

# menggambar fungsi

rr = np.linspace(0, 2, 100) # masukkan nilai tebakan awal

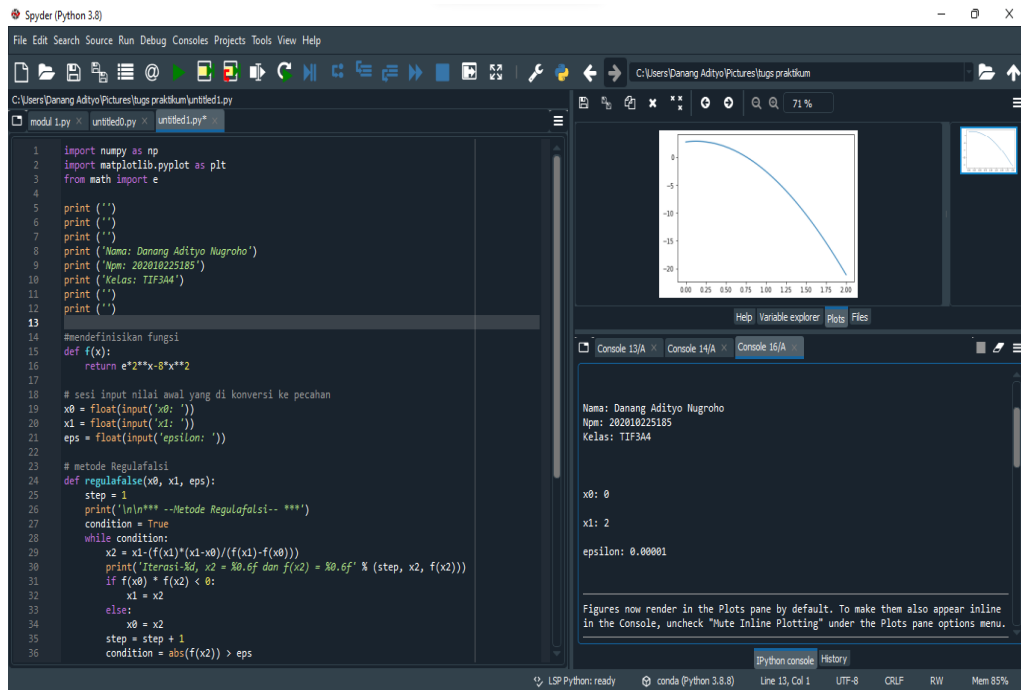
plt.plot(rr, f(rr))

plt.show()

plt.savefig("biseksi.png") # untuk menyimpan gambar fungsi

```

- Regula falsi



Coding RegulaFalsi

import numpy as np

import matplotlib.pyplot as plt

from math import e

print (")

print (")

print (")

print ('Nama: Danang Adityo Nugroho')

print ('Npm: 202010225185')

print ('Kelas: TIF3A4')

print (")

print (")

#mendefinisikan fungsi

def f(x):

```
return e*2**x-8*x**2
```

```
# sesi input nilai awal yang di konversi ke pecahan
```

```
x0 = float(input('x0: '))
```

```
x1 = float(input('x1: '))
```

```
eps = float(input('epsilon: '))
```

```
# metode Regulafalsi
```

```
def regulafalse(x0, x1, eps):
```

```
    step = 1
```

```
    print('\n\n*** --Metode Regulafalsi-- ***')
```

```
    condition = True
```

```
    while condition:
```

```
        x2 = x1-(f(x1)*(x1-x0)/(f(x1)-f(x0)))
```

```
        print('Iterasi-%d, x2 = %0.6f dan f(x2) = %0.6f' % (step, x2, f(x2)))
```

