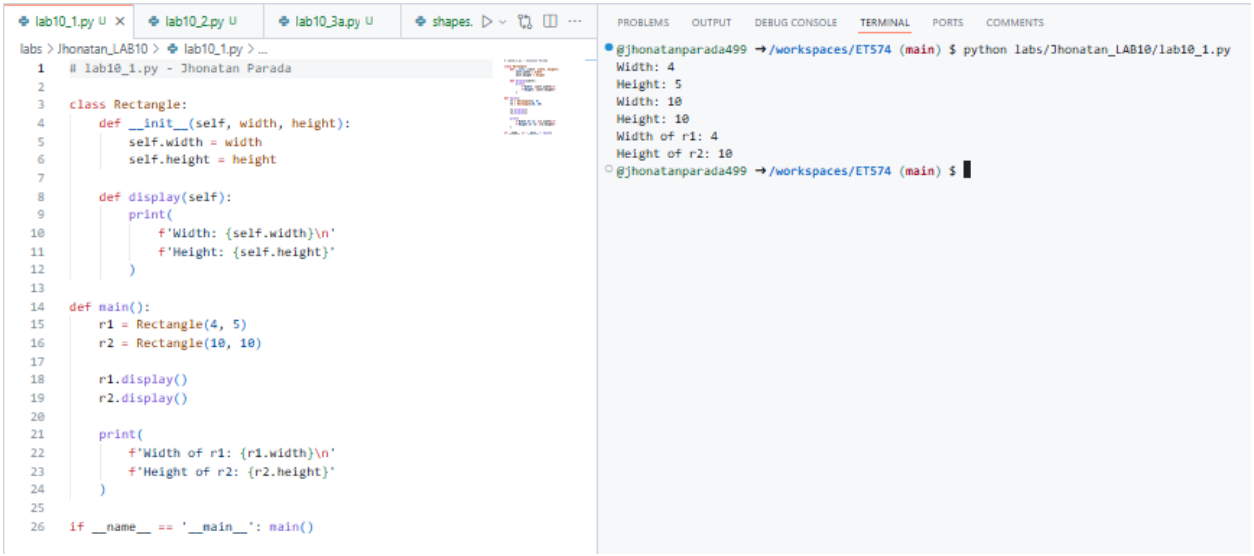
Lab 10 Summary

Jhonatan Parada

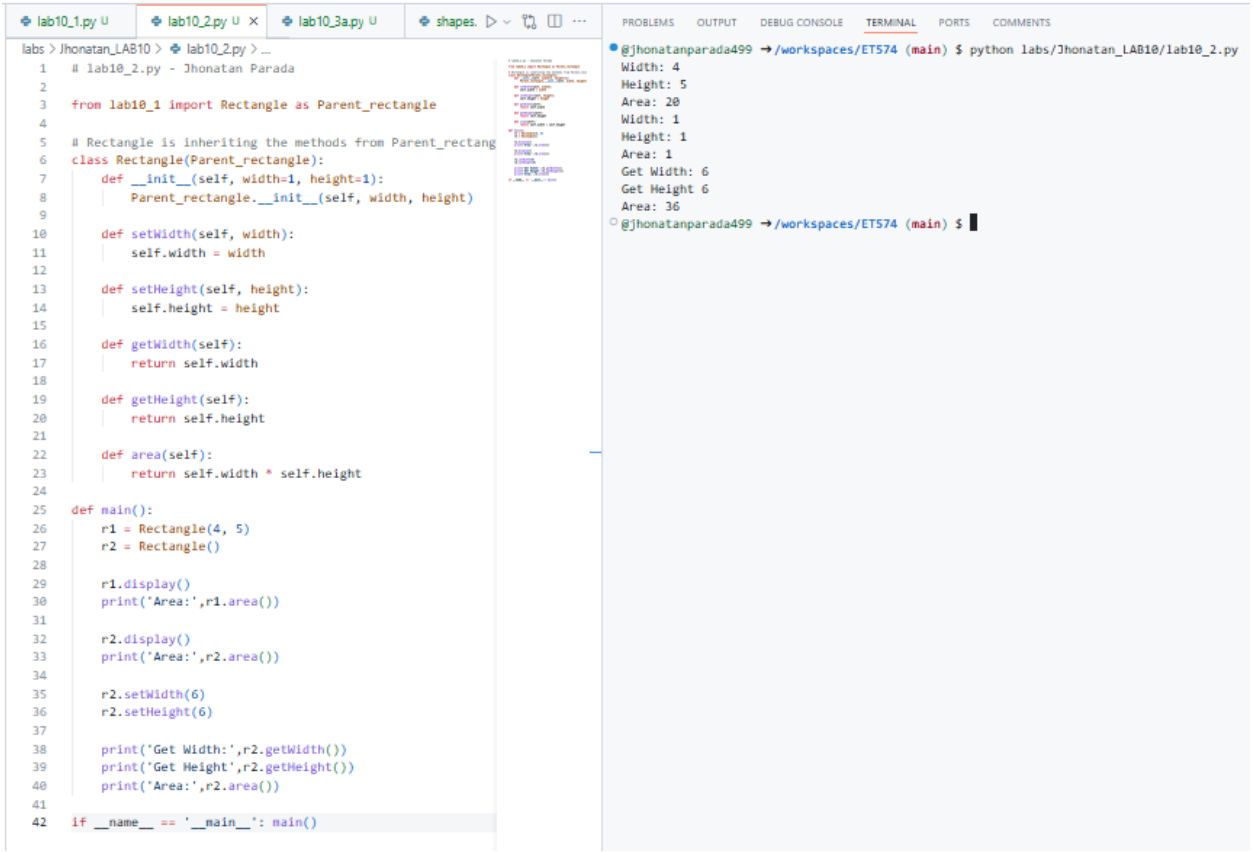
ET574

1.



```python
# lab10_1.py - Jhonatan Parada

class Rectangle:
    def __init__(self, width, height):
        self.width = width
        self.height = height

    def display(self):
        print(
            f'Width: {self.width}\n'
            f'Height: {self.height}'
        )

def main():
    r1 = Rectangle(4, 5)
    r2 = Rectangle(10, 10)

    r1.display()
    r2.display()

    print(
        f'Width of r1: {r1.width}\n'
        f'Height of r2: {r2.height}'
    )

if __name__ == '__main__': main()
```

```
@jhonatanparada499 ➜ /workspaces/ET574 (main) $ python labs/Jhonatan_LAB10/lab10_1.py
Width: 4
Height: 5
Width: 10
Height: 10
Width of r1: 4
Height of r2: 10
@jhonatanparada499 ➜ /workspaces/ET574 (main) $
```



```python
# lab10_2.py - Jhonatan Parada

from lab10_1 import Rectangle as Parent_rectangle

# Rectangle is inheriting the methods from Parent_rectang
class Rectangle(Parent_rectangle):
    def __init__(self, width=1, height=1):
        Parent_rectangle.__init__(self, width, height)

    def setWidth(self, width):
        self.width = width

    def setHeight(self, height):
        self.height = height

    def getWidth(self):
        return self.width

    def getHeight(self):
        return self.height

    def area(self):
        return self.width * self.height

def main():
    r1 = Rectangle(4, 5)
    r2 = Rectangle()

    r1.display()
    print('Area:',r1.area())

    r2.display()
    print('Area:',r2.area())

    r2.setWidth(6)
    r2.setHeight(6)

    print('Get Width:',r2.getWidth())
    print('Get Height',r2.getHeight())
    print('Area:',r2.area())

if __name__ == '__main__': main()
```

```
@jhonatanparada499 ➜ /workspaces/ET574 (main) $ python labs/Jhonatan_LAB10/lab10_2.py
Width: 4
Height: 5
Area: 20
Width: 1
Height: 1
Area: 1
Get Width: 6
Get Height 6
Area: 36
@jhonatanparada499 ➜ /workspaces/ET574 (main) $
```

lab10_1.py U | lab10_2.py U | lab10_3a.py U × | shapes.

