

Name: Kassim Musa Abass
Reg: SPH32/3338/2025
Program: Embedded Systems Intro

Python Classes & Polymorphism

Objective:

To understand and implement Object-Oriented Programming (OOP) concepts in Python, specifically inheritance and polymorphism, by creating a class hierarchy for geometric shapes.

Introduction:

This assignment explores the creation of a base Shape class and derived classes (Circle, Square, Rectangle, RightTriangle). Each derived class implements its own overridden area() method to calculate the area based on specific dimensions.

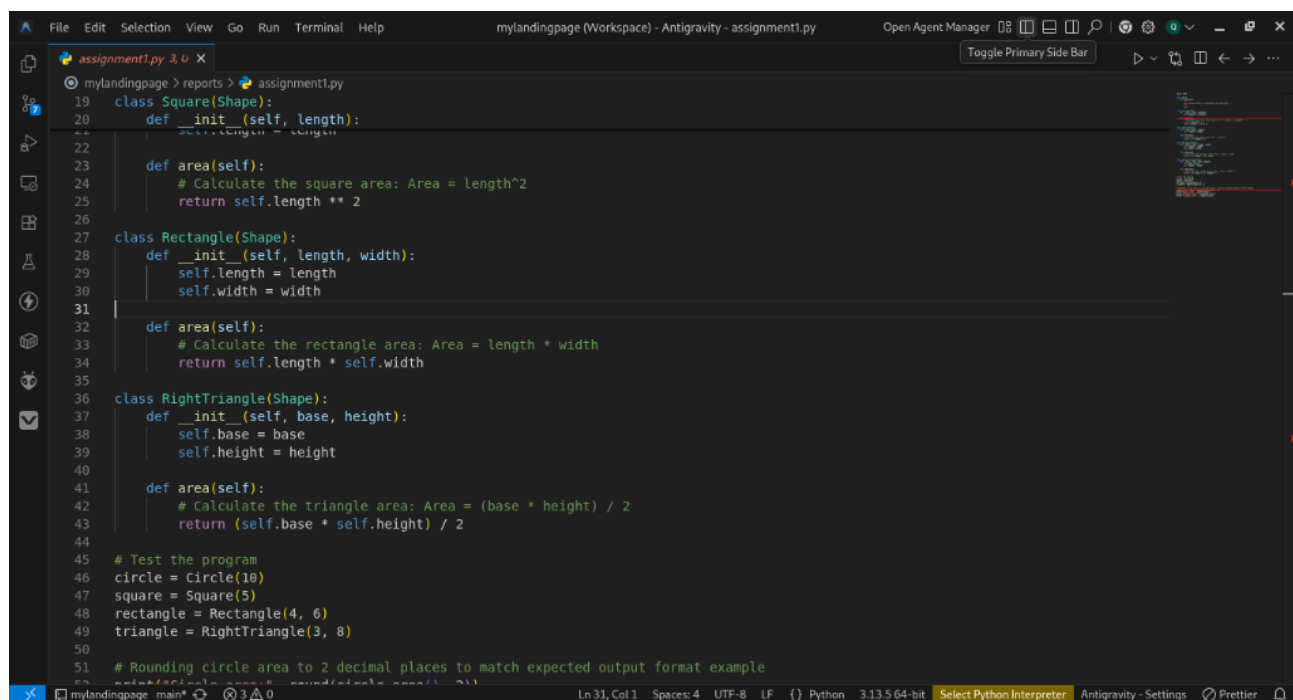
Method:

1. Documentation

The Shape class defines a generic area() method. Each subclass overrides this method using specific formulas: Circle ($\pi * r^2$), Square (length^2), Rectangle ($\text{length} * \text{width}$), and Right-Angled Triangle ($(\text{base} * \text{height}) / 2$). The program demonstrates polymorphism by calling the area() method on different shape objects.

