

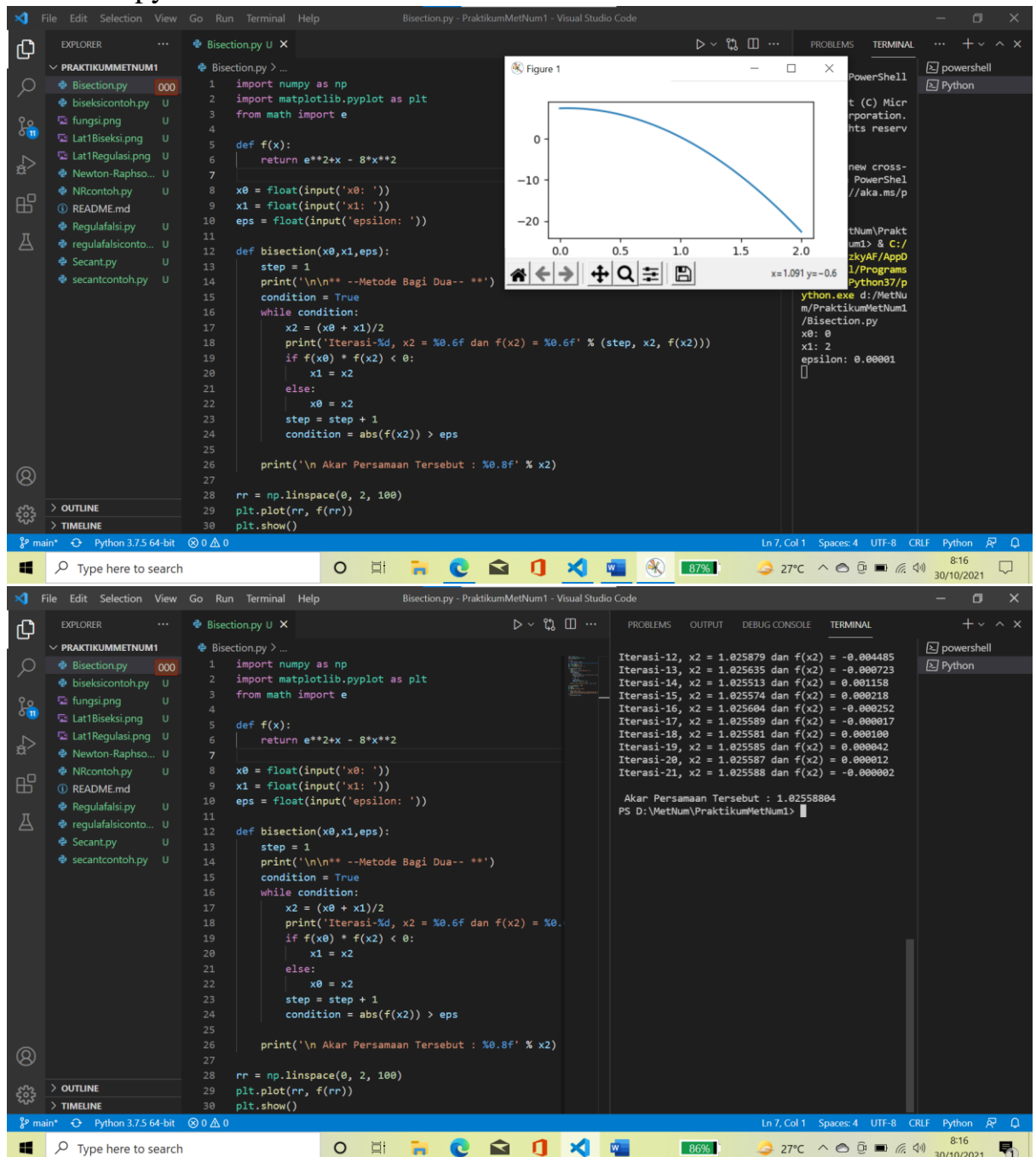
Tugas Latihan1

Nama : Rizky Antonio Figo

Kelas : TF3A5

Npm : 202010225227

1. Bisection.py



2. Regulasifalsi.py

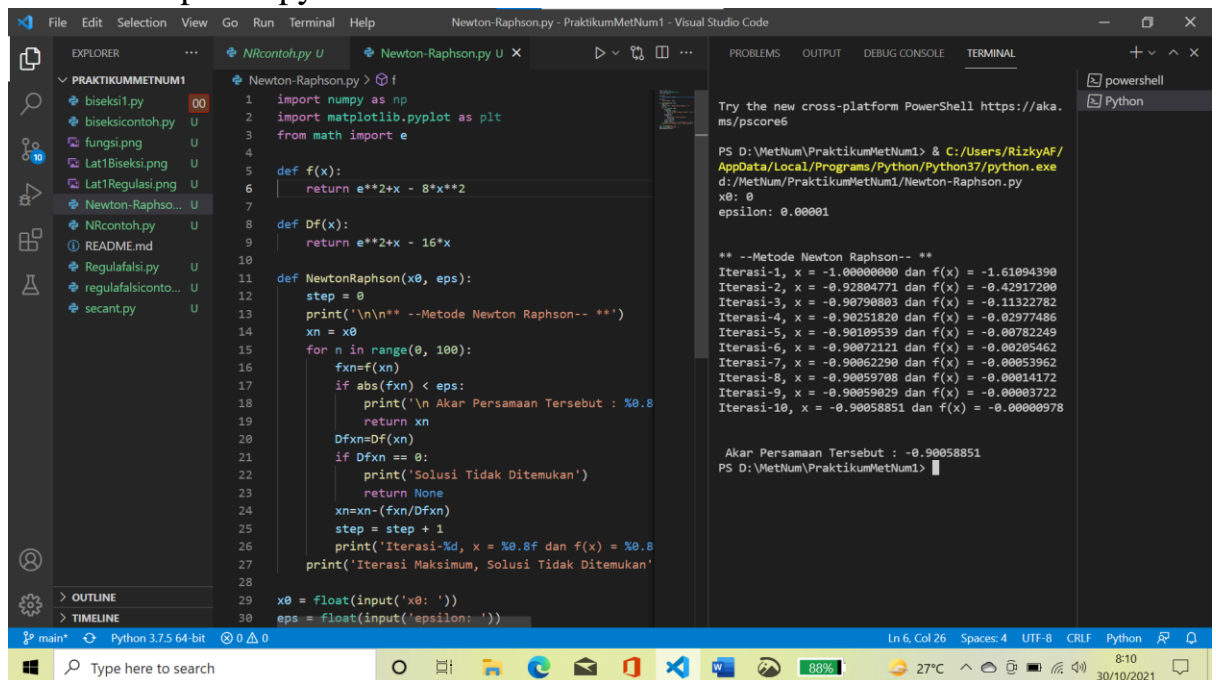
The image shows a Visual Studio Code editor with a Python script named `Regulasifalsi.py` open. The script implements the Regula-Falsi method for finding roots of a function. The function `f(x)` is defined as $f(x) = e^{2x} - 8x^2$. The algorithm takes initial values `x0` and `x1` and a tolerance `eps` as input. It iteratively updates the root estimate until the absolute value of the function at the estimate is less than the tolerance.

The terminal window shows the output of the script, displaying the iterative results of the Regula-Falsi method. The root value is found to be approximately 1.02558737.

```
Iterasi-5, x2 = 1.016249 dan f(x2) = 0.143216
Iterasi-6, x2 = 1.022440 dan f(x2) = 0.048423
Iterasi-7, x2 = 1.024529 dan f(x2) = 0.016302
Iterasi-8, x2 = 1.025232 dan f(x2) = 0.005481
Iterasi-9, x2 = 1.025468 dan f(x2) = 0.001842
Iterasi-10, x2 = 1.025548 dan f(x2) = 0.000619
Iterasi-11, x2 = 1.025574 dan f(x2) = 0.000288
Iterasi-12, x2 = 1.025583 dan f(x2) = 0.000079
Iterasi-13, x2 = 1.025586 dan f(x2) = 0.000023
Iterasi-14, x2 = 1.025587 dan f(x2) = 0.000008

Akar Persamaan Tersebut : 1.02558737
PS D:\Methum\PraktikumMethum1>
```

3. Newton-Raphson.py



The screenshot shows the Visual Studio Code editor with the file `Newton-Raphson.py` open. The code defines a function `f(x)` and its derivative `Df(x)`, and implements the Newton-Raphson method. The terminal output shows the execution of the script, displaying the iterations and the final root found.