labs > Jhonatan_LAB10 > lab10_3a.py > ...

```python
from math import pi
from lab10_2 import Rectangle

class Circle:
    def __init__(self, radius=1):
        self.radius = radius

    def display(self):
        print('Radius:',self.radius)

    def setRadius(self, radius):
        self.radius = radius

    def getRadius(self):
        return self.radius

    def area(self):
        return pi * (self.radius**2)

    def circumference(self):
        return 2 * pi * self.radius

def main():
    r = Rectangle()
    s = Circle(0)

    r.display()
    r.setWidth(1.25)
    r.setHeight(1.25)

    print('Get Width:',r.getWidth())
    print('Get Height:',r.getHeight())
    print(f'Area: {r.area():.5f}')
    print()

    s.display()
    s.setRadius(10)

    print('Get Radius:', s.getRadius())
    print(f'Area: {s.area():.5f}')
    print(f'Circumference: {s.circumference():.5f}')

if __name__ == '__main__': main()
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

```
@jhonatanparada499 ➜ /workspaces/ET574 (main) $ python labs/Jhonatan_LAB10/lab10_3a.py
Width: 1
Height: 1
Get Width: 1.25
Get Height: 1.25
Area: 1.56250

Radius: 0
Get Radius: 10
Area: 314.15927
Circumference: 62.83185
@jhonatanparada499 ➜ /workspaces/ET574 (main) $
```

0_1.py U | lab10_2.py U | lab10_3a.py U | shapes.py U ×

labs > Jhonatan_LAB10 > shapes.py

```python
# shapes.py - Jhonatan Parada

from lab10_2 import Rectangle
from lab10_3a import Circle
```

_2.py U | lab10_3a.py U | shapes.py U | lab10_3b.py U ×

labs > Jhonatan_LAB10 > lab10_3b.py > ...

```python
# lab10_3b.py - Jhonatan Parada

from shapes import Rectangle, Circle

def main():
    r = Rectangle()
    s = Circle(0)

    r.display()
    r.setWidth(1.25)
    r.setHeight(1.25)

    print('Get Width:',r.getWidth())
    print('Get Height:',r.getHeight())
    print(f'Area: {r.area():.5f}')
    print()

    s.display()
    s.setRadius(10)

    print('Get Radius:', s.getRadius())
    print(f'Area: {s.area():.5f}')
    print(f'Circumference: {s.circumference():.5f}')

if __name__ == '__main__': main()
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

```
@jhonatanparada499 ➜ /workspaces/ET574 (main) $ python labs/Jhonatan_LAB10/lab10_3b.py
Width: 1
Height: 1
Get Width: 1.25
Get Height: 1.25
Area: 1.56250

Radius: 0
Get Radius: 10
Area: 314.15927
Circumference: 62.83185
@jhonatanparada499 ➜ /workspaces/ET574 (main) $
```

2. I noticed something interesting with the default parameters in the class Rectangle in lab10_2.py. Since height and width were set to be 1 as default, that would form a 1x1 square, therefore, should that class still be called Rectangle?

On this lab, I did not follow the instructions strictly as I was asked, instead, I made use of a powerful concept in python called class inheritance. I did not see the need to create a rectangle.py file and import it to lab10_2.py because lab10_2.py has the class Rectangle already. By checking that the attribute __name__ is __main__ in lab10_2, I can run code in it while making it safe to import it to other modules. What I did see possibly useful was the implementation of shapes.py, since it would make more sense to add this class and other shapes classes to a directory called shapes for better management/organization/accessibility. To accomplish this, there is no need to create circle.py or rectangle.py since the classes already exist in the current lab files. Therefore, the only thing to do is to import Rectangle and Circle from the corresponding files that host these classes to shapes.py. In lab10_3b we then would write the import command as 'from shapes import Rectangle, Circle'.

I did not see the need to write 2 different versions of the class Rectangle, that's why I made the class Rectangle from lab10_2 to inherit all the methods (in this case just 1) and attributes (2) from the class Rectangle in lab10_1, so I did not have to rewrite the methods and attributes again, I did so by putting in practice classes inheritance. The class from which we want to inherit all of it must be put inside a parenthesis after declaring the name of the class we want to get the inheritance (child class) such as: 'Rectangle(Another_Class_Rectangle)' and then, in order to not overwrite the method __init__ from the parent class, we write the following under the method __init__ from the child class: Another_Class_Rectangle.__init__(self,x,y) or using the super function as in super().__init__(x,y). This way we can inherit the initial attributes set by the parent class.