Lab Assignment 05

Inspiring Excellence

Course Code:	**CSE111**
Course Title:	**Programming Language II**
Topic:	**Instance Variable, and Instance Method**
Number of Tasks:	**11**

[You are not allowed to change the driver codes of any of the tasks]

Task 1

Design the **Course** class to generate the correct output from the driver code provided below:

Course Class:

```
public class Course{
  public String cName;
  public String code;
  public int credit;
  // Write your code here
}
```

Driver Code	Output

public class Tester1{
  public static void main(String[] args) {
  Course c1 = new Course();
  Course c2 = new Course();
 	
  System.*out*.println("========== 1 ==========");
  c1.createCourse("Programming Language I", "CSE110", 3);
  c1.displayCourse();
 	
  System.*out*.println("========== 2 ==========");
  c2.createCourse("Data Structures", "CSE220", 3);
  c2.displayCourse();
 	
  System.*out*.println("========== 3 ==========");
  c1.updateCourse("Programming Language II", "CSE111", 3);
  c1.displayCourse();
  }
}
	========== 1 ==========
	Course Name:
	Programming Language I
	Course Code: CSE110
	Course Credit: 3
	========== 2 ==========
	Course Name: Data
	Structures
	Course Code: CSE220
	Course Credit: 3
	========== 3 ==========
	Course Name:
	Programming Language II
	Course Code: CSE111
	Course Credit: 3

Task 2

Create a **Dog** class so that the tester code generates the given output:

Driver Code	Expected Output

`public class Tester2{`
`  public static void main (String[] args) {`
`  Dog scooby = new Dog();`
`  Dog oldie = new Dog();`
`  Dog goofy = new Dog();`
 	
`  scooby.changeName("Scooby");`
`  goofy.changeName("Goofy");`
 	
`  System.out.println("1. ==========");`
`  System.out.println(scooby.bark());`
`  System.out.println("2. ==========");`
`  System.out.println(oldie.bark());`
`  System.out.println("3. ==========");`
`  oldie.changeColor("White");`
`  System.out.println("4. ==========");`
`  System.out.println(oldie.bark());`
`  System.out.println("5. ==========");`
`  System.out.println(goofy.bark());`
`  System.out.println("6. ==========");`
`  scooby.changeColor("Brown");`
`  System.out.println("7. ==========");`
`  System.out.println(scooby.bark());`
`  System.out.println("8. ==========");`
`  goofy.changeColor("Black");`
`  }`
`}`
	`1. ==========`
	`Scooby is barking`
	`2. ==========`
	`A dog is barking`
	`3. ==========`
	`This dog is White`
	`4. ==========`
	`White dog is barking`
	`5. ==========`
	`Goofy is barking`
	`6. ==========`
	`Scooby is Brown`
	`7. ==========`
	`Scooby the Brown dog is barking`
	`8. ==========`
	`Goofy is Black`

Task 3

Create an **Employee** class to provide the expected output.

- An employee will have a name, salary and designation.
- The name will be assigned inside the newEmployee() method

- Whenever a New Employee joins his/her salary will be **Tk. 30,000** and the designation will be **junior**.
- Employees with salaries greater than **Tk. 50,000** and **Tk. 30,000** need to pay **30%** and **10%** of salary as tax respectively.
- Employees can be promoted to **senior**, **lead** and **manager** positions. Based on their promotion they will get an increment of **Tk. 25,000**, **Tk. 50,000** and **Tk. 75,000** respectively.

Driver Code	Expected Output

public class Tester3{
  public static void main(String[] args){
  Employee emp1 = new Employee();
  Employee emp2 = new Employee();
  Employee emp3 = new Employee();
 	
  emp1.newEmployee("Harry Potter");
  emp2.newEmployee("Hermione Granger");
  emp3.newEmployee("Ron Weasley");
  System.*out*.println("1 ==========");
  emp1.displayInfo();
  System.*out*.println("2 ==========");
  emp2.displayInfo();
  System.*out*.println("3 ==========");
  emp3.displayInfo();
  System.*out*.println("4 ==========");
  emp1.calculateTax();
  System.*out*.println("5 ==========");
  emp1.promoteEmployee("lead");
  System.*out*.println("6 ==========");
  emp1.calculateTax();
  System.*out*.println("7 ==========");
  emp1.displayInfo();
  System.*out*.println("8 ==========");
  emp3.promoteEmployee("manager");
  System.*out*.println("9 ==========");
  emp3.calculateTax();
  System.*out*.println("10 ==========");
  emp3.displayInfo();
  }
}
	1 ==========
	Employee Name: Harry Potter
	Employee Salary: 30000.0 Tk
	Employee Designation: junior
	2 ==========
	Employee Name: Hermione Granger
	Employee Salary: 30000.0 Tk
	Employee Designation: junior
	3 ==========
	Employee Name: Ron Weasley
	Employee Salary: 30000.0 Tk
	Employee Designation: junior
	4 ==========
	No need to pay tax
	5 ==========
	Harry Potter has been promoted to lead
	New Salary: 80000.00 Tk
	6 ==========
	Harry Potter Tax Amount: 24000.00 Tk
	7 ==========
	Employee Name: Harry Potter
	Employee Salary: 80000.0 Tk
	Employee Designation: lead
	8 ==========
	Ron Weasley has been promoted to manager
	New Salary: 105000.00 Tk
	9 ==========
	Ron Weasley Tax Amount: 31500.00 Tk
	10 ==========
	Employee Name: Ron Weasley
	Employee Salary: 105000.0 Tk
	Employee Designation: manager

Task 4

You are building a tracker system that will keep track of a person's income and expenses.

- When the ***createTracker()*** method is invoked it sets the balance to 1.0 taka.
- The ***info()*** method **returns** a String with the trackers information.
- If the total balance becomes 0 after the ***expense()*** method is called it prints "You're broke!" . Again if the available balance is less than the expense it prints "Not enough balance.". Otherwise the method prints "Balance updated" after updating the balance.
- The last expense and income history can be seen by using the ***history()*** method.

Driver Code	Output

public class Tester4{
  public static void main(String[] args) {
  MoneyTracker tr1 = new MoneyTracker();
  System.*out*.println(tr1.info());
  tr1.createTracker("John");
  System.*out*.println("1 ==========");
  System.*out*.println(tr1.info());
  System.*out*.println("2 ==========");
  tr1.income(1000);
  System.*out*.println(tr1.info());
  System.*out*.println("3 ==========");
  tr1.expense(800);
  tr1.expense(100);
  System.*out*.println(tr1.info());
  System.*out*.println("4 ==========");
  tr1.showHistory();
  System.*out*.println("5 ==========");
  tr1.expense(101);
  System.*out*.println("6 ==========");
  tr1.expense(200);
  System.*out*.println("7 ==========");
  tr1.income(200);
  tr1.showHistory();
  System.*out*.println("8 ==========");
  }
}
	Name: null
	Current Balance: 0.0
	1 ==========
	Name: John
	Current Balance: 1.0
	2 ==========
	Balance Updated!
	Name: John
	Current Balance: 1001.0
	3 ==========
	Balance Updated.
	Balance Updated.
	Name: John
	Current Balance: 101.0
	4 ==========
	Last added: 1000.0
	Last spent: 100.0
	5 ==========
	You're broke!
	6 ==========
	Not enough balance.
	7 ==========
	Balance Updated!
	Last added: 200.0
	Last spent: 100.0
	8 ==========

Task 5

Create a **MagicItem** class to provide the expected output. A character will have a name, energy level, and three individual magic items (item1, item2, and item3).

- The name will be assigned inside the **newCharacter()** method. Whenever a new character is created, they will start with 0 energy and no magic items.
- Characters can find and use magic items, each with a specific energy boost. Magic items include "Potion" (+50), "Elixir" (+100), and "Amulet" (+200).
- Characters can use a magic item if they have it, which increases their energy level.

