Lab Assignment 06

Inspiring Excellence

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

[Submit all the Coding Tasks in the Google Form shared on buX before the next lab. Submit the Tracing Tasks handwritten to your Lab Instructors at the beginning of the lab]

Task 1

Implement the "**Assignment**" class with necessary properties, so that the given output is produced for the provided driver code.

Driver Class	Output

public class AssignmentTester{
  public static void main(String [] args){
  Assignment as1 = new Assignment();
  as1.printDetails();
  System.out.println("1---------------");
  as1.tasks = 11;
  as1.difficulty = "Moderate";
  as1.submission = true;
  as1.printDetails();
  System.out.println("2---------------");
  as1.makeOptional();
  System.out.println("3---------------");
  as1.printDetails();
  System.out.println("4---------------");
  Assignment as2 = new Assignment();
  as2.tasks = 12;
  as2.difficulty = "Hard";
  as2.submission = false;
  as2.printDetails();
  System.out.println("5---------------");
  as2.makeOptional();
  }
}
	Number of tasks: 0
	Difficulty level: null
	Submission required: false
	1---------------
	Number of tasks: 11
	Difficulty level: Moderate
	Submission required: true
	2---------------
	Assignment will not require submission
	3---------------
	Number of tasks: 11
	Difficulty level: Moderate
	Submission required: false
	4---------------
	Number of tasks: 12
	Difficulty level: Hard
	Submission required: false
	5---------------
	Submission is already not required

Task 2

Implement the "**Shelf**" class with necessary properties, so that the given output is produced for the provided driver code.

Driver Class	Output

public class ShelfTester{
  public static void main(String [] args){
  Shelf shelf = new Shelf();
  shelf.showDetails();
  System.out.println("1---------------");
  shelf.addBooks(3);
  System.out.println("2---------------");
  shelf.capacity = 7;
  shelf.addBooks(3);
  System.out.println("3---------------");
  shelf.showDetails();
  System.out.println("4---------------");
  shelf.addBooks(5);
  shelf.showDetails();
  shelf.capacity += 4;
  System.out.println("6---------------");
  shelf.addBooks(5);
  shelf.showDetails();
  }
}
	Shelf capacity: 0
	Number of books: 0
	1---------------
	Zero capacity. Cannot add books.
	2---------------
	3 books added to shelf
	3---------------
	Shelf capacity: 7
	Number of books: 3
	4---------------
	Exceeds capacity
	Shelf capacity: 7
	Number of books: 3
	6---------------
	5 books added to shelf
	Shelf capacity: 11
	Number of books: 8

Task 3

Implement the "**LightController**" class with necessary properties to produce the given output for the provided driver code.

Driver Class	Output

public class LightControllerTester{
  public static void main(String args []){
  LightController c1 = new LightController();
  c1.showLightStatus();
  System.out.println("1---------------");
  c1.adjustBrightness(4);
  c1.switchLight();
  System.out.println("2---------------");
  c1.showLightStatus();
  System.out.println("3---------------");
  c1.adjustBrightness(4);
  System.out.println("4---------------");
  c1.showLightStatus();
  System.out.println("5---------------");
  c1.adjustBrightness(-2);
  c1.adjustBrightness(9);
  System.out.println("6---------------");
  c1.showLightStatus();
  System.out.println("7---------------");
  System.out.println(c1.resetSettings());
  c1.showLightStatus();
  System.out.println("8---------------");
  c1.switchLight();
  System.out.println("9---------------");
  c1.showLightStatus();
  }
}
	Light status: OFF
	Brightness Level: 0
	1---------------
	Please turn on the light first!
	Lights are now ON.
	2---------------
	Light status: ON
	Brightness Level: 1
	3---------------
	Brightness adjusted.
	4---------------
	Light status: ON
	Brightness Level: 5
	5---------------
	Brightness adjusted.
	Brightness out of range. Set between 0 to 10.
	6---------------
	Light status: ON
	Brightness Level: 3
	7---------------
	Light settings have been reset.
	Light status: ON
	Brightness Level: 1
	8---------------
	Lights are now OFF.
	9---------------
	Light status: OFF
	Brightness Level: 0

Task 4

Implement the "**ChickenBurger**" class with necessary properties, so that the given output is produced for the provided driver code.
[**Note:**
1. There are four available spice levels: **Mild**, **Spicy**, **Naga** and **Extreme**. You can store these values in a String array.
2. You might need to use the *.equals()* method to compare two string values.]

Driver Class	Output

public class BurgerMaker{
  public static void main(String [] args){
  ChickenBurger b1 = new ChickenBurger();
  System.out.println(b1.bun);
  System.out.println(b1.price);
  System.out.println(b1.sauceOption);
  System.out.println(b1.spiceLevel);
  System.out.println("----------1----------");
  System.out.println(b1.serveBurger());
  System.out.println("----------2----------");
  b1.customizeSpiceLevel("Extreme Jhaal");
  b1.customizeSpiceLevel("Spicy");
  System.out.println("----------3----------");
  System.out.println(b1.serveBurger());
  System.out.println("----------4----------");
  ChickenBurger b2 = new ChickenBurger();
  b2.bun = "Brioche";
  b2.price += 50;
  b2.sauceOption = "Regular";
  b2.customizeSpiceLevel("Naga");
  System.out.println("----------5----------");
  System.out.println(b2.serveBurger());
  }
}
	Sesame
	200
	Less
	Not Set
	----------1----------
	Cannot serve now. Customize Spice Level first.
	----------2----------
	This spice level is unavailable.
	Spice level set to Spicy.
	----------3----------
	The burger is being served:-
	Bun Type: Sesame
	Price: 200
	Sauce Option: Less
	Spice Level: Spicy
	----------4----------
	Spice level set to Naga.
	----------5----------
	The burger is being served:-
	Bun Type: Brioche
	Price: 250
	Sauce Option: Regular
	Spice Level: Naga

Task 5

Implement the "**MobilePhone**" class with necessary properties to produce the given output for the provided driver code.

Driver Class	Output

public class MobilePhoneTester{
  public static void main(String args []){
  MobilePhone m1 = new MobilePhone();
  MobilePhone m2 = new MobilePhone();
  m1.setContactCapacity(5);
  m2.setContactCapacity(100);
  m1.details();
  System.out.println("1----------");
  m1.addContact("John", 9866);
  m1.addContact("Maria", 7865);
  System.out.println("2----------");
  m1.makeCall(9866);
  System.out.println("3----------");
  m1.addContact("Henry", 2365);
  System.out.println("4----------");
  m1.makeCall(7552);
  m1.makeCall(2365);
  System.out.println("5----------");
  m1.addContact("Gomes", 4589);
  m1.addContact("Antony", 8421);
  m1.addContact("Tony", 5789);
  System.out.println("6----------");
  m1.details();
  }
}
	Total Contacts: 0
	Contact List:
	1----------
	The contact of John is added.
	The contact of Maria is added.
	2----------
	Calling John . . .
	3----------
	The contact of Henry is added.
	4----------
	Calling 7552 . . .
	Calling Henry . . .
	5----------
	The contact of Gomes is added.
	The contact of Antony is added.
	Storage Full!!
	6----------
	Total Contacts: 5
	Contact List:
	John:9866
	Maria:7865
	Henry:2365
	Gomes:4589
	Antony:8421

Task 6

Implement the "**Course**" class with necessary properties, so that the given output is produced for the provided driver code.
[Note: Each course can have at max 4 contents in its syllabus]

Driver Class	Output

public class CourseTester{
  public static void main(String [] args){
  Course c1 = new Course();
  c1.createCourse("PL II", "CS11");
  System.out.println("--------1--------");
  c1.printDetails();
  System.out.println("--------2--------");
  c1.addOneContent("Overloading");
  c1.printDetails();
  System.out.println("--------3--------");
  c1.addOneContent("Encapsulation");
  c1.addTwoContent("Static", "Polymorphism");
  c1.printDetails();
  System.out.println("--------4--------");
  c1.addOneContent("Inheritance");
  System.out.println("--------5--------");
  Course c2 = new Course();
  c2.createCourse("DS", "CS22");
  c2.addOneContent("Stack");
  c2.addTwoContent("Recursion","Tree");
  c2.addTwoContent("Heap","Hashing");
  System.out.println("--------6--------");
  c2.printDetails();
  }
}
	--------1--------
	Course details:
	Course Name: PL II
	Course Code: CS11
	Course Syllabus:
	No content yet.
	--------2--------
	Overloading was added.
	Course details:
	Course Name: PL II
	Course Code: CS11
	Course Syllabus:
	Overloading
	--------3--------
	Encapsulation was added.
	Static was added.
	Polymorphism was added.
	Course details:
	Course Name: PL II
	Course Code: CS11
	Course Syllabus:
	Overloading, Encapsulation, Static, Polymorphism
	--------4--------
	Cannot add more content
	--------5--------
	Stack was added.
	Recursion was added.
	Tree was added.
	Heap was added.
	Cannot add more content
	--------6--------
	Course details:
	Course Name: DS
	Course Code: CS22
	Course Syllabus:
	Stack, Recursion, Tree, Heap

Task 7

Task 7 looks very much similar to Task 6. But there are some slight differences. Can you figure those out? Once you figure those out, write the differences as a comment in your code and create **"Course2"** class.

