

# Lab Assignment 02



Inspiring Excellence

Course Code:	CSE111
Course Title:	Programming Language II
Topic:	OOP Basics, Instance Variable and Instance Method
Number of Tasks:	12 (Classwork: 06, Homework: 06)

*[Submit all the Coding Tasks (Homework: Task 1 to 4) in the Google Form shared on buX before the next lab. Submit the Tracing Tasks (Homework: Task 5 to 6) handwritten to your Lab Instructors at the beginning of the lab]*

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

## CLASSWORK

### Task 1

You are given the following “University” class:

```
public class University{  
    public String name;  
    public String country;  
}
```

Now write a Java **tester** class named “UniversityTester”.

- a. Write the main method and create 2 objects of **University** class and print the location of the objects and print the instance variables of the objects. Are the location of the objects the same?
- b. Now change the instance variables of the first object.  
name = “Imperial College London”  
country = “England”

Now change the instance variables of the second object.

name = “BRAC University”  
country = “Bangladesh”

Now check if the instance variables of both objects have changed or not and whether the instance variables of both objects are of the same value or not.

### Task 2

Design the “Student” class so that the main method prints the following:

Tester Class	Output
<pre>public class StudentTester2{     public static void main(String [] args){         Student s1 = new Student();         System.out.println("Name of the Student: "+s1.name);         System.out.println("ID of the Student: "+s1.id);         s1.name = "Bob";         s1.id = 123;         System.out.println("Name of the Student: "+s1.name);         System.out.println("ID of the Student: "+s1.id);     } }</pre>	Name of the Student: Default ID of the Student: 0 Name of the Student: Bob ID of the Student: 123

### Task 3

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

Driver Code	Output
<pre>public class Tester3{     public static void main(String[] args) {         Course c1 = new Course();         Course c2 = new Course();         c1.updateDetails("Programming Language I", "CSE110", 3);         System.out.println("===== 1 =====");         c1.displayCourse();         c2.updateDetails("Data Structures", "CSE220", 3);         System.out.println("===== 2 =====");         c2.displayCourse();         c1.updateDetails("Programming Language II", "CSE111", 3);         System.out.println("===== 3 =====");         c1.displayCourse();     } }</pre>	<pre>===== 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</pre>

## Task 4

Design the **CellPhone** class so that the **main** method of tester class can produce the following output:

Tester Class	Output
<pre>public class Tester4{     public static void main(String[]args){         CellPhone phone1 = new CellPhone();         phone1.printDetails();         phone1.model ="Nokia 1100";         System.out.println("1#####         phone1.storeContact("Joy - 01834");         System.out.println("=====");         phone1.printDetails();         System.out.println("2#####         phone1.storeContact("Toya - 01334");         phone1.storeContact("Aayan - 01135");         System.out.println("=====");         phone1.printDetails();         System.out.println("3#####         phone1.storeContact("Sani - 01441");         System.out.println("=====");         phone1.printDetails();     } }</pre>	<pre>Phone Model unknown Contacts Stored 0 1##### Contact Stored ===== Phone Model Nokia 1100 Contacts Stored 1 Stored Contacts: Joy - 01834 2##### Contact Stored Contact Stored ===== Phone Model Nokia 1100 Contacts Stored 3 Stored Contacts: Joy - 01834 Toya - 01334 Aayan - 01135 3##### Memory full. New contact can't be stored. ===== Phone Model Nokia 1100 Contacts Stored 3 Stored Contacts: Joy - 01834 Toya - 01334 Aayan - 01135</pre>

## Task 5

Consider the following class:

```
public class Human{  
    public int age;  
    public double height;  
}
```

**Show the output of the following sequence of statements:**

Code	Output

## Task 6

1	<code>public class Task11 {</code>
2	<code>    public int p = 3, y = 2, sum;</code>
3	<code>    public void methodA(){</code>
4	<code>        int x = 0, y = 0;</code>
5	<code>        y = y + this.y;</code>
6	<code>        x = sum + 2 + p;</code>
7	<code>        sum = x + y + methodB(p, y);</code>
8	<code>        System.out.println(x + " " + y+ " " + sum);</code>
9	<code>    }</code>
10	<code>    public int methodB(int p, int n){</code>
11	<code>        int x = 0;</code>
12	<code>        y = y + (++p);</code>
13	<code>        x = x + 2 + n;</code>
14	<code>        sum = sum + x + y;</code>
15	<code>        System.out.println(x + " " + y+ " " + sum);</code>
16	<code>    return sum;</code>
17	<code>}</code>
18	<code>}</code>

**Driver code:**

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

	Outputs		
	x	y	Sum

# HOMEWORK

## Task 1

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

Driver Code	Output
<pre>public class CourseTester{     public static void main(String args[]){         CSECourse c1 = new CSECourse();         System.out.println("Course Name: "+c1.courseName);         System.out.println("Course Code: "+c1.courseCode);         System.out.println("Credit: "+c1.credit);     } }</pre>	<p>Course Name: Programming Language II Course Code: CSE111 Credit: 3</p>

## Task 2

Design the “**ImaginaryNumber**” class to generate the **output** given below:

Tester Class	Output
<pre>public class Tester8{     public static void main(String [] args){         ImaginaryNumber num1 = new ImaginaryNumber();         String p = num1.printNumber();         System.out.println(p);         System.out.println("1*****");         num1.realPart=3;         num1.imaginaryPart=7;         System.out.println(num1.printNumber());         System.out.println("2*****");         ImaginaryNumber num2 = new ImaginaryNumber();         num2.realPart=1;         num2.imaginaryPart=9;         System.out.println(num2.printNumber());     } }</pre>	<p>0 + 0i 1***** 3 + 7i 2***** 1 + 9i</p>

