

Lab Assignment 08



Inspiring Excellence

Course Code:	CSE111
Course Title:	Programming Language II
Topic:	Static Variable & Static Method
Number of Tasks:	11

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

Task 1

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

- Passenger class has two static variables *no_of_passenger* and *total_fare*.
- Each passenger has to pay 20 TK/Distance and extra 10 TK/BaggageWeight.

Given Code	Expected Output
<pre>public class PassengerTester{ public static void main(String args[]){ System.out.println("Total Passenger: "+ Passenger.no_of_passenger); System.out.println("Total Fare: "+ Passenger.total_fare + " TK"); System.out.println("=====1====="); Passenger p1 = new Passenger("Lara", 5.6); p1.passengerDetails(); System.out.println("=====2====="); Passenger p2 = new Passenger("Kevin", 10.0); p2.setBaggageWeight(6.8); p2.passengerDetails(); System.out.println("=====3====="); Passenger p3 = new Passenger("Robin", 2.3); p3.setBaggageWeight(5); p3.passengerDetails(); System.out.println("=====4====="); System.out.println("Total Passenger: "+ Passenger.no_of_passenger); System.out.println("Total Fare: "+ Passenger.total_fare + " TK"); } }</pre>	<pre>Total Passenger: 0 Total Fare: 0.0 TK =====1===== Name: Lara Fare: 112.0 TK =====2===== Name: Kevin Fare: 268.0 TK =====3===== Name: Robin Fare: 96.0 TK =====4===== Total Passenger: 3 Total Fare: 476.0 TK</pre>

Task 2

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

- The Book class has two static variables: total_books_sold and total_revenue.
- Each book has a base price of 150 TK. If the discountPercentage is applied, the book's price is reduced by that percentage.
- The Book class should have a method to calculate the price after the discount

Given Code	Expected Output
<pre>public class BookTester { public static void main(String[] args) { System.out.println("Total Books Sold: " + Book.total_books_sold); System.out.println("Total Revenue: "+Book.total_revenue + " TK"); System.out.println("=====1====="); Book b1 = new Book("Java Programming", 10); // 10% discount b1.bookDetails(); System.out.println("=====2====="); Book b2 = new Book("Python Programming", 15); // 15% discount b2.bookDetails(); System.out.println("=====3====="); Book b3 = new Book("Data Structures", 5); // 5% discount b3.bookDetails(); System.out.println("=====4====="); System.out.println("Total Books Sold: " + Book.total_books_sold); System.out.println("Total Revenue: "+Book.total_revenue + " TK"); } }</pre>	<pre>Total Books Sold: 0 Total Revenue: 0.0 TK =====1===== Title: Java Programming Price after Discount: 135.0 TK =====2===== Title: Python Programming Price after Discount: 127.5 TK =====3===== Title: Data Structures Price after Discount: 142.5 TK =====4===== Total Books Sold: 3 Total Revenue: 405.0 TK</pre>

Task 3

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

Driver Code	Output
<pre>public class StudentTester { public static void main(String[] args) { Student.printDetails(); System.out.println("-----"); Student mikasa = new Student("Mikasa", 3.75); mikasa.individualDetail(); System.out.println("-----"); Student.printDetails(); System.out.println("-----"); Student harry = Student.createStudent("Harry", 2.5, "Charms"); harry.individualDetail(); System.out.println("-----"); Student.printDetails(); System.out.println("-----"); Student levi = new Student("Levi", 3.33); levi.individualDetail(); System.out.println("-----"); Student.printDetails(); } }</pre>	<pre>Total Student(s): 0 CSE Student(s): 0 Other Department Student(s): 0 ----- ID: 1 Name: Mikasa CGPA: 3.75 Department: CSE ----- Total Student(s): 1 CSE Student(s): 1 Other Department Student(s): 0 ----- ID: 2 Name: Harry CGPA: 2.5 Department: Charms ----- Total Student(s): 2 CSE Student(s): 1 Other Department Student(s): 1 ----- ID: 3 Name: Levi CGPA: 3.33 Department: CSE ----- Total Student(s): 3 CSE Student(s): 2 Other Department Student(s): 1</pre>

Task 4

Suppose you have opened a new library, from where your friends can borrow books. Initially you have bought 3 books (Pather Panchali, Durgesh Nandini & Anandmath) each of 3 copies only. Design the **Borrower** class in such a way that the following code provides the expected output.

- You are given the arrays **book_count** and **book_name** to keep track of the number of books available. For simplicity, assume that there will be no other books in the library.
- You must reuse the **remainingBooks()** method when needed.

Given Code	Expected Output
<pre>public class Tester{ public static void main(String args[]){ Borrower.bookStatus(); System.out.println("*****1*****"); Borrower b1 = new Borrower("Nabila"); b1.borrowBook("Pather Panchali"); b1.borrowBook("Anandmath"); b1.borrowerDetails(); System.out.println("*****2*****"); Borrower b2 = new Borrower("Sadia"); b2.borrowBook("Anandmath"); b2.borrowBook("Durgesh Nandini"); b2.borrowBook("Pather Panchali"); b2.borrowerDetails(); System.out.println("*****3*****"); System.out.println(Borrower.remainingBooks("Anandmath")+ "copies of Anandmath is remaining."); System.out.println("*****4*****"); Borrower b3 = new Borrower("Anika"); b3.borrowBook("Anandmath"); Borrower.bookStