

# Lab Assignment 04



Inspiring Excellence

Course Code:	CSE111
Course Title:	Programming Language II
Topic:	Reference Passing & Multiclass Problem
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

Design the **Triangle** Class that will produce the following output. We will consider both triangles to have the same sides if all sides are equal in the same orientation/sequence only.

Types of Triangle:

- Equilateral: When all sides in the same orientation are equal.
- Isosceles: When any two sides of a triangle in the same orientation are equal.
- Scalene: When all sides are of different lengths.

Driver Code	Output
<pre>public class TriangleTester{     public static void main(String args[]){         Triangle t1 = new Triangle(4, 4, 4);         Triangle t2 = new Triangle(4, 5, 6);         Triangle t3 = new Triangle(4, 5, 6);         Triangle t4 = new Triangle(5, 4, 6);          t1.triangleDetails();         System.out.println("-----1-----");         System.out.println(t1.printTriangleType());         System.out.println("-----2-----");         t3.triangleDetails();         System.out.println(t3.printTriangleType());         System.out.println("-----3-----");         t4.triangleDetails();         System.out.println(t4.printTriangleType());         System.out.println("-----4-----");         t2.compareTriangles(t3);         System.out.println("-----5-----");         t1.compareTriangles(t2);         System.out.println("-----6-----");         t1 = t2;         t1.compareTriangles(t2);         System.out.println("-----7-----");         t3.compareTriangles(t4);     } }</pre>	<p>Three sides of the triangle are: 4, 4, 4 Perimeter: 12 -----1----- This is an Equilateral Triangle. -----2----- Three sides of the triangle are: 4, 5, 6 Perimeter: 15 This is a Scalene Triangle. -----3----- Three sides of the triangle are: 5, 4, 6 Perimeter: 15 This is a Scalene Triangle. -----4----- Addresses are different but the sides of the triangles are equal. -----5----- Addresses, length of the sides and perimeter all are different. -----6----- These two triangle objects have the same address. -----7----- Only the perimeter of both triangles is equal.</p>

## Task 2

Design the **Shop** class with necessary properties so that the following output is produced.

Driver Code	Output
<pre>public class ShopTester{     public static void main(String[] args){         Shop s1 = new Shop("Agora", 4);         Shop s2 = new Shop(3);         System.out.println("1==========");         s1.info();         System.out.println("2==========");         s2.info();         System.out.println("3==========");         s1.addItem("KitKat", 25);         s1.addItem("Hershey's", 100);         s1.addItem("Dairy Milk", 150);         s1.addItem("Feastables", 500);         s1.addItem("Prime", 1200);         System.out.println("4==========");         s1.info();         System.out.println("5==========");         s1.purchase("KitKat");         s1.purchase("Snickers");         System.out.println("6==========");         s2.info();         System.out.println("7==========");         s2.addItem(s1, "KitKat");         s2.addItem("Toblerone", 80);         s2.addItem(s1, "Mars");         System.out.println("8==========");         s2.info();     } }</pre>	<p>Agora shop created! Mega shop created! 1=====</p> <p>Shop Name: Agora Item Details: No items in shop 2=====</p> <p>Shop Name: Mega Item Details: No items in shop 3=====</p> <p>Prime, could not be added 4=====</p> <p>Shop Name: Agora Item Details: 4 / 4 KitKat - 25 Tk Hershey's - 100 Tk Dairy Milk - 150 Tk Feastables - 500 Tk 5=====</p> <p>Purchase Complete! Snickers, is not available in this shop 6=====</p> <p>Shop Name: Mega Item Details: No items in shop 7=====</p> <p>Mars, not found in Agora! 8=====</p> <p>Shop Name: Mega Item Details: 2 / 3 KitKat - 25 Tk Toblerone - 80 Tk</p>

## Task 3

Design the Team and Player class to get the output as shown.