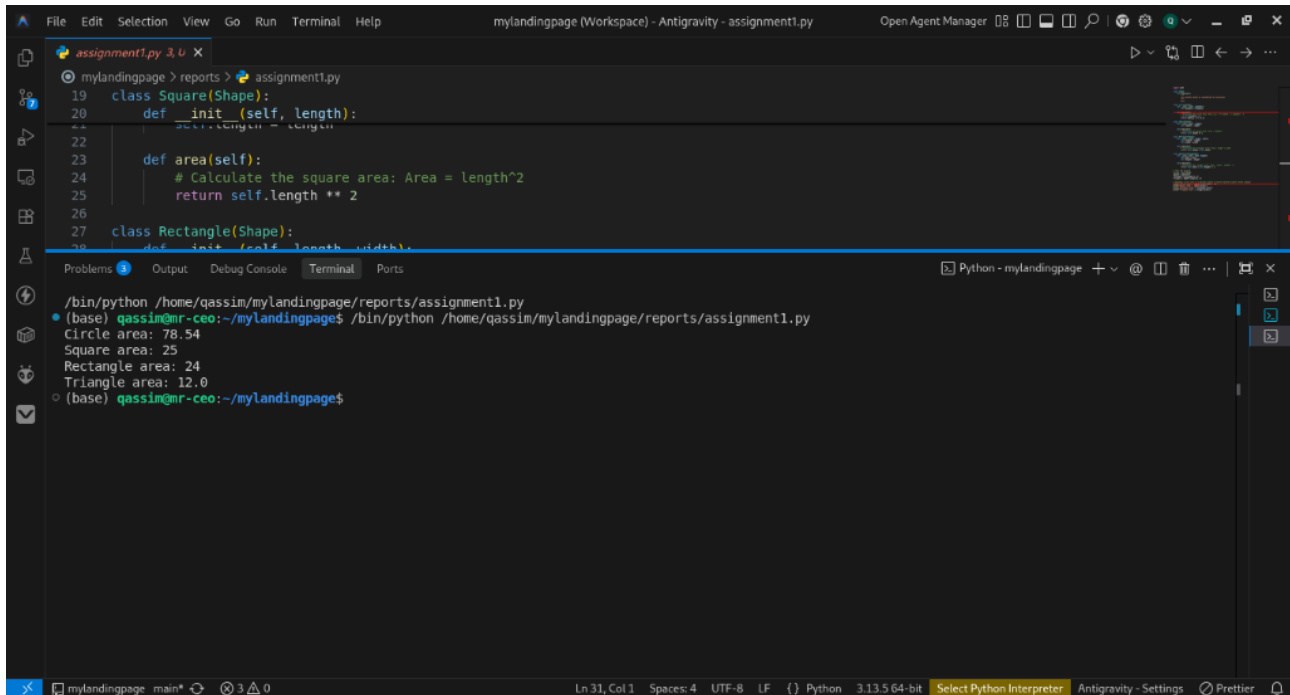
2. Code Listing

Source Code (assignment1.py):



```
19 class Shape:
20     def __init__(self, length):
21         self.length = length
22
23     def area(self):
24         # Calculate the square area: Area = length^2
25         return self.length ** 2
26
27 class Rectangle(Shape):
28     def __init__(self, length, width):
29         self.length = length
30         self.width = width
31
32     def area(self):
33         # Calculate the rectangle area: Area = length * width
34         return self.length * self.width
35
36 class RightTriangle(Shape):
37     def __init__(self, base, height):
38         self.base = base
39         self.height = height
40
41     def area(self):
42         # Calculate the triangle area: Area = (base * height) / 2
43         return (self.base * self.height) / 2
44
45 # Test the program
46 circle = Circle(10)
47 square = Square(5)
48 rectangle = Rectangle(4, 6)
49 triangle = RightTriangle(3, 8)
50
51 # Rounding circle area to 2 decimal places to match expected output format example
```

3. Testing and Results



The screenshot shows a VS Code editor window with a workspace named 'mylandingpage'. The file 'assignment1.py' is open, showing a Python script with two classes: 'Square' and 'Rectangle', both inheriting from a 'Shape' base class. The 'Square' class has an 'area' method that calculates the area as length^2 . The 'Rectangle' class has an 'area' method that calculates the area as $\text{length} \times \text{width}$. The script also includes a 'main' function that creates instances of these classes and prints their areas.

```
19 class Square(Shape):
20     def __init__(self, length):
21         self.length = length
22
23     def area(self):
24         # Calculate the square area: Area = length^2
25         return self.length ** 2
26
27 class Rectangle(Shape):
28     def __init__(self, length, width):
29         self.length = length
30         self.width = width
31
32     def area(self):
33         # Calculate the rectangle area: Area = length * width
34         return self.length * self.width
35
36 if __name__ == '__main__':
37     square = Square(5)
38     rectangle = Rectangle(4, 3)
39     circle = Circle(5)
40     triangle = Triangle(5, 10)
41
42     print(f"Circle area: {circle.area()}")
43     print(f"Square area: {square.area()}")
44     print(f"Rectangle area: {rectangle.area()}")
45     print(f"Triangle area: {triangle.area()}")
```

The terminal output shows the execution of the script, which prints the following results:

```
(base) qassim@mr-ceo:~/mylandingpage$ /bin/python /home/qassim/mylandingpage/reports/assignment1.py
Circle area: 78.54
Square area: 25
Rectangle area: 24
Triangle area: 12.0
(base) qassim@mr-ceo:~/mylandingpage$
```

4. Analysis, Discussion and Conclusion

The program successfully demonstrates inheritance and polymorphism. By defining a common interface in the Shape base class, we ensure that all derived shapes can be interacted with uniformly. The calculated areas matched the expected output, validating the mathematical formulas implemented in each subclass.