### Task 3

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

Driver Class	Output
<pre>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-----");         System.out.println(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-----");         System.out.println(as2.makeOptional());     } }</pre>	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 4

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
<pre>public class Tester10{     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();     } }</pre>	<pre>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: 24000.0 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: 31500.0 Tk 10 ====== Employee Name: Ron Weasley Employee Salary: 105000.0 Tk Employee Designation: manager</pre>

## Task 5

Consider the following class:

```
public class Student{  
    public String name;  
    public double cgpa;  
}
```

Show the output of the following sequence of statements:

Code	Output
Student s1 = new Student();	
Student s2 = new Student();	
Student s3 = null;	
s1.name = "Student One";	
s1.cgpa = 2.3;	
s3 = s1;	
s2.name = "Student Two";	
s2.cgpa = s3.cgpa + 1;	
s3.name = "New Student";	
System.out.println(s1.name);	
System.out.println(s2.name);	
System.out.println(s3.name);	
System.out.println(s1.cgpa);	
System.out.println(s2.cgpa);	
System.out.println(s3.cgpa);	
s3 = s2;	
s1.name = "old student";	
s2.name = "older student";	
s3.name = "oldest student";	
s2.cgpa = s1.cgpa - s3.cgpa + 4.5;	
System.out.println(s1.name);	
System.out.println(s2.name);	
System.out.println(s3.name);	
System.out.println(s1.cgpa);	
System.out.println(s2.cgpa);	
System.out.println(s3.cgpa);	

## Task 6

```
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 Tester5{  
    public static void main (String args[]){  
        Test1 t1 = new Test1();  
        t1.methodB(5,-8);  
        Test1 t2 = new Test1();  
        t2.methodA();  
    }  
}
```

### Outputs


## Ungraded Tasks (Optional)

(You don't have to submit the ungraded tasks)

### Task 1

Complete the “Cat” class so the main method produces the following output:

Test Class	Output
<pre>public class Test7{     public static void main(String [] args){         Cat c1 = new Cat();         System.out.println("=====");         c1.printCat();         c1.color = "Black";         System.out.println("=====");         c1.printCat();         c1.color = "Brown";         c1.action = "jumping";         System.out.println("=====");         c1.printCat();     } }</pre>	<pre>===== White cat is sitting ===== Black cat is sitting ===== Brown cat is jumping</pre>

### Task 2

Complete the **Bird** class so that main method produces the following **output**:

Test class	Output
<pre>public class Test8{     public static void main(String args[]) {         Bird b1 = new Bird();         b1.name = "Parrot";         b1.flyUp(3);         b1.makeNoise();         b1.flyDown(5);         b1.flyDown(2);         b1.flyDown(1);         Bird b2 = new Bird();         b2.name = "Eagle";         b2.flyUp(5);</pre>	<pre>Parrot has flown up 3 feet. Squawk Parrot cannot fly down 5 feet. Parrot has flown down 2 feet. Parrot has flown down 1 feet and landed. Eagle has flown up 5 feet. Eagle has flown down 5 feet and landed. Squeee</pre>

```

        b2.flyDown(5);
        b2.makeNoise();
    }
}

```

### Task 3

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
<pre> 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";     } } </pre>	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

```

        b2.customizeSpiceLevel("Naga");
        System.out.println("-----5-----");
        System.out.println(b2.serveBurger());
    }
}

```

## Task 4

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

Driver Class	Output
<pre> 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.details();         System.out.println("3-----");         m1.makeCall(9866);         System.out.println("4-----");         m1.addContact("Henry", 2365);         System.out.println("5-----");         m1.makeCall(7552);         m1.makeCall(2365);         System.out.println("6-----");         m1.addContact("Gomes", 4589);         m1.addContact("Antony", 8421);         m1.addContact("Tony", 5789);         System.out.println("7-----");         m1.details();     } } </pre>	<pre> Total Contacts: 0 Contact List: 1----- The contact of John is added. The contact of Maria is added. 2----- Total Contacts: 2 Contact List: John:9866 Maria:7865 3----- Calling John . . . 4----- The contact of Henry is added. 5----- Calling 7552 . . . Calling Henry . . . 6----- The contact of Gomes is added. The contact of Antony is added. Storage Full!! 7----- Total Contacts: 5 Contact List: John:9866 Maria:7865 Henry:2365 Gomes:4589 Antony:8421 </pre>

## Task 5

1	public class A {
2	public int temp = 4;
3	public int sum, y, x;
4	public void methodA(int m){
5	int [] n = {2,5};
6	int x = 0;
7	y = m + this.methodB(x++,m)+(temp++);
8	x = this.x + 2 + n[0];
9	sum = sum + x + y;
10	n[0] = sum + 2;
11	System.out.println(n[0]+" "+ x+ " " + sum);
12	}
13	public int methodB(int m, int n){
14	int y = 4 + this.y + m;
15	x = this.y + y + (++temp) - n;
16	sum = x + y + this.sum;
17	System.out.println(y+ " " + this.x + " " +sum);
18	return x;
19	}
20	}

```
public class Tester {
    public static void main(String [] args){
        A t1 = new A();
        t1.methodA(5);
        A t2 = new A();
        t2.methodB(12, 2);
    }
}
```

### Outputs