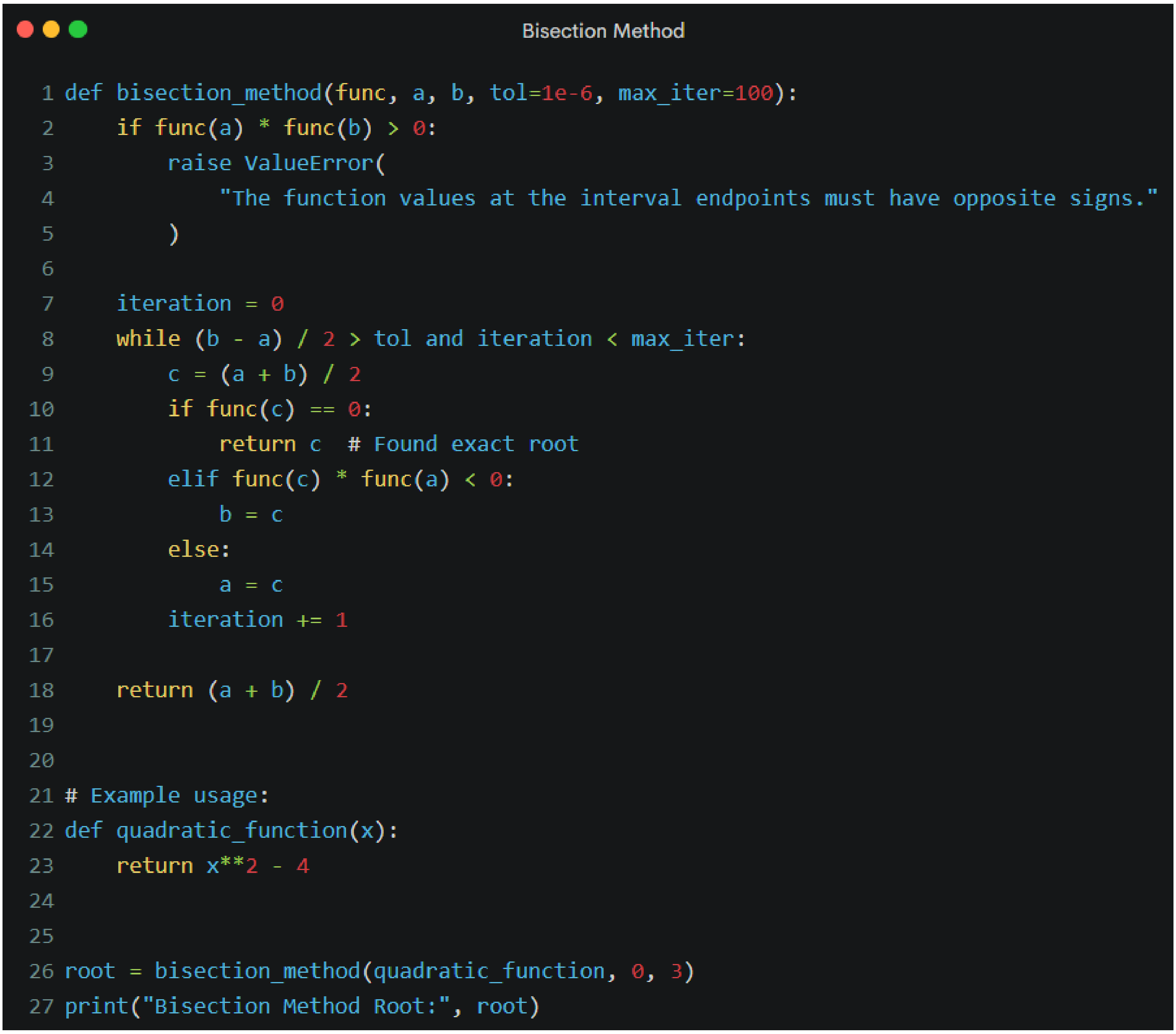
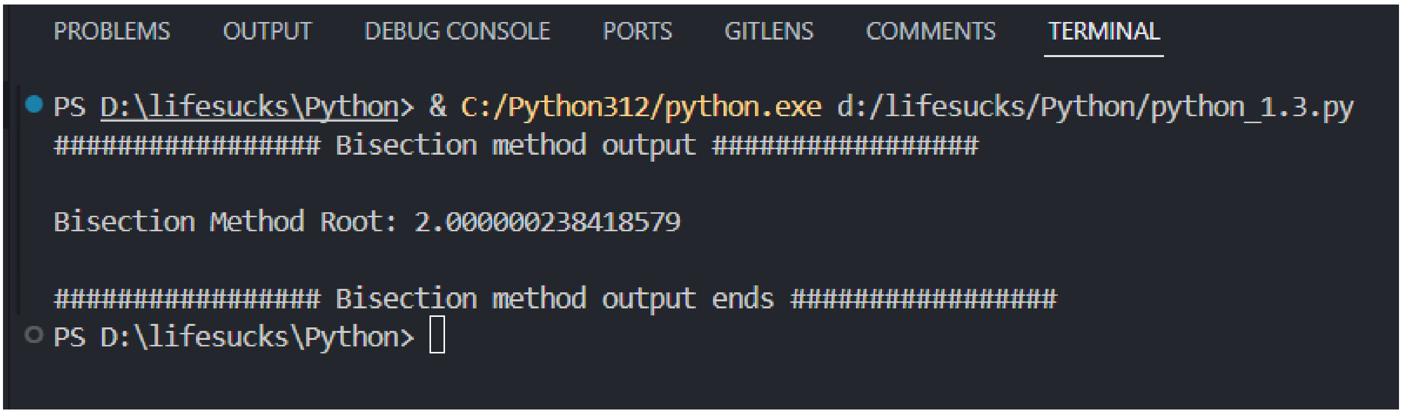
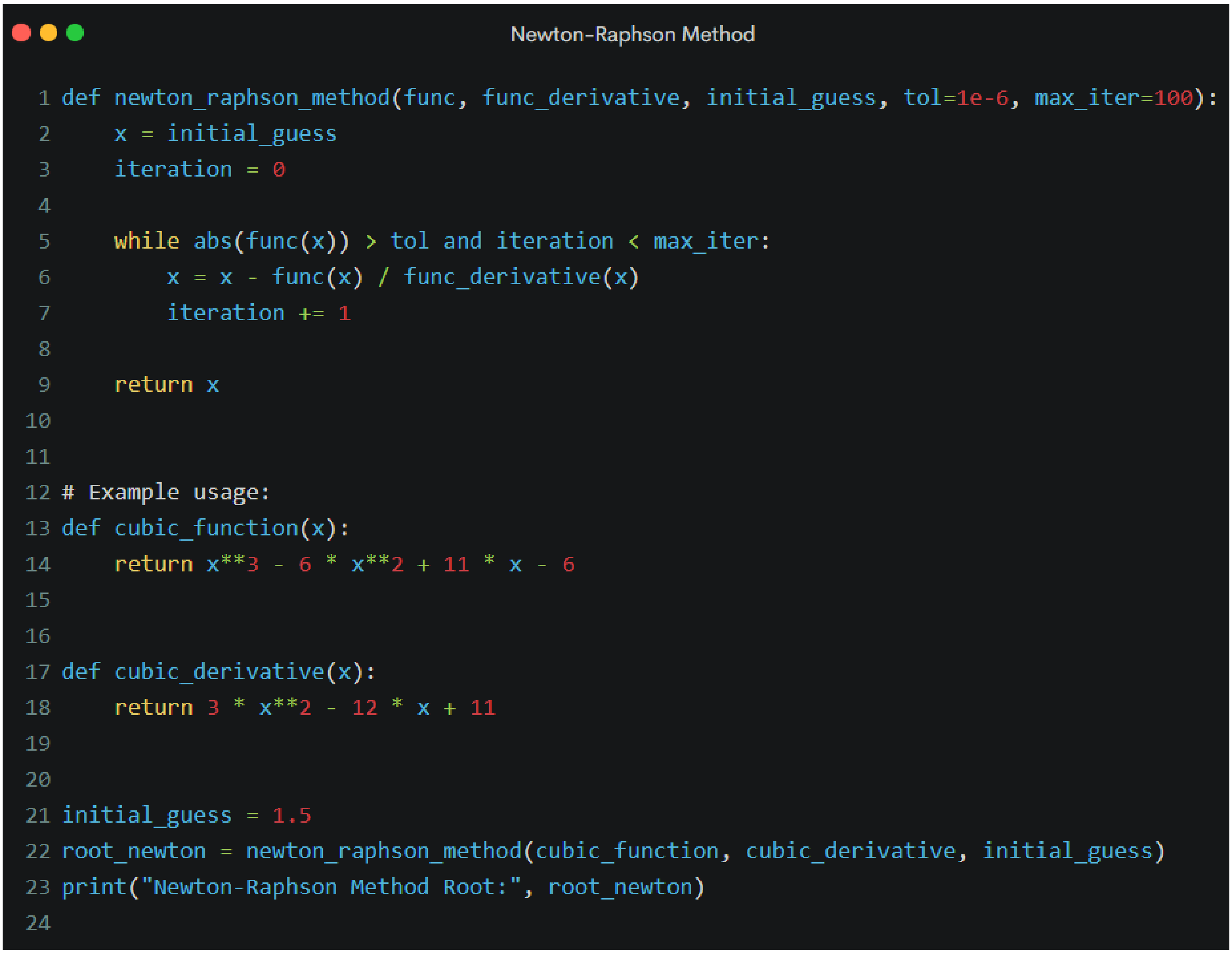


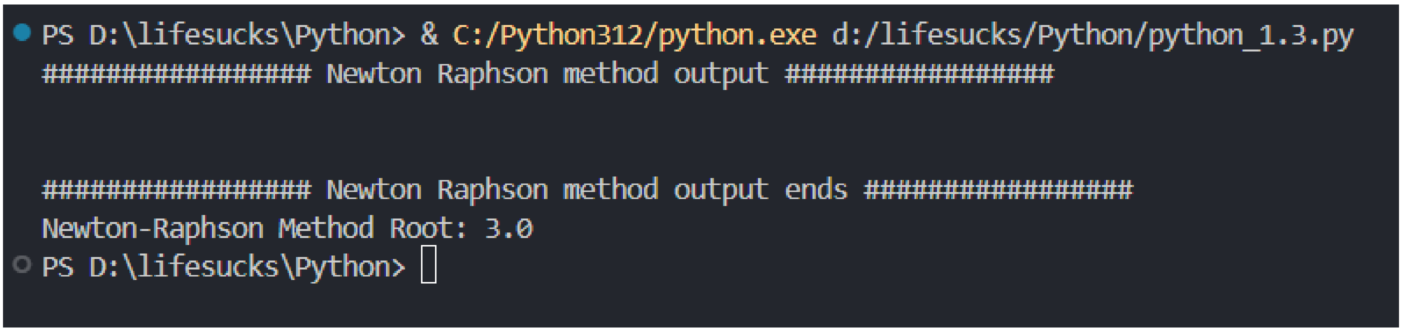
```python
def bisection_method(func, a, b, tol=1e-6, max_iter=100):
    if func(a) * func(b) > 0:
        raise ValueError(
            "The function values at the interval endpoints must have opposite signs."
        )

    iteration = 0
    while (b - a) / 2 > tol and iteration < max_iter:
        c = (a + b) / 2
        if func(c) == 0:
            return c  # Found exact root
        elif func(c) * func(a) < 0:
            b = c
        else:
            a = c
        iteration += 1