Driver Code	Output

public class StrangerMagic {
  public static void main(String[] args){
  MagicItem char1 = new MagicItem();
  MagicItem char2 = new MagicItem();
 	
  char1.newCharacter("Eleven");
  char2.newCharacter("Mike Wheeler");
 	
  System.*out*.println("1 ==========");
  char1.displayInfo();
  System.*out*.println("2 ==========");
  char2.displayInfo();
  System.*out*.println("3 ==========");
  char1.findItem("Potion");
  char1.findItem("Elixir");
  char1.findItem("Elixir");
  char2.findItem("Potion");
  System.*out*.println("4 ==========");
  char1.findItem("Amulet");
  System.*out*.println("5 ==========");
  char1.displayInfo();
  System.*out*.println("6 ==========");
  char1.useItem("Potion");
  char1.useItem("Elixir");
  System.*out*.println("7 ==========");
  char1.displayInfo();
  System.*out*.println("8 ==========");
  char1.findItem("Amulet");
  System.*out*.println("9 ==========");
	1 ==========
	Character Name: Eleven
	Energy Level: 0
	Item 1: null
	Item 2: null
	Item 3: null
	2 ==========
	Character Name: Mike Wheeler
	Energy Level: 0
	Item 1: null
	Item 2: null
	Item 3: null
	3 ==========
	Eleven found a Potion
	Eleven found a Elixir
	Eleven found a Elixir
	Mike Wheeler found a Potion
	4 ==========
	All item slots occupied.
	5 ==========
	Character Name: Eleven
	Energy Level: 0
	Item 1: Potion
	Item 2: Elixir
	Item 3: Elixir
	6 ==========
	Eleven used a Potion
	Energy Level after using item: 50
	Eleven used a Elixir
	Energy Level after using item: 150
	7 ==========
	Character Name: Eleven
	Energy Level: 150
	Item 1: null
	Item 2: null

```
    char1.displayInfo();
    System.out.println("10 ==========");
    char2.useItem("Amulet");
    System.out.println("11 ==========");
    char2.displayInfo();
  }
}
```

```
Item 3: Elixir
8 ==========
Eleven found a Amulet
9 ==========
Character Name: Eleven
Energy Level: 150
Item 1: Amulet
Item 2: null
Item 3: Elixir
10 ==========
Item not in inventory.
11 ==========
Character Name: Mike Wheeler
Energy Level: 0
Item 1: Potion
Item 2: null
Item 3: null
```

Task 6

Complete the following **Cart** class to generate the given output from the tester code:

- A cart will have a cart number which will be assigned in ***create_cart()*** method.
- Each cart can hold up to 3 items (at max).
- Each cart must have two arrays to store items and their respective prices.
- The items inside a cart will be added in ***addItem()*** method only if the cart items do not exceed 3.
- The ***giveDiscount()*** method saves the discount given to that cart object and updates the price accordingly.

Driver Code	Expected Output

public class Tester6{
  public static void main(String [] args){
  Cart c1 = new Cart ();
  Cart c2 = new Cart ();
  Cart c3 = new Cart ();
 	
  c1.create_cart(1);
  c2.create_cart(2);
	====1====
	Table added to cart 1.
	You have 1 item(s) in your cart now.
	Chair added to cart 1.
	You have 2 item(s) in your cart now.
	Television added to cart 1.
	You have 3 item(s) in your cart now.
	You already have 3 items on your cart
	====2====
	Stove added to cart 2.

```
    c3.create_cart(3);
    System.out.println("====1====");
    c1.addItem("Table", 3900.5);
    c1.addItem("Chair", 1400.76);
    c1.addItem("Television", 5400.87);
    c1.addItem("Refrigerator", 5000);

    System.out.println("====2====");
    c2.addItem("Stove",439.90);

    System.out.println(""====3====");
    c3.addItem("Chair",1400.5);
    c3.addItem("Chair",3400);

    System.out.println(""====4====");
    c1.cartDetails();

    System.out.println(""====5====");
    c2.cartDetails();

    System.out.println(""====6====");
    c3.cartDetails();
    c1.giveDiscount(10);

    System.out.println(""====7====");
    c1.cartDetails();
  }
}
```

```
You have 1 item(s) in your cart now.
====3====
Chair added to cart 3.
You have 1 item(s) in your cart now.
Chair added to cart 3.
You have 2 item(s) in your cart now.
====4====
Your cart(c1) :
Table - 3900.5
Chair - 1400.76
Television - 5400.87
Discount Applied: 0.0%
Total price: 10702.130000000001
====5====
Your cart(c2) :
Stove - 439.9
Discount Applied: 0.0%
Total price: 439.9
====6====
Your cart(c3) :
Chair - 1400.5
Chair - 3400.0
Discount Applied: 0.0%
Total price: 4800.5
====7====
Your cart(c1) :
Table - 3900.5
Chair - 1400.76
Television - 5400.87
Discount Applied: 10.0%
Total price: 9631.917000000001
```

Task 7

Design the **Reader** class in such a way so that the following code provides the expected output.

