Course Title: Programming Language II
Course Code: CSE 111
Lab Assignment no: 5

Task 1

Write a class called **Marks** with a required constructor. Then perform the addition using operator overloading

<table>
<tr>
<td>

#Write your code here

#Do not change the following lines of code
```
Q1 = Marks(int(input("Quiz 1 (out of 10): ")))
Q2 = Marks(int(input("Quiz 2 (out of 10): ")))
Lab = Marks(int(input("Lab (out of 30): ")))
Mid = Marks(int(input("Mid (out of 20): ")))
Final = Marks(int(input("Final (out of 30): ")))
total = Q1 + Q2 + Lab + Mid + Final
print("Total marks: {}".format(total.mark))
```

</td>
<td>

Sample Input 1:
Quiz 1 (out of 10): 10
Quiz 2 (out of 10): 10
Lab (out of 30): 30
Mid (out of 20): 20
Final (out of 30): 30

Sample Output 1:
Total marks: 100

Sample Input 2:
Quiz 1 (out of 10): 10
Quiz 2 (out of 10): 8
Lab (out of 30): 30
Mid (out of 20): 20
Final (out of 30): 29

Sample Output 2:
Total marks: 97

</td>
</tr>
</table>

Task 2

Design the program to get the output as shown.

Subtasks:

1. You will need to create 2 classes: **Teacher** and **Course**
2. Make all the variables in the Teacher class **private**.
3. Make all the variables in the Course class **public**.
4. Write the required codes in the Teacher and Course classes.

[You are not allowed to change the code below]

# Write your code here for subtasks 1-4	Output:
| ```
t1 = Teacher("Saad Abdullah", "CSE")
t2 = Teacher("Mumit Khan", "CSE")
t3 = Teacher("Sadia Kazi", "CSE")
c1 = Course("CSE 110 Programming Language I")
c2 = Course("CSE 111 Programming Language-II")
c3 = Course("CSE 220 Data Structures")
c4 = Course("CSE 221 Algorithms")
c5 = Course("CCSE 230 Discrete Mathematics")
c6 = Course("CSE 310 Object Oriented
Programming")
c7 = Course("CSE 320 Data Communications")
c8 = Course("CSE 340 Computer Architecture")
t1.addCourse(c1)
t1.addCourse(c2)
t2.addCourse(c3)
t2.addCourse(c4)
t2.addCourse(c5)
t3.addCourse(c6)
t3.addCourse(c7)
t3.addCourse(c8)
t1.printDetail()
t2.printDetail()
t3.printDetail()
``` | ```
====================================
Name:  Saad Abdullah
Department:  CSE
List of courses
====================================
CSE 110 Programming Language I
CSE 111 Programming Language-II
====================================
====================================
Name:  Mumit Khan
Department:  CSE
List of courses
====================================
CSE 220 Data Structures
CSE 221 Algorithms
CCSE 230 Discrete Mathematics
====================================
====================================
Name:  Sadia Kazi
Department:  CSE
List of courses
====================================
CSE 310 Object Oriented Programming
CSE 320 Data Communications
CSE 340 Computer Architecture
====================================
``` |

Task 3

Design the program to get the output as shown.

Subtasks:

1. You will need to create 2 classes: **Team** and **Player**
2. Make all the variables in the Team class **private**.
3. Make all the variables in the Player class **public**.
4. Write the required codes in the Team and Player classes

Hints:

- Create a list in team class to store the player's name in that list
- Use constructor overloading technique for Team class

[You are not allowed to change the code below]

# Write your code here for subtasks 1-4	Output:

`b = Team()`
`b.setName('Bangladesh')`
`mashrafi = Player("Mashrafi")`
`b.addPlayer(mashrafi)`
`tamim = Player("Tamim")`
`b.addPlayer(tamim)`
`b.printDetail()`
`a = Team("Australia")`
`ponting = Player("Ponting")`
`a.addPlayer(ponting)`
`lee = Player("Lee")`
`a.addPlayer(lee)`
`a.printDetail()`
	`====================`
	`Team: Bangladesh`
	`List of Players:`
	`['Mashrafi', 'Tamim']`
	`====================`
	`====================`
	`Team: Australia`
	`List of Players:`
	`['Ponting', 'Lee']`
	`====================`

Task 4

Look at the code and the sample inputs and outputs below to design the program accordingly.

1. Write a class called **Color** that only adds the 3 primary colors (red, blue and yellow).
2. Write a required constructor for the class.
3. You have to use operator overloading to get the desired outputs as shown.