Driver Class	Output

public class CourseTester2{
  public static void main(String [] args){
  Course2 c1 = new Course2();
  c1.createCourse("PL II", "CS11");
  System.out.println("--------1--------");
  c1.printDetails();
  System.out.println("--------2--------");
  c1.addContent("Overloading");
  c1.printDetails();
  System.out.println("--------3--------");
  c1.addContent("Encapsulation");
  c1.addContent("Static", "Polymorphism");
  c1.printDetails();
  System.out.println("--------4--------");
  c1.addContent("Inheritance");
  System.out.println("--------5--------");
  Course2 c2 = new Course2();
  c2.createCourse("DS", "CS22");
  c2.addContent("Stack");
  c2.addContent("Recursion","Tree");
  c2.addContent("Heap","Hashing");
  System.out.println("--------6--------");
  c2.printDetails();
  }
}
	--------1--------
	Course details:
	Course Name: PL II
	Course Code: CS11
	Course Syllabus:
	No content yet.
	--------2--------
	Overloading was added.
	Course details:
	Course Name: PL II
	Course Code: CS11
	Course Syllabus:
	Overloading
	--------3--------
	Encapsulation was added.
	Static was added.
	Polymorphism was added.
	Course details:
	Course Name: PL II
	Course Code: CS11
	Course Syllabus:
	Overloading, Encapsulation, Static, Polymorphism
	--------4--------
	Cannot add more content
	--------5--------
	Stack was added.
	Recursion was added.
	Tree was added.
	Heap was added.
	Cannot add more content
	--------6--------
	Course details:
	Course Name: DS
	Course Code: CS22
	Course Syllabus:
	Stack, Recursion, Tree, Heap

Task 8

Implement the "**Shape**" class with necessary properties to produce the given output for the provided driver code.

Driver Class	Output

public class ShapeTester{
  public static void main(String args []){
  Shape circle = new Shape();
  Shape triangle = new Shape();
  Shape trapezium = new Shape();
 	
  circle.setParameters("Circle", 5);
  triangle.setParameters("Triangle", 4, 7);
  trapezium.setParameters("Trapezium", 2, 4, 9);
 	
  System.out.println(circle.details());
  System.out.println("---------------");
  System.out.println(triangle.details());
  System.out.println("---------------");
  System.out.println(trapezium.details());
  }
}
	Shape Name: Circle
	Area: 78.54
	--------------- 	
	Shape Name: Triangle
	Area: 14.0
	--------------- 	
	Shape Name: Trapezium
	Area: 27.0

Task 9

```
1  public class Test1{
2    public int sum;
3    public int y;
4    public void methodA(){
5      int x=2, y =3;
6      int [] msg ={3, 7};
7      y = this.y + msg[0];
8      methodB(msg[1]++, msg[0]);
9      x = x + this.y + msg[1];
10     sum = x + y + msg[0];
11     System.out.println(x + " " + y+ " " + sum);
12   }
13   public void methodB(int mg2, int mg1){
14     int x = 0;
15     y = this.y + mg2;
16     x = x + 19 + mg1;
17     sum = this.sum + x + y;
18     mg2 = y + mg1;
19     mg1 = mg2 + x + 2;
20     System.out.println(x + " " + y+ " " + sum);
21   }
22 }
```

```
public class Tester1{
  public static void main (String args[]){
    Test1 t1 = new Test1();
    t1.methodB(5,-8);
    Test1 t2 = new Test1();
    t2.methodA();
  }
}
```

Outputs		

Task 10

```
1  public class Test2 {
2      int x = 3, y = 1, z = -4;
3      double p = 2.5;
4      public void methodA(int n, int x) {
5          this.x = methodB(x, n);
6          p = this.x + n % x * 2.0;
7          y = (z++) + methodB(z, (int) p) + (++z);
8          System.out.println(this.x + " " + (n + y) + " " + (x + z)) ;
9      }
10     public int methodB(int q, int n) {
11         int arr[] = {2, -5, 6};
12         arr[0] = arr[2] - this.x + n;
13         arr[1] = q - arr[1];
14         arr[2] = arr[q % 3] + arr[n % 2];
15         System.out.println(arr[0] + " " + arr[1] + " " + arr[2]) ;
16         return arr[1] + arr[2] - arr[0];
17     }
18 }
```

```
public class Tester2{
    public static void main(String [] args){
        Test2 t = new Test2();
        t.methodA(3, 4);
    }
}
```

Outputs		