    return (a + b) / 2


# Example usage:
def quadratic_function(x):
    return x**2 - 4


root = bisection_method(quadratic_function, 0, 3)
print("Bisection Method Root:", root)
```



```
PS D:\lifesucks\Python> & C:/Python312/python.exe d:/lifesucks/Python/python_1.3.py
################## Bisection method output ##################

Bisection Method Root: 2.000000238418579

################## Bisection method output ends ##################
PS D:\lifesucks\Python> 
```



```python
def newton_raphson_method(func, func_derivative, initial_guess, tol=1e-6, max_iter=100):
    x = initial_guess
    iteration = 0

    while abs(func(x)) > tol and iteration < max_iter:
        x = x - func(x) / func_derivative(x)
        iteration += 1

    return x


# Example usage:
def cubic_function(x):
    return x**3 - 6 * x**2 + 11 * x - 6


def cubic_derivative(x):
    return 3 * x**2 - 12 * x + 11


initial_guess = 1.5
root_newton = newton_raphson_method(cubic_function, cubic_derivative, initial_guess)
print("Newton-Raphson Method Root:", root_newton)

```



```
PS D:\lifesucks\Python> & C:/Python312/python.exe d:/lifesucks/Python/python_1.3.py
################# Newton Raphson method output #################


################# Newton Raphson method output ends #################
Newton-Raphson Method Root: 3.0
PS D:\lifesucks\Python> 
```