Inspiring Excellence

Course Code:	**CSE111**
Course Title:	**Programming Language II**
Topic:	**OOP (Encapsulation+Multi class)**
Number of tasks:	**6**

Task 1

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.

[You are not allowed to change the code below]

# Write your code here for subtasks 1-5 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())	**Output:** First circle radius: 4 First circle area: 50.26548245743669 Second circle radius: 5 Second circle area: 78.53981633974483

Task 2

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()** methods to access variables.

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

[You are not allowed to change the code below]

# Write your code here for subtasks 1-5	**Output:**
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())	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

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

Class Description:

Spaceship: This class represents a spaceship. Each spaceship has a **name** and a **capacity** (the maximum weight it can carry).

Cargo: This class represents a piece of cargo. Each cargo item has a **name** and a **weight**. Both attributes should be **private** which means they cannot be accessed directly from outside of the class.

A `Spaceship` contains (HAS) `Cargo`. That means each spaceship can carry multiple cargo items, but the total weight of the cargo cannot exceed the spaceship's capacity.

Your task is to design the **Spaceship** and **Cargo** class with necessary properties so that the given output is produced for the provided driver code.

Driver Code	Output

`# Creating spaceships`
`falcon = Spaceship("Falcon", 50000)`
`apollo = Spaceship("Apollo", 100000)`
`enterprise = Spaceship("Enterprise", 220000)`
`print("1.==========")`
`# Creating cargo`
`gold = Cargo("Gold", 20000)`
`platinum = Cargo("Platinum", 25000)`
`dilithium = Cargo("Dilithium", 50000)`
`trilithium = Cargo("Trilithium", 70000)`
`neutronium = Cargo("Neutronium", 80000)`
`print("2.==========")`
`# Loading cargo onto spaceships`
`falcon.load_cargo(gold)`
`falcon.load_cargo(platinum)`
`falcon.display_details()`
`print("3.==========")`
`apollo.load_cargo(gold)  # Apollo will not reach its total capacity`
`apollo.display_details()`
`print("4.==========")`
`falcon.load_cargo(neutronium)  # This should exceed Falcon's capacity`
	1.==========
	2.==========
	Spaceship Name: Falcon
	Capacity: 50000
	Current Cargo Weight: 45000
	Cargo: ['Gold', 'Platinum']
	3.==========
	Spaceship Name: Apollo
	Capacity: 100000
	Current Cargo Weight: 20000
	Cargo: ['Gold']
	4.==========
	Warning: Unable to load Neutronium inside Falcon. Exceeds capacity by 75000.
	5.==========
	Spaceship Name: Enterprise
	Capacity: 220000
	Current Cargo Weight: 200000
	Cargo: ['Dilithium', 'Trilithium', 'Neutronium']

```
print("5.==================================")
enterprise.load_cargo(dilithium)
enterprise.load_cargo(trilithium)
enterprise.load_cargo(neutronium)  # This
should not exceed Enterprise's capacity
enterprise.display_details()
```

Task 5

Design the required class/es so that the following output is generated. Read the following description:

1. You may assume that to board a bus, a student must have the bus pass, and his/her destination must match the route of the bus.
2. Additionally, the default maximum capacity of the bus is 2.

Driver Code	Output
st1 = BracuStudent("Afif", "Mirpur") print("1==========") st2 = BracuStudent("Shanto", "Motijheel") st3 = BracuStudent("Taskin", "Mirpur") st1.show_details() st2.show_details() print("2==========") st3.show_details() print("3==========") bus1 = BracuBus("Mirpur") bus2 = BracuBus("Azimpur", 5) bus1.show_details() bus2.show_details() print("4==========") st2.get_pass() st3.get_pass() print("5==========") st2.show_details()	1========== Student Name: Afif Lives in Mirpur Have Bus Pass? False Student Name: Shanto Lives in Motijheel Have Bus Pass? False 2========== Student Name: Taskin Lives in Mirpur Have Bus Pass? False 3========== Bus Route: Mirpur Passengers Count: 0 (Max: 2) Passengers On Board: [] Bus Route: Azimpur Passengers Count: 0 (Max: 5) Passengers On Board: [] 4========== 5========== Student Name: Shanto Lives in Motijheel Have Bus Pass? True Student Name: Taskin Lives in Mirpur

st3.show_details() print("6==========") bus1.board() print("7==========") bus1.board(st1, st2) print("8==========") st1.get_pass() st2.home = "Mirpur" st1.show_details() st2.show_details() print("9==========") bus1.board(st1, st2, st3) print("10==========") bus1.show_details()	Have Bus Pass? True 6========== No passengers! 7========== You don't have a bus pass! You got on the wrong bus! 8========== Student Name: Afif Lives in Mirpur Have Bus Pass? True Student Name: Shanto Lives in Mirpur Have Bus Pass? True 9========== Afif boarded the bus. Shanto boarded the bus. Bus is full! 10========== Bus Route: Mirpur Passengers Count: 2 (Max: 2) Passengers On Board: ['Afif', 'Shanto']

Task 6

Design the required class/es so that the following output is generated.

Read the following description:

- The Library class has two dictionaries: one contains borrower information(the name of borrowers and the number of books they borrowed) and the other contains book availability information (book type and their remaining number)
- A reader cannot borrow more than 5 books.
- If a book's availability is 0 in the Library, then the reader cannot borrow that book.
- The readerInfo method in the Reader class prints the type and the number of all books borrowed if no parameter is passed, else it prints the number of books borrowed of the specific type mentioned in the parameter. You may use the default argument for this.

Driver Code	Output
L1=Library('Dhaka',{'Arts':15,'Fiction':135,'Politics':2,'Science':11,'Poetry':15}) L1.details() print("1----------")	Dhaka Library details Borrower details: {} Books availability: {'Arts': 15, 'Fiction': 135, 'Politics': 2, 'Science': 11, 'Poetry': 15} 1----------

r1=Reader('Aladdin') r1.borrow(L1,'Arts','Fiction','Fiction','Politics') print("2----------") r1.borrow(L1,'Politics','Fiction') print("3----------") r1.readerInfo() print("4----------") r1.readerInfo('Fiction') print("5----------") L1.details() print("6----------") r2=Reader('Jasmine') r2.borrow(L1,'Politics','Poetry') print("7----------") r2.readerInfo() print("8----------") L1.details()	Arts book is borrowed successfully. Fiction book is borrowed successfully. Fiction book is borrowed successfully. Politics book is borrowed successfully. 2---------- Politics book is borrowed successfully. You cannot borrow more than 5 books. 3---------- Aladdin, you have 5 book(s) with you. Books on Arts: 1 Books on Fiction: 2 Books on Politics: 2 4---------- Aladdin, you have 2 Fiction book(s) with you. 5---------- Dhaka Library details Borrower details: {'Aladdin': 5} Books availability: {'Arts': 14, 'Fiction': 133, 'Politics': 0, 'Science': 11, 'Poetry': 15} 6---------- Politics books are not available at the moment. Poetry book is borrowed successfully. 7---------- Jasmine, you have 1 book(s) with you. Books on Poetry: 1 8---------- Dhaka Library details Borrower details: {'Aladdin': 5, 'Jasmine': 1} Books availability: {'Arts': 14, 'Fiction': 133, 'Politics': 0, 'Science': 11, 'Poetry': 14}