Hints:

- Create an array in the Team class to store the Player's object
- Use constructor overloading technique for Team class
- A team can add maximum 11 players

Driver Code	Output
<pre>public class TeamTester {     public static void main(String[] args) {         Team b = new Team();         b.updateName("Bangladesh");         Player mashrafi = new Player("Mashrafi", 42, 100);         b.addPlayer(mashrafi);         Player tamim = new Player("Tamim", 35, 70);         b.addPlayer(tamim);         System.out.println("=====");         b.printDetail();         System.out.println("=====");         Team a = new Team("Australia");         Player ponting = new Player("Ponting", 50, 300);         a.addPlayer(ponting);         Player lee = new Player("Lee", 49, 200);         a.addPlayer(lee);         a.printDetail();     } }</pre>	<pre>===== Team: Bangladesh List of players: Name: Mashrafi Age: 42, Total Matches: 100 Name: Tamim Age: 35, Total Matches: 70 ===== Team: Australia List of players: Name: Ponting Age: 50, Total Matches: 300 Name: Lee Age: 49, Total Matches: 200</pre>

## Task 4

Write the **Teacher** and **Course** classes so that the TestTeacher class produces the outputs given.  
Hint: A teacher can add a maximum of 3 courses.

Driver Code	Output
<pre>public class TestTeacher{     public static void main(String [] args){         Teacher t1 = new Teacher("Matin Saad Abdullah", "MSA");         Teacher t2 = new Teacher("Mumit Khan", "MMK");         Teacher t3 = new Teacher("Sadia Hamid Kazi", "SKZ");         Course c1 = new Course("CSE 110");         Course c2 = new Course("CSE 111");         Course c3 = new Course("CSE 220");         Course c4 = new Course("CSE 221");         Course c5 = new Course("CSE 230");         Course c6 = new Course("CSE 310");         Course c7 = new Course("CSE 320");         Course c8 = new Course("CSE 340");         t1.addCourse(c1);         t1.addCourse(c2);         t2.addCourse(c3);         t2.addCourse(c4);         t2.addCourse(c5);         t3.addCourse(c6);         t3.addCourse(c7);         t3.addCourse(c8);         System.out.println("1=====");         t1.printDetail();         System.out.println("2=====");         t2.printDetail();         System.out.println("3=====");         t3.printDetail();     } }</pre>	<p>A new teacher has been created A new teacher has been created A new teacher has been created =====</p> <p>Name: Matin Saad Abdullah Initial: MSA List of courses: CSE 110 CSE 111 =====</p> <p>Name: Mumit Khan Initial: MMK List of courses: CSE 220 CSE 221 CSE 230 =====</p> <p>Name: Sadia Hamid Kazi Initial: SKZ List of courses: CSE 310 CSE 320 CSE 340</p>

## Task 5

1	public class Trace{
2	public int x, y = 5, temp = -4, sum = 3;
3	public Trace(){
4	y = temp + 3 ;
5	sum = 3 + temp + 2;
6	temp -= 2;
7	}
8	public Trace(Trace t){
9	sum = t.sum++;
10	x = t.x + 2;
11	t.methodB(2,3);
12	}
13	public void methodA(int m, int n){
14	int x = 2;
15	y = y + m + (temp++);
16	x = x + 5 + n;
17	sum = sum + x + y;
18	System.out.println(x + " " + y+ " " + sum);
19	}
20	public void methodB(int m, int n){
21	int y = 0;
22	y = y + this.y;
23	x = this.y + 2 + temp;
24	methodA(x, y);
25	sum = x + y + sum;
26	System.out.println(x + " " + y+ " " + sum);
27	}
28	}

```
public class Tester5 {  
    public static void main(String args []){  
        Trace t1 = new Trace();  
        Trace t2 = new Trace(t1);  
        t1.methodA(1, 2);  
        t2.methodB(3, 2);  
    }  
}
```

### Outputs


## Task 6

```
1 public class TracingX {  
2     public int x, y = 1;  
3     public int metA(int y){  
4         y += x + 3;  
5         int temp = y + this.y;  
6         if (temp % 2 == 0){  
7             return temp;  
8         }  
9         TracingX t = new TracingX();  
10        t.y = this.x - (++x) + t.x;  
11        this.y = y + t.metA(t.x);  
12        System.out.println(x +" "+ y +" "+temp);  
13        return temp+this.y;  
14    }  
15 }
```