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3 from math import e
4
5 def f(x):
6     return e**2*x - 8*x**2
7
8 def Df(x):
9     return e**2*x - 16*x
10
11 def NewtonRaphson(x0, eps):
12     step = 0
13     print('\n\n--Metode Newton Raphson-- **')
14     xn = x0
15     for n in range(0, 100):
16         fxn=f(xn)
17         if abs(fxn) < eps:
18             print('\n Akar Persamaan Tersebut : %0.8'
19                 % xn)
20             return xn
21         Dfxn=Df(xn)
22         if Dfxn == 0:
23             print('Solusi Tidak Ditemukan')
24             return None
25         xn=xn-(fxn/Dfxn)
26         step = step + 1
27         print('Iterasi-%d, x = %0.8f dan f(x) = %0.8'
28             % (step, xn, f(xn)))
29     print('Iterasi Maksimum, Solusi Tidak Ditemukan')
30
31 x0 = float(input('x0: '))
32 eps = float(input('epsilon: '))
```

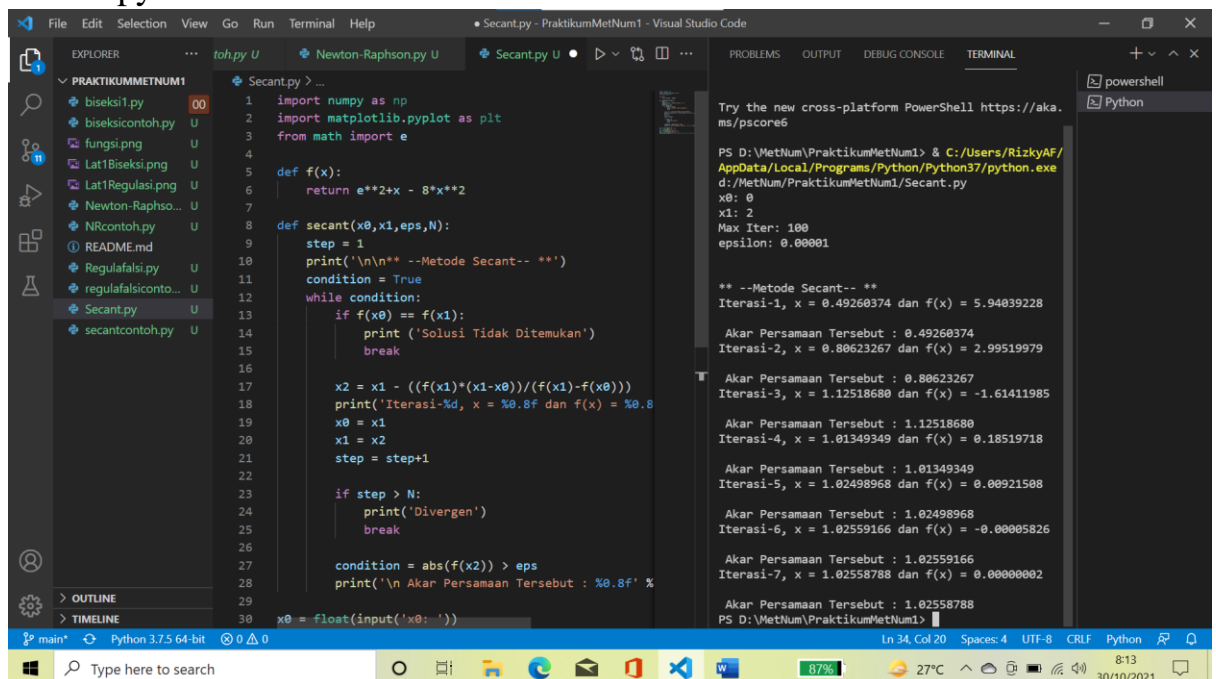
Terminal Output:

```
PS D:\MetNum\PraktikumMetNum1> & C:\Users\RiskyaF\AppData\Local\Programs\Python\Python37\python.exe d:/MetNum/PraktikumMetNum1/Newton-Raphson.py
x0: 0
epsilon: 0.00001

--Metode Newton Raphson-- **
Iterasi-1, x = -1.00000000 dan f(x) = -1.61094390
Iterasi-2, x = -0.92804771 dan f(x) = -0.42917200
Iterasi-3, x = -0.90790803 dan f(x) = -0.11322782
Iterasi-4, x = -0.90251820 dan f(x) = -0.02977486
Iterasi-5, x = -0.90109539 dan f(x) = -0.00782249
Iterasi-6, x = -0.90072121 dan f(x) = -0.00205462
Iterasi-7, x = -0.90062290 dan f(x) = -0.00053962
Iterasi-8, x = -0.90059708 dan f(x) = -0.00014172
Iterasi-9, x = -0.90059029 dan f(x) = -0.00003722
Iterasi-10, x = -0.90058851 dan f(x) = -0.00000978

Akar Persamaan Tersebut : -0.90058851
PS D:\MetNum\PraktikumMetNum1>
```

4. Secant.py



The screenshot shows the Visual Studio Code editor with the file `Secant.py` open. The code defines a function `f(x)` and implements the Secant method. The terminal output shows the execution of the script, displaying the iterations and the final root found.

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3 from math import e
4
5 def f(x):
6     return e**2*x - 8*x**2
7
8 def secant(x0,x1,eps,N):
9     step = 1
10    print('\n\n--Metode Secant-- **')
11    condition = True
12    while condition:
13        if f(x0) == f(x1):
14            print('Solusi Tidak Ditemukan')
15            break
16
17        x2 = x1 - ((f(x1)*(x1-x0))/(f(x1)-f(x0)))
18        print('Iterasi-%d, x = %0.8f dan f(x) = %0.8'
19            % (step, x2, f(x2)))
20        x0 = x1
21        x1 = x2
22        step = step+1
23
24        if step > N:
25            print('Divergen')
26            break
27
28        condition = abs(f(x2)) > eps
29        print('\n Akar Persamaan Tersebut : %0.8f' %
30            x2)
31
32    x0 = float(input('x0: '))
```

Terminal Output:

```
PS D:\MetNum\PraktikumMetNum1> & C:\Users\RiskyaF\AppData\Local\Programs\Python\Python37\python.exe d:/MetNum/PraktikumMetNum1/Secant.py
x0: 0
x1: 2
Max Iter: 100
epsilon: 0.00001

--Metode Secant-- **
Iterasi-1, x = 0.49260374 dan f(x) = 5.94039228
Akar Persamaan Tersebut : 0.49260374
Iterasi-2, x = 0.80623267 dan f(x) = 2.99519979
Akar Persamaan Tersebut : 0.80623267
Iterasi-3, x = 1.12518680 dan f(x) = -1.61411985
Akar Persamaan Tersebut : 1.12518680
Iterasi-4, x = 1.01349349 dan f(x) = 0.18519718
Akar Persamaan Tersebut : 1.01349349
Iterasi-5, x = 1.02498968 dan f(x) = 0.00921508
Akar Persamaan Tersebut : 1.02498968
Iterasi-6, x = 1.02559166 dan f(x) = -0.00005826
Akar Persamaan Tersebut : 1.02559166
Iterasi-7, x = 1.02558788 dan f(x) = 0.00000002
Akar Persamaan Tersebut : 1.02558788
PS D:\MetNum\PraktikumMetNum1>
```