Course Title: Programming Language II
Course Code: CSE 111
Semester: Summer 2020
Lab 8 Assignment

Task - 1

Given the following classes, write the code for the **BBA_Student** class so that the following output is printed:

```python
class Student:
    def __init__(self, name='Just a student', dept=nothing'):
        self.__name = name
        self.__department = dept
    def set_department(self, dept):
        self.__department = dept
    def get_name(self):
        return self.__name
    def set_name(self,name):
        self.__name = name
    def __str__(self):
        return 'Name: '+self.__name+' Department: '+self.__department



print(BBA_Student())
print(BBA_Student('Humpty Dumpty'))
print(BBA_Student('Little Bo Peep'))
```

Output

```
Name: default Department: BBA
Name: Humpty Dumpty Department: BBA
Name: Little Bo Peep Department: BBA
```

Task – 2

```python
class Vehicle:
    def __init__(self):
        self.x = 0
        self.y = 0
    def moveUp(self):
        self.y+=1
    def moveDown(self):
        self.y-=1
    def moveRight(self):
        self.x+=1
    def moveLeft(self):
        self.x-=1
    def __str__(self):
        return '('+str(self.x)+' , '+str(self.y)+')'

print('Part 1')
print('------')
car = Vehicle()
print(car)
car.moveUp()
print(car)
car.moveLeft()
print(car)
car.moveDown()
print(car)
car.moveRight()
print(car)
print('------')
print('Part 2')
print('------')
car1 = Vehicle2010()
print(car1)
car1.moveLowerLeft()
print(car1)
car2 = Vehicle2010()
car2.moveLeft()
print(car1.equals(car2))
car2.moveDown()
print(car1.equals(car2))
print('------')
```

```
Part 1
------
(0 , 0)
(0 , 1)
(-1 , 1)
(-1 , 0)
(0 , 0)
------
Part 2
------
(0 , 0)
(-1 , -1)
False
True
------
```

A vehicle assumes that the whole world is a 2-dimensional graph paper. It maintains its x and y coordinates (both are integers). The vehicle gets manufactured (constructed) at (0, 0) coordinate.

Subtasks:

1. Design a **Vehicle2010 class** which inherits movement methods from **Vehicle** and adds new methods called **move UpperRight, UpperLeft, LowerRight, LowerLeft.** Each of these diagonal move methods must re-use two inherited and appropriate move methods.
2. Write an “**equals”** method which tests if significant class properties are the same (in this case x and y).

Note: All moves are 1 step. That means a single call to any move method changes value of either x or y or both by 1.

<u>Task - 3</u>

Let’s Play with **Numbers**!!!

Write the **ComplexNumber** class so that the following code generates the output below.

```
class RealNumber:

    def __init__(self, r=0):
        self.__realValue = r
    def getRealValue(self):
        return self.__realValue
    def setRealValue(self, r):
        self.__realValue = r
    def ping(self):
        print('I am in RealNumber class')
    def __str__(self):
        return 'RealPart: '+str(self.getRealValue())


cn1 = ComplexNumber()
print(cn1)
print('---------')

cn2 = ComplexNumber(5,7)
print(cn2)
print('---------')
```

```
RealPart: 1.0
ImaginaryPart: 1.0
--------------------
RealPart: 5.0
ImaginaryPart: 7.0
--------------------
```

Task - 4

Write the **CheckingAccount** class so that the following code generates the output below:

`class Account:`
`    def __init__(self,balance):`
`        self._balance = balance`
 	
`    def getBalance(self):`
`        return self._balance`
 	
`print('Number of Checking Accounts: '+CheckingAccount.numberOfAccount)`
`print(CheckingAccount())`
`print(CheckingAccount(100.00))`
`print(CheckingAccount(200.00))`
`print('Number of Checking Accounts: '+CheckingAccount.numberOfAccount)`
	Number of Checking Accounts: 0
	Account Balance: 0.0
	Account Balance: 100.0
	Account Balance: 200.0
	Number of Checking Accounts: 3

Task - 5

Given the following classes, write the code for the **Dog** and the **Cat** class so that the following output is printed.

```python
class Animal:
    def __init__(self,sound):
        self.__sound = sound

    def makeSound(self):
        return self.__sound

class Printer:
    def printSound(self, a):
        print(a.makeSound())


d1 = Dog('bark')
c1 = Cat('meow')
a1 = Animal('Animal does not make sound')
pr = Printer()
pr.printSound(a1)
pr.printSound(c1)
pr.printSound(d1)
```

```
Animal does not make sound
meow
bark
```

Task - 6

Write the **Mango** and the **Jackfruit** classes so that the following code generates the output below:

```python
class Fruit:
    def __init__(self, formalin=False, name=''):
        self.__formalin = formalin
        self.name = name

    def getName(self):
        return self.name

    def hasFormalin(self):
        return self.__formalin

class testFruit:
    def test(self, f):
        print('----Printing Detail----')
        if f.hasFormalin():
            print('Do not eat the',f.getName(),'.')
            print(f)
        else:
            print('Eat the',f.getName(),'.')
            print(f)

m = Mango()
j = Jackfruit()
t1 = testFruit()
t1.test(m)
t1.test(j)
```

```
----Printing Detail-----
Do not eat the Mango.
Mangos are bad for you
----Printing Detail-----
Eat the Jackfruit.
Jackfruits are good for you
```