Driver code:

```
public class TesterX {  
    public static void main(String[] args) {  
        TracingX t1 = new TracingX();  
        t1.y = t1.x = 5;  
        TracingX t2 = new TracingX();  
        t2.x = t1.metA(2);  
        t2.y = t2.metA(4);  
        System.out.println(t1.y +t1.x +" "+t2.x +" "+t2.y);  
    }  
}
```

Output:


# HOMEWORK

## Task 1

Design the **Player** class to obtain the following output from the driver code. You can assume, each player can defeat at most 5 villains.

Driver Code	Expected Output
<pre>public class PlayerTester {     public static void main(String[] args) {         Player ben = new Player("Ben", 10);         System.out.println("=====1=====");         ben.viewInfo();         System.out.println("=====2=====");         ben.defeatVillain("Vilgax", 100);         System.out.println("=====3=====");         ben.defeatVillain("Yamcha", 10);         System.out.println("=====4=====");         ben.viewInfo();         System.out.println("=====5=====");         ben.defeatVillain('8', "Vilgax", 100);         System.out.println("=====6=====");         ben.viewInfo();         System.out.println("=====7=====");         Player kevin = new Player("Kevin");         System.out.println("=====8=====");         kevin.viewInfo();         System.out.println("=====9=====");         ben.defeatVillain(kevin);         System.out.println("=====10=====");         ben.viewInfo();         System.out.println("=====11=====");         Player goku = new Player("Goku", 9000);         System.out.println("=====12=====");         ben.defeatVillain(goku);     } }</pre>	<pre>Ben joined the game HP: 10 =====1===== Player Name: Ben Current HP: 10 =====2===== failed to defeat Vilgax =====3===== defeated Yamcha =====4===== Player Name: Ben Current HP: 20 Defeated: Yamcha, =====5===== HP with 8x boost: 160 defeated Vilgax =====6===== Player Name: Ben Current HP: 260 Defeated: Yamcha, Vilgax, =====7===== Kevin joined the game HP: 100 =====8===== Player Name: Kevin Current HP: 100 =====9===== defeated Kevin =====10===== Player Name: Ben Current HP: 360 Defeated: Yamcha, Vilgax, Kevin, =====11===== Goku joined the game HP: 9000 =====12===== failed to defeat Goku</pre>

## Task 2

Write “**Student**” class to show the following expected outputs

**Note:**

- ❖ A student can't take any course until the CGPA is set.
- ❖ A student cannot take more than 4 courses.
- ❖ A student with CGPA below 3 cannot take more than 3 courses.

Driver Code	Expected Output
<pre>public class StudentDriver {     public static void main(String[] args){         Student student1 = new Student(12345678);         System.out.println("1-----");         student1.addCourse("CSE110");         System.out.println("2-----");         student1.storeCG(2.5);         student1.addCourse("CSE110");         student1.addCourse("ENG101");         student1.showAdvisee();         System.out.println("3-----");         student1.removeAllCourse();         student1.showAdvisee();         System.out.println("4-----");         student1.storeID(54652365);         String[] courses = {"SOC101", "CSE111", "ENG102"};         student1.addCourse(courses);         student1.showAdvisee();         System.out.println("5-----");         student1.addCourse("CSE230");         student1.showAdvisee();         System.out.println("6-----");         Student student2 = new Student(975738383, 3.7);         System.out.println("7-----");         String[] courses2 =         {"CSE220", "PHY112", "MAT120", "BUS101", "CHN101"};         student2.addCourse(courses2);         student2.showAdvisee();     } }</pre>	<p>A student with ID 12345678 has been created. 1----- Failed to add CSE110 Set CG first 2----- Student ID: 12345678, CGPA: 2.5 Added courses are: CSE110 ENG101 3----- Student ID: 12345678, CGPA: 2.5 No courses added. 4----- Student ID: 54652365, CGPA: 2.5 Added courses are: SOC101 CSE111 ENG102 5----- Failed to add CSE230 CG is low. Can't add more than 3 courses. Student ID: 54652365, CGPA: 2.5 Added courses are: SOC101 CSE111 ENG102 6----- A student with ID 975738383 and cgpa 3.7 has been created. 7----- Failed to add CHN101 Maximum 4 courses allowed. Student ID: 975738383, CGPA: 3.7 Added courses are: CSE220 PHY112 MAT120 BUS101</p>

## Task 3

Design the **Student** and the **Connect** class so that the following output is produced.