- A reader will have a name, capacity to read and an array of books they are reading.
- The initial capacity of a reader will be 0. The initial name will be "New user".
- A new array is created every time a reader's capacity is increased, which replaces the initial array.

Driver Code	Expected Output

public class Reader_tester {
  public static void main(String[] args){
  Reader r1 = new Reader();
  Reader r2 = new Reader();
 	
  r1.createReader("Messi", 2);
  r2.createReader("Ronaldo", 5);
 	
  System.*out*.println("1 ==========");
  r1.readerInfo();
 	
  System.*out*.println("2 ==========");
  r2.addBook("Java");
  r2.addBook("Python");
  r2.addBook("C++");
  r2.readerInfo();
 	
  System.*out*.println("3 ==========");
  r1.addBook("C#");
  r1.addBook("Rust");
  r1.addBook("GoLang");
 	
  System.*out*.println("4 ==========");
  r1.increaseCapacity(5);
  r1.addBook("Python");
 	
  System.*out*.println("5 ==========");
  r1.readerInfo();
  }
}
	1 ==========
	Name: Messi
	Capacity: 2
	Books:
	No books added yet
	2 ==========
	Name: Ronaldo
	Capacity: 5
	Books:
	Book 1: Java
	Book 2: Python
	Book 3: C++
	3 ==========
	No more space for new book
	4 ==========
	Messi's capacity increased to 5
	5 ==========
	Name: Messi
	Capacity: 5
	Books:
	Book 1: C#
	Book 2: Rust
	Book 3: Python

Task 8

You are building a ride booking app called UberApp. Using this app, a customer can book 3 rides.

- ***BookRide(Location, Distance)*** method books rides for a user and prints the fare for that ride based on the distance. After booking the ride, fare will be calculated as below:

  Fare = 30 * distance
- A person can change the location of their last booked ride using ***changeLocation(Location, Distance)*** method. The new fare is calculated as;

  Fare = 30 * distance + 20% of new Fare. i.e. If, new Fare = 210, then the total fare after changing location will be 210 + 210 * 0.2 = 252
- The UberApp keeps track of all the locations visited by the user in an array of String.
- The ***resetMonth()*** method resets the location visited in a month as well as the number of remaining rides of that month.

Design the **UberApp class** that will produce the following output.

Driver Code	Output

public class AppTester {
  public static void main(String args[]){
 	
  UberApp account1 = new UberApp();
  UberApp account2 = new UberApp();
 	
  account1.createProfile("Jonas Kahnwald", 24, "017111111111");
  account2.createProfile("Martha Nielsen", 28, "018111111111");
 	
  account1.showProfile();
  System.*out*.println("===== 1 ====");
  System.*out*.println("You have "+ account1.remainingRides() +" ride(s) remaining.");
 	