```
        if f(x0) * f(x2) < 0:
```

```
            x1 = x2
```

```
        else:
```

```
            x0 = x2
```

```
        step = step + 1
```

```
        condition = abs(f(x2)) > eps
```

```
    print('\n Akar persamaan tersebut : %0.8f' % x2)
```

```
# Menggambar fungsi
```

```
rr= np.linspace(0,2,100) # masukan nilai tebakan awal
```

```
plt.plot(rr,f(rr))
```

```
plt.show()
```

```
plt.savefig("regulafalsi.png")
```

```
# pengecekan nilai awal
```

```
if f(x0) * f(x1) > 0.0:
```

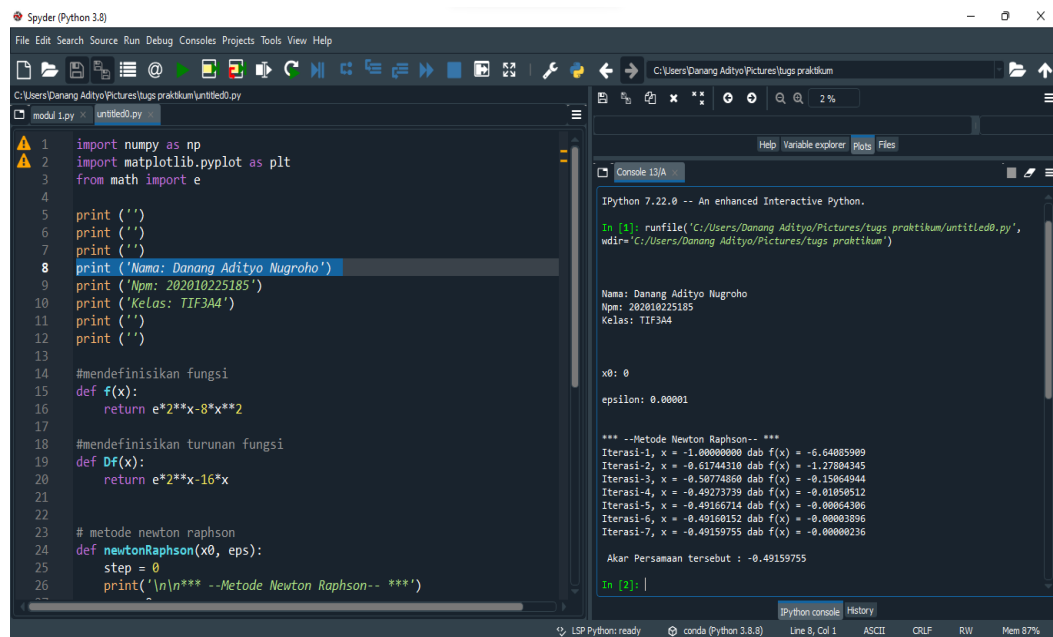
```
    print('Nilai yang di prediksi tidak mengurung akar')
```

```
    print('Silahkan mencoba ulang prediksi nilai baru')
```

```
else:
```

```
    regulafalse(x0, x1, eps)
```

- Newton Raphson



The screenshot shows the Spyder Python IDE interface. The editor on the left contains a Python script for the Newton-Raphson method. The console on the right shows the execution output, including user input and the iterative results of the method.

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3 from math import e
4
5 print ('')
6 print ('')
7 print ('')
8 print ('Nama: Danang Aditya Nugroho')
9 print ('Npm: 202010225185')
10 print ('Kelas: TIF3A4')
11 print ('')
12 print ('')
13
14 #mendefinisikan fungsi
15 def f(x):
16     return e**2*x-8*x**2
17
18 #mendefinisikan turunan fungsi
19 def Df(x):
20     return e**2*x-16*x
21
22 # metode newton raphson
23 def newtonRaphson(x0, eps):
24     step = 0
25     print('\n\n*** --Metode Newton Raphson-- ***)
26
```

Console Output:

```
IPython 7.22.0 -- An enhanced Interactive Python.

In [1]: runfile('C:/Users/Danang Aditya/Pictures/tugs praktikum/untitled0.py',
wdir='C:/Users/Danang Aditya/Pictures/tugs praktikum')

Nama: Danang Aditya Nugroho
Npm: 202010225185
Kelas: TIF3A4

x0: 0
epsilon: 0.00001

*** --Metode Newton Raphson-- ***
Iterasi-1, x = -1.000000000 dab f(x) = -6.44085909
Iterasi-2, x = -0.61744310 dab f(x) = -1.27804345
Iterasi-3, x = -0.50774860 dab f(x) = -0.15064944
Iterasi-4, x = -0.49273739 dab f(x) = -0.01895812
Iterasi-5, x = -0.49166714 dab f(x) = -0.00064306
Iterasi-6, x = -0.49160152 dab f(x) = -0.00003896
Iterasi-7, x = -0.49159755 dab f(x) = -0.00000236

Akar Persamaan tersebut : -0.49159755

In [2]:
```