Note:

- A student's email, password, and login status are null by default while creating an object of the **Student** class.
- Your code should satisfy the conditions mentioned in the output only.
- **Connect** class will have two instance variables: `totalAdvisee` and an array of **Student** type to store the student object. The array will be updated inside the `advising()` method only when the advising is successful.
- **Connect** can take at most 5 advisees.

Driver Code	Expected Output
<pre> public class ConnectTester {     public static void main(String[] args) {         Student rakib = new Student("Rakib", 12301455, "CSE");         Student roy = new Student("Roy", 12501345, "CS");         System.out.println("1*****");         Connect connectObj = new Connect();         System.out.println("2*****");         connectObj.login(rakib);         System.out.println("3*****");         connectObj.advising(rakib);         System.out.println("4*****");         rakib.email = "rakib@hotmail.com";         rakib.password = "1234";         System.out.println("5*****");         connectObj.login(rakib);         System.out.println("6*****");         connectObj.advising(rakib);         System.out.println("7*****");         connectObj.advising(rakib, "CSE110", "PHY111",                             "MAT110", "CSE260");         System.out.println("8*****");         connectObj.advising(rakib, "CSE110",                             "PHY111", "MAT110");         System.out.println("9*****");         connectObj.allAdviseeInfo();         System.out.println("10*****");         roy.email = "roy@hotmail.com";         roy.password = "abcd";         connectObj.login(roy);         System.out.println("11*****");         connectObj.advising(roy, "CSE110", "ENG101", "PHY112");         System.out.println("12*****");         connectObj.allAdviseeInfo();     } } </pre>	<p>Student object is created      Student object is created      1*****      Connect is ready to use!      2*****      Email and password need to be set.      3*****      Please login to advise courses!      4*****      5*****      Login successful      6*****      You haven't selected any courses.      7*****      You need special approval to take more than 3 courses.      8*****      Advising successful!      9*****      Total Advisee: 1      Name: Rakib ID: 12301455      Department: CSE      Advised Courses:      CSE110 PHY111 MAT110      ======      10*****      Login successful      11*****      Advising successful!      12*****      Total Advisee: 2      Name: Rakib ID: 12301455      Department: CSE      Advised Courses:      CSE110 PHY111 MAT110      ======      Name: Roy ID: 12501345      Department: CS      Advised Courses:      CSE110 ENG101 PHY112      ====== </p>

## Task 4

Design the **TravelPrep** class so that the given output is generated.

- This class keeps track of the package info and the destination that goes into it.
- The initial budget will be 1250 yen for any traveller.
- A traveller can visit at most 3 destinations if that falls under their budget.

Driver Code	Expected Output
<pre>public class TravelPlan {     public static void main(String[] args) {         TravelPrep t1 = new TravelPrep();         System.out.println("1=====");         TravelPrep t2 = new TravelPrep("Package-2", 2100);         System.out.println("2=====");         TravelPrep d1 = new TravelPrep("Fushimi", "Shrine");         TravelPrep d2 = new TravelPrep("Lake", "Kawaguchi", 550);         TravelPrep d3 = new TravelPrep("Shrine", "Hieizan", 1000);         TravelPrep d4 = new TravelPrep("Lake", "Ashi", 620);         System.out.println("3=====");         t2.add_to_itinerary(d1);         t2.add_to_itinerary(d2, d3);         t2.show_itinerary();         System.out.println("4=====");         System.out.println(d2.updateCost(60));         System.out.println("5=====");         t1.t_name = "Package-1";         t1.add_to_itinerary(d2, d4);         t1.add_to_itinerary(d3);         System.out.println("6=====");         t1.show_itinerary();     } }</pre>	<p>Unknown package would cost at most 1250 yen 1=====</p> <p>Package-2 would cost at most 2100 yen 2=====</p> <p>Fushimi Shrine costs 300 yen Kawaguchi Lake costs 550 yen Hieizan Shrine costs 1000 yen Ashi Lake costs 620 yen 3=====</p> <p>Itinerary for Package-2 1. Fushimi Shrine - 300 yen 2. Kawaguchi Lake - 550 yen 3. Hieizan Shrine - 1000 yen Total cost of 3 destinations: 1850 yen 4=====</p> <p>Cost of Kawaguchi Lake updated to 610 5=====</p> <p>Budget going overboard 6=====</p> <p>Itinerary for Package-1 1. Kawaguchi Lake - 610 yen 2. Ashi Lake - 620 yen Total cost of 2 destinations: 1230 yen</p>

