BRAC UNIVERSITY

Inspiring Excellence

Course Title: Programming Language II
Course Code: CSE 111
Lab Assignment no: 7

*** You are not allowed to change any of the code of the tasks*
*** Use **Inheritance** to solve all problems*

Task - 1

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

```
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

#write your code here

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)+')'
#write your code here

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))
```

OUTPUT:

```
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.

Task - 3

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

```
class Tournament:
    def __init__(self,name='Default'):
        self.__name = name
    def set_name(self,name):
        self.__name = name
    def get_name(self):
        return self.__name

#write your code here

ct1 = Cricket_Tournament()
print(ct1.detail())
print("----------------------")
ct2 = Cricket_Tournament("IPL",10,"t20")
print(ct2.detail())
print("----------------------")
tt = Tennis_Tournament("Roland Garros",128)
print(tt.detail())
```

```
OUTPUT:
Cricket Tournament Name: Default
Number of Teams: 0
Type: No type
----------------------
Cricket Tournament Name: IPL
Number of Teams: 10
Type: t20
----------------------
Tennis Tournament Name: Roland Garros
Number of Players: 128
```

Task - 4

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

```
class Product:
    def __init__(self,id, title, price):
        self.__id = id
        self.__title = title
        self.__price = price
    def get_id_title_price(self):
        return "ID: "+str(self.__id)+" Title:"+self.__title+
"Price: "+str(self.__price)

#write your code here

book = Book(1,"The Alchemist",500,"97806","HarperCollins")
print(book.printDetail())
print("-----------------------")
cd = CD(2,"Shotto",300,"Warfaze",50,"Hard Rock")
print(cd.printDetail())
```

OUTPUT:

```
ID: 1 Title: The Alchemist Price: 500
ISBN: 97806 Publisher: HarperCollins
-------------------------
ID: 2 Title: Shotto Price: 300
Band: Warfaze Duration: 50 minutes
Genre: Hard Rock
```

<u>Task - 5</u>

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

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

    def makeSound(self):
        return self.__sound


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

#write your code here

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

Task - 6

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

```
class Shape:

  def __init__(self, name='Default', height=0, base=0):
    self.area = 0
    self.name = name
    self.height = height
    self.base = base

  def get_height_base(self):
    return "Height: "+str(self.height)+",Base: "+str(self.base)

#write your code here

tri_default = triangle()
tri_default.calcArea()
print(tri_default.printDetail())
print('-------------------------')
tri = triangle('Triangle', 10, 5)
tri.calcArea()
print(tri.printDetail())
print('--------------------------')
trap = trapezoid('Trapezoid', 10, 6, 4)
trap.calcArea()
print(trap.printDetail())
```

OUTPUT:

```
Shape name: Default
Height: 0, Base: 0
Area: 0.0
---------------------------
Shape name: Triangle
Height: 10, Base: 5
Area: 25.0
---------------------------
Shape name: Trapezoid
Height: 10, Base: 6, Side_A: 4
Area: 50.0
```

Task - 7

Given the following classes, write the code for the **Player** and the **Manager** class so that the following output is printed. To calculate the match earning use the following formula:

1. Player: (total_goal * 1000) + (total_match * 10)
2. Manager: match_win * 1000

```
class Football:

  def __init__(self, team_name, name, role):
    self.__team = team_name
    self.__name = name
    self.role = role
    self.earning_per_match = 0

  def get_name_team(self):
    return 'Name: '+self.__name+', Team Name: ' +self.__team

#write your code here

player_one = Player('Juventus', 'Ronaldo', 'Striker', 25, 32)
player_one.calculate_ratio()
player_one.print_details()
print('----------------------------------------')
manager_one = Manager('Real Madrid', 'Zidane', 'Manager', 25)
manager_one.print_details()
```

OUTPUT:

```
Name: Ronaldo, Team Name: Juventus
Team Role: Striker
Total Goal: 25, Total Played: 32
Goal Ratio: 0.78125
Match Earning: 25320K
----------------------------------
Name: Zidane, Team Name: Real Madrid
Team Role: Manager
Total Win: 25
Match Earning: 25000K
```