Hint:

There will be only one constructor and only one method tackling the addition operation. No other methods are required.
Note: Order of the color given as input should not matter. For example, in sample input 1, if the first input was yellow and then red, the output would still be orange.

#*Write your code here*	**Sample Input 1:**
	First Color: red
#Do not change the following lines of code	Second Color: yellow
C1 = Color(input("First Color: ").lower())	**Sample Output 1:**
C2 = Color(input("Second Color: ").lower())	Color formed: Orange
C3 = C1 + C2	
print("Color formed:", C3.clr)	**Sample Input 2:**
	First Color: red
	Second Color: blue
	Sample Output 2:
	Color formed: Violet
	Sample Input 3:
	First Color: yellow
	Second Color: BLUE
	Sample Output 3:
	Color formed: Green

Task 5

Write a class called Circle with the required constructor and methods to get the following output.

Subtasks:

1. Create a **class** called Circle.

2. Create the required **constructor**. Use **Encapsulation** to protect the variables. [**Hint:** Assign the variables in **private**]

3. Create **getRadius()** and **setRadius()** method to access variables.

4. Create a **method** called area to calculate the area of circles.

5. Handle the **operator overloading** by using a **special method** to calculate the radius and area of circle 3.

[You are not allowed to change the code below]

# Write your code here for subtasks 1-5	Output:
c1 = Circle(4) print("First circle radius:" , c1.getRadius()) print("First circle area:" ,c1.area()) c2 = Circle(5) print("Second circle radius:" ,c2.getRadius()) print("Second circle area:" ,c2.area()) c3 = c1 + c2 print("Third circle radius:" ,c3.getRadius()) print("Third circle area:" ,c3.area())	First circle radius: 4 First circle area: 50.26548245743669 Second circle radius: 5 Second circle area: 78.53981633974483 Third circle radius: 9 Third circle area: 254.46900494077323

Task 6

Write a class called Triangle with the required constructor and methods to get the following output.

Subtasks:

1. Create a **class** called Triangle.

2. Create the required **constructor**. Use **Encapsulation** to protect the variables. [**Hint:** Assign the variables in **private**]

3. Create **getBase(), getHeight(), setBase** and **setHeight()** method to access variables.

4. Create a **method** called area to calculate the area of triangles.

5. Handle the **operator overloading** by using a **special method** to calculate the radius and area of triangle 3.

[You are not allowed to change the code below]

# Write your code here for subtasks 1-5 t1 = Triangle(10, 5) print("First Triangle Base:" , t1.getBase()) print("First Triangle Height:" , t1.getHeight()) print("First Triangle area:" ,t1.area()) t2 = Triangle(5, 3) print("Second Triangle Base:" , t2.getBase()) print("Second Triangle Height:" , t2.getHeight()) print("Second Triangle area:" ,t2.area()) t3 = t1 - t2 print("Third Triangle Base:" , t3.getBase()) print("Third Triangle Height:" , t3.getHeight()) print("Third Triangle area:" ,t3.area())	**Output:** First Triangle Base: 10 First Triangle Height: 5 First Triangle area: 25.0 Second Triangle Base: 5 Second Triangle Height: 3 Second Triangle area: 7.5 Third Triangle Base: 5 Third Triangle Height: 2 Third Triangle area: 5.0

Task 7

Observe the given code carefully. Try to understand from the given code and the outputs what to write in your class **Dolls**.

# Write your code here	Output:
| ```python
obj_1 = Dolls("Tweety", 2500)
print(obj_1.detail())
if obj_1 > obj_1:
    print("Congratulations! You get the Tweety as a gift!")
else:
    print("Thank you!")

print("=========================")
obj_2 = Dolls("Daffy Duck", 1800)
print(obj_2.detail())
if obj_2 > obj_1:
    print("Congratulations! You get the Tweety as a gift!")
else:
    print("Thank you!")

print("=========================")
obj_3 = Dolls("Bugs Bunny", 3000)
print(obj_3.detail())
if obj_3 > obj_1:
    print("Congratulations! You get the Tweety as a gift!")
else:
    print("Thank you!")

print("=========================")
obj_4 = Dolls("Porky Pig", 1500)
print(obj_4.detail())
if obj_4 > obj_1:
    print("Congratulations! You get the Tweety as a gift!")
else:
    print("Thank you!")