## Task 5

1	public class TestX{
2	public int result = 17, i, n = 7, m = 5;
3	public int[] num={6,8,11};
4	public void compute(int[] nums){
5	n = nums[0] + this.m;
6	this.result += nums[i];
7	TestX temp = new TestX();
8	boolean check = !(temp.modify(temp, num));
9	if (check) {
10	m = nums[1] + temp.result;
11	} else {
12	m = result - nums[3];
13	}
14	System.out.println(m + " " + n + " " + result);
15	}
16	public boolean modify(TestX obj, int[] arr){
17	arr[1]++;
18	i=i+1;
19	obj.m = this.result + arr[i];
20	obj.result = obj.m - obj.n + n;
21	System.out.println(obj.m + " " + obj.n + " " + result);
22	return arr[1] % 2 == 0;
23	}
24	public boolean modify(TestX obj){
25	num[i]++;
26	i=i+1;
27	obj.m = this.result + num[i];
28	obj.result = num[2] - obj.n + n;
29	System.out.println(obj.m + " " + obj.n + " " + result);
30	return num[1] % 2 != 0;
31	}
32	}

```
public class tester11{
    public static void main(String[] args){
        TestX tx = new TestX();
        int[] data = {4, 6, 2, 3, 5};
        tx.compute(data);
        tx.modify(tx);
    }
}
```

**Outputs**


## Task 6

1	public class A {
2	public int x, y = 2, sum = 5;
3	public A(){
4	x = sum - y;
5	this.methodB(1);
6	}
7	public A(int x){
8	this.x = x;
9	}
10	public void methodB(int y) {
11	int temp = (y++) + (++this.y);
12	sum = y + temp - ++this.x;
13	System.out.println(y + " " + temp + " " + (sum++));
14	this.y = (++temp) + methodC(temp, y) + x;
15	}
16	public void methodA(int temp, int x) {
17	A a2 = new A(6);
18	a2.sum = this.methodC(a2);
19	System.out.println(this.x + " " + a2.sum + " " + y);
20	this.sum = a2.methodC(this);
21	System.out.println(a2.x + " " + this.sum + " " + a2.y);
22	}
23	public int methodC(A a) {
24	return this.x + a.y + a.sum;
25	}
26	public int methodC(int sum, int y) {
27	y = (this.x++) + sum + 3;
28	return y;
29	}
30	}

```
public class Tester12{
    public static void main(String [] args) {
        A a1 = new A();
        a1.methodA(4, 3);
    }
}
```

