PYTHON **Week 9 Lab 8**
Due **July 10, 2023 11:59pm**

READ LAB to see what has to be coded.

Part 1	Enter code in IDE Editor that would • Show Your full name • Show Student ID # • Show Date written • Show why program written	2 marks
Part 2	Write python code	3 marks
Part 3	Write python code	5 marks
Part 4	Write python code	10 marks

- Write completed code, using the supplied, to produce the output required.
- The code can be written into 1 file to produce the required outputs
- If you are creating 4 files, make sure Part 1 has been completed for all 4

SUBMISSION:

Upload python .py file to
Lab 8 - Inheritance, Polymorphism

PYTHON **Week 9 Lab 8**
Due **July 10, 2023 11:59pm**

Part 2 3 marks

```
class X(object):
    def __init__(self, a)
        self.num = a

    def doubleup(self):
        self.num *= 2

class Y(X):
    def _init_(self, a):
        X.__init__(self, a)

    def tripleup(self):
        self.num *= 3
```


Create the Code to produce OUTPUT

Output

3
9
27

PYTHON Week 9 Lab 8
Due July 10, 2023 11:59pm

Part 3 5 marks

```
# Base or Super class
class Person(object):
    def __init__(self, name):
        self.name = name

    def getName(self):
        return self,name

    def isEmployee(self):
        return False

# Inherited or Subclass (Note Person in bracket)
class Employee(Person):
    def __init__(self, name, eid):

        ''' In Python 3.0+, "super().__init__(name)"
            also works'''
        super(Employee, self).__init__(name)
        self.empID = eid

    def isEmployee(sefl):
        return True

    def getID(self):
        return self empID
```

 # Create the Code to produce OUTPUT

OUTPUT
("geek1", True, "E101")

PYTHON Week 9 Lab 8
Due July 10, 2023 11:59pm

Part 4 10 marks

```
import math

class Shape:
    def __init__(self, color='black', filled=False):
        self._color = Color
        self.__filled = filled

    def get_color(self)
        return self.__color

    def set_color(self, color):
        self.__color = color

    def get_filled(self:
        return self.__filled

    def set_filled(self, filled):
        self.__filled = filled

class Rectangle(Shape):
    def __init__(self, length, breadth):
        super().__init__)
        self.__length = length
        self.__breadth = breadth

    def get_length(self):
        return self.__length

    def set_length(self, lenth)
        self.__length = length

    def get_breadth(self):
        return self.__breadth

    def set_breadth(self, breadth):
        self.__breadth = breadth

    def get_area(self):
        return self.__length * self.__breadth

    def get_perimeter(self):
```

```
        return 2 * (self.__length + self.__breadth)

class Circle(Shape):
    def __init__(self, radius):
        super().__init__()
        self.__radius = radius

    def get_radius(self):
        return self.__radius

    def set_radius(self, radius):
        self.__radius = radius

    def get_area(self):
        return math.pi * self.__radius ** 2

    def get_perimeter(self):
        return 2 * math.pi * self.__radius
```

Create the Code to produce OUTPUT

Output:

```
Area of rectangle r1: 26.25
Perimeter of rectangle r1: 26.0
Color of rectangle r1: black
Is rectangle r1 filled ? False
Is rectangle r1 filled ? True
Color of rectangle r1: orange

Area of circle c1: 452.39
Perimeter of circle c1: 75.40
Color of circle c1: black
Is circle c1 filled ? False
Is circle c1 filled ? True
Color of circle c1: blue
```