print("=========================")
obj_5 = obj_2 + obj_3
print(obj_5.detail())
if obj_5 > obj_1:
    print("Congratulations! You get the Tweety as a gift!")
else:
    print("Thank you!")
``` | Doll: Tweety<br>Total Price: 2500 taka<br>Thank you!<br>=========================<br>Doll: Daffy Duck<br>Total Price: 1800 taka<br>Thank you!<br>=========================<br>Doll: Bugs Bunny<br>Total Price: 3000 taka<br>Congratulations! You get the Tweety as a gift!<br>=========================<br>Doll: Porky Pig<br>Total Price: 1500 taka<br>Thank you!<br>=========================<br>Dolls: Daffy Duck Bugs Bunny<br>Total Price: 4800 taka<br>Congratulations! You get the Tweety as a gift! |

[You are not allowed to change the code above]

Subtasks:

1. Create a Doll class.
2. Create the required constructor.
3. Write a method to print the name and the price of the object
4. Use operator overloading for the addition operators.
5. Write a method to handle operator overloading for the ">" logical operator that compares the price of the objects.

Hints*:*

- *Notice that the price of each object is being checked with the price of obj in the given code.*
- *Notice the word Doll in the first 4 outputs and the last output. You have to print exactly as represented here.*

Task 8

Design a class called **Coordinates** with an appropriate constructor. Then perform the subtraction, multiplication and equality check operations in the given code using operator overloading.

#*Write your code here*	Sample Input 1:
#Do not change the following lines of code	1
	2
	3
p1 = Coordinates(int(input()),int(input()))	4
p2 = Coordinates(int(input()),int(input()))	**Sample Output 1:**
	(-2,-2)
p4 = p1 - p2	(3,8)
print(p4.detail())	The calculated coordinates are NOT the same.
	Sample Input 2:
p5 = p1 * p2	0
print(p5.detail())	0
	0
point_check = (p4 == p5)	0
print(point_check)	**Sample Output 2:**
	(0,0)
	(0,0)
	The calculated coordinates are the same.

Task 9

```python
class Test:
    def __init__(self):
        self.sum = 0
        self.y = 0

    def methodA(self):
        x=0
        y =0
        y = y + 7
        x = y + 11
        self.sum = x + y
        print(x , y, self.sum)

    def methodB(self):
        x = 0
        self.y = self.y + 11
        x = x + 33 + self.y
        self.sum = self.sum + x + self.y
        print(x , self.y, self.sum)
```

t1 = Test() t1.methodA() t1.methodA() t1.methodB() t1.methodB()	X	y	sum

Task 10

```python
class Test3:
    def __init__(self):
        self.sum, self.y = 0, 0

    def methodA(self):
        x, y = 2, 3
        msg = [0]
        msg[0] = 3
        y = self.y + msg[0]
        self.methodB(msg, msg[0])
        x = self.y + msg[0]
        self.sum = x + y + msg[0]
        print(x, y, self.sum)

    def methodB(self, mg2: list, mg1: int):
        x = 0
        self.y = self.y + mg2[0]
        x = x + 33 + mg1
        self.sum = self.sum + x + self.y
        mg2[0] = self.y + mg1
        mg1 = mg1 + x + 2
        print(x, self.y, self.sum)
```

t3 = Test3() t3.methodA() t3.methodA() t3.methodA() t3.methodA() t3.methodA() t3.methodA()	x	y	sum

Task 11

```python
class Test4:
    def __init__(self):
        self.sum, self.y = 0, 0

    def methodA(self):
        x, y = 0, 0
        msg = [0]
        msg[0] = 5
        y = y + self.methodB(msg[0])
        x = y + self.methodB(msg, msg[0])
        self.sum = x + y + msg[0]
        print(x, y, self.sum)

    def methodB(self, *args):
        if len(args) == 1:
            mg1 = args[0]
            x = 0
            y = 0
            y = y + mg1
            x = x + 33 + mg1
            self.sum = self.sum + x + y
            self.y = mg1 + x + 2
            print(x, y, self.sum)
            return y
        else:
            mg2, mg1 = args
            x = 0
            self.y = self.y + mg2[0]
            x = x + 33 + mg1
            self.sum = self.sum + x + self.y
            mg2[0] = self.y + mg1
            mg1 = mg1 + x + 2
            print(x, self.y, self.sum)
            return self.sum
```

t3 = Test3() t3.methodA() t3.methodA() t3.methodA() t3.methodA() t3.methodA()	x	y	sum