Outputs


## Ungraded Tasks (Optional)

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

### Task 1

Driver Code	Output
<pre>public class ConnectTester{     public static void main(String[] args) {         ConnectFriends sanaf = new ConnectFriends("Sanaf");         System.out.println("=====1=====");         ConnectFriends mantasha = new ConnectFriends("Mantasha", 3);         ConnectFriends mostafiz = new ConnectFriends("Mostafiz");         ConnectFriends matt = new ConnectFriends("Matt", 4);         System.out.println("=====2=====");         sanaf.sendFriendRequest(mantasha);         System.out.println("=====3=====");         sanaf.sendFriendRequest(mostafiz, matt);         System.out.println("=====4=====");         sanaf.showDetails();         System.out.println("=====5=====");         sanaf.removeRequest("Mantasha");         System.out.println("=====6=====");         sanaf.showDetails();         System.out.println("=====7=====");         sanaf.removeRequest("Murdock");         System.out.println("=====8=====");         sanaf.removeRequest("Matt");         sanaf.removeRequest("Mostafiz");         sanaf.showDetails();         System.out.println("=====9=====");         mantasha.showDetails();     } }</pre>	<pre>Welcome to ConnectFriends, Sanaf =====1=====  Welcome to ConnectFriends, Mantasha Welcome to ConnectFriends, Mostafiz Welcome to ConnectFriends, Matt =====2=====  Sanaf sent a friend request to Mantasha. =====3=====  Sanaf sent a friend request to Mostafiz. Sanaf has reached the friend request limit! =====4=====  User Name: Sanaf Maximum number of Sent Friend Request: 2 Total Friends Request: 2 Sent Friends Request: Mantasha Mostafiz =====5=====  Reuest to add Mantasha is removed for Sanaf. =====6=====  User Name: Sanaf Maximum number of Sent Friend Request: 2 Total Friends Request: 1 Sent Friends Request: Mostafiz =====7=====  Murdock is not in Sanaf's sent request list. =====8=====  Matt is not in Sanaf's sent request list. Reuest to add Mostafiz is removed for Sanaf. User Name: Sanaf Maximum number of Sent Friend Request: 2 Total Friends Request: 0 Sent Friends Request: =====9=====  User Name: Mantasha Maximum number of Sent Friend Request: 3 Total Friends Request: 0 Sent Friends Request:</pre>

## Task 2

```
1 public class QuizA {  
2     public int x, y;  
3     public int sum = 1;  
4     public QuizA(int x, int y) {  
5         this.x = y;  
6         this.y = x;  
7     }  
8     public void methodA() {  
9         int x = 3;  
10        y = this.y + x;  
11        QuizA exam = new QuizA(5, 11);  
12        exam.sum = x;  
13        exam.y = this.y;  
14        x = this.x + x + exam.sum;  
15        this.y = this.sum + methodB(exam.sum, exam);  
16        System.out.println(exam.x + " " + this.y + " " + sum);  
17        sum = x % 2 + this.x;  
18        y = x + y + exam.sum;  
19        System.out.println(x + " " + y + " " + sum);  
20    }  
21    public int methodB(int x1, QuizA x2) {  
22        int y = 0;  
23        y = this.y + x2.sum;  
24        x2.sum = x1 + x2.x;  
25        sum = sum + x + y;  
26        System.out.println(this.x + " " + this.y + " " + sum);  
27        return x2.sum;  
28    }  
29 }
```

Driver Code	Output		
public class QuizTesterA{ public static void main(String []args){ QuizA q1 = new QuizA(3,4); q1.methodA(); } }			

### Task 3

1	public class msgClass{
2	public int content;
3	}
4	class FinalT5A{
5	public int sum = 2, y = 1, x = 1;
6	public void methodA(){
7	int x=6, y =0;
8	msgClass myMsg = new msgClass();
9	myMsg.content = this.x;
10	x = x + myMsg.content;
11	this.y = this.y + methodB(myMsg, myMsg.content);
12	System.out.println(x + " " + this.y+ " " + sum);
13	y = this.y/2 + this.x;
14	x = y + sum/2;
15	sum = x + y + myMsg.content;
16	System.out.println(x + " " + y+ " " + sum);
17	}
18	public int methodB(msgClass mg2, int mg1){
19	int x = 0;
20	y = y + mg2.content;
21	mg2.content = y + mg1;
22	x = this.x + 3 + mg1;
23	sum = sum + x + y;
24	System.out.println(this.x + " " + this.y+ " " + sum);
25	mg2.content = sum - mg1 ;
26	return sum;
27	}
28	}

DRIVER CODE	OUTPUTS		
public class Tester10{ public static void main(String args []){ FinalT5A ft5A = new FinalT5A(); ft5A.methodA(); } }			