Coding newton Raphson :

```
import numpy as np
```

```

import matplotlib.pyplot as plt
from math import e

#mendefinisikan fungsi
def f(x):
    return e**2*x-8*x**2

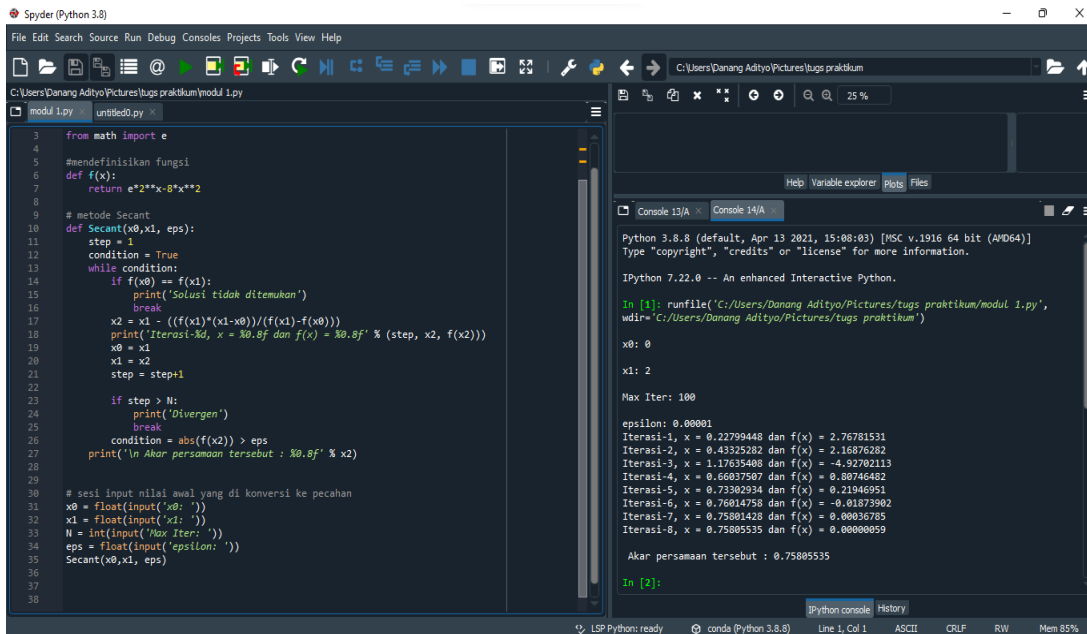
#mendefinisikan turunan fungsi
def Df(x):
    return e**2*x-16*x

# metode newton raphson
def newtonRaphson(x0, eps):
    step = 0
    print('\n\n*** --Metode Newton Raphson-- ***')
    xn = x0
    for n in range(0,100): #maksimal iterasi adalah 100
        fxn=f(xn)
        if abs(fxn)<eps:
            print('\n Akar Persamaan tersebut : %0.8f' % xn)
            return xn
        Dfxn = Df(xn)
        if Dfxn == 0:
            print('Solusi tidak ditemukan')
            return None
        xn = xn-(fxn/Dfxn)
        step = step + 1
        print('Iterasi-%d, x = %0.8f dab f(x) = %0.08f' % (step,xn,f(xn)))
    print('Iterasi maksimum, solusi tidak ditemukan')

# sesi input nilai awal yang di konversi ke pecahan
x0 = float(input('x0: '))
eps = float(input('epsilon: '))
newtonRaphson(x0, eps)

```

- Secant



Coding Secanf

```
import numpy as np
```

```
import matplotlib.pyplot as plt
```

```
from math import e
```

```
#mendefinisikan fungsi
```

```
def f(x):
```

```
    return e**x-8*x**2
```

```
# metode Secant
```

```
def Secant(x0,x1, eps):
```

```
    step = 1
```

```
    condition = True
```

```
    while condition:
```

```
        if f(x0) == f(x1):
```

```
            print('Solusi tidak ditemukan')
```

```
            break
```

```
        x2 = x1 - ((f(x1)*(x1-x0))/(f(x1)-f(x0)))
```

```
        print('Iterasi-%d, x = %0.8f dan f(x) = %0.8f' % (step, x2, f(x2)))
```



```
x0 = x1
```

```
x1 = x2
```

```
step = step+1
```

```
if step > N:
```

```
    print('Divergen')
```

```
    break
```

```
condition = abs(f(x2)) > eps
```

```
    print("\n Akar persamaan tersebut : %0.8f" % x2)
```

```
# sesi input nilai awal yang di konversi ke pecahan
```

```
x0 = float(input('x0: '))
```

```
x1 = float(input('x1: '))
```

```
N = int(input('Max lter: '))
```

```
eps = float(input('epsilon: '))
```

```
Secant(x0,x1, eps)
```