  System.*out*.println("==== 2 ====");
  account2.showProfile();
  System.*out*.println("You have "+ account2.remainingRides() +" ride(s) remaining.");
	Hello! This is your Profile:
	Full Name: Jonas Kahnwald
	Age: 24
	Phone Number: 017111111111
	===== 1 ====
	You have 3 ride(s) remaining.
	==== 2 ====
	Hello! This is your Profile:
	Full Name: Martha Nielsen
	Age: 28
	Phone Number: 018111111111
	You have 3 ride(s) remaining.
	==== 3 ====
	Jonas Kahnwald has booked a ride!
	Destination: Merul Badda
	Fare: 360.0 Taka
	==== 4 ====
	Jonas Kahnwald has booked a ride!
	Destination: Dhanmondi 27
	Fare: 129.0 Taka
	Jonas Kahnwald has changed the destination of his current ride to Wari
	New fare after adding 20% change fees: 201.6 Taka.

```
        System.out.println("==== 3 ====");
        account1.bookRide("Merul Badda", 12.0);

        System.out.println("==== 4 ====");
        account1.bookRide("Dhanmondi 27", 4.3);
        account1.changeLocation("Wari", 5.6);

        System.out.println("==== 5 ====");
        account1.ridingHistory();

        System.out.println("==== 6 ====");
        account2.ridingHistory();

        System.out.println("==== 7 ====");
        account1.bookRide("Banani 11", 6.8);
        account1.bookRide("Gulshan 1", 2.1);

        System.out.println("==== 8 ====");
        account1.resetMonth();
        account1.bookRide("Gulshan 1", 2.1);
        account1.ridingHistory();
        System.out.println("You have "+
account1.remainingRides() +" ride(s) remaining.");
    }
}
```

```
==== 5 ====
Jonas Kahnwald, you have visited
Merul Badda, Wari this month.
==== 6 ====
Martha Nielsen, you haven't visited
anywhere this month.
==== 7 ====
Jonas Kahnwald has booked a ride!
Destination: Banani 11
Fare: 204.0 Taka
Jonas Kahnwald, please update your
plan to premium or wait till next
month!
==== 8 ====
Jonas Kahnwald has booked a ride!
Destination: Gulshan 1
Fare: 63.0 Taka
Jonas Kahnwald, you have visited
Gulshan 1 this month.
You have 2 ride(s) remaining.
```

Task 9

```
1  public class Task09 {
2     public int p = 3, y = 2, sum;
3     public void methodA(){
4         int x = 0, y = 0;
5         y = y + this.y;
6         x = sum + 2 + p;
7         sum = x + y + methodB(p, y);
8         System.out.println(x + " " + y+ " " + sum);
9     }
10    public int methodB(int p, int n){
11        int x = 0;
12        y = y + (++p);
13        x = x + 2 + n;
14        sum = sum + x + y;
15        System.out.println(x + " " + y+ " " + sum);
16        return sum;
17    }
18 }
```

Driver code:

```
public class Tester09 {
   public static void main(String [] args){
       Task09 t1 = new Task09 ();
       t1.methodA();
       t1.methodA();
   }
}
```

Outputs		
x	**y**	**Sum**

Task 10

```
1  public class test1 {
2     public int x = 3;
3     public int y = 0;
4     public int z = -1;
5     public void case1(int x){
6         int y = 12;
7         this.x = y + 4 + x;
8         y += this.y +1;
9         case2(this.y, z);
10        System.out.println(x + " "+ y + " "+ z);
11        this.y = this.x + z;
12        System.out.println(this.x + " "+ this.y + " "+ this.z);
13    }
14    public void case2(int temp, int z){
15        this.x = z + temp + this.z;
16        this.z = y + z;
17        System.out.println(x + " "+ y + " "+ z);
18        y  = x + y + z;
19        temp = x;
20        x = this.z;
21        this.z = temp;
22        System.out.println(this.x + " "+ this.y + " "+ this.z);
23    }
24 }
```

Driver code:

```
public class test1Driver {
   public static void main(String [] args){
       test1 t1 = new test1();
       t1.case1(4);
       t1.case2(5, 6);
   }
}
```

Outputs		

Task 11

```
1  public class Task11 {
2     public int temp = 4;
3     public int sum;
4     public int y;
5     public int x;
6     public void methodA(int m){
7         int [] n = {2,5};
8         int x = 0;
9         y = y + m + this.methodB(x,m)+(temp++)+y;
10        x = this.x + 2 + (++n[0]);
11        sum = sum + x + y;
12        n[0] = sum + 2;
13       System.out.println(n[0] + x + " " + y+ " " + sum);
```

```
14          }
15      public int methodB(int m, int n){
16          int [] y = {1};
17          this.y = y[0] + this.y + m;
18          x = this.y + 2 + temp - n;
19          sum = x + y[0] + this.sum;
20          System.out.println(y[0]+ x + " " + y[0] + " " +sum);
21          return y[0];
22      }
23  }
```

```
public class Tester11 {
    public static void main(String [] args){
        Task11 t1 = new Task11();
        t1.methodA(5);
        t1.methodA(3);
    }
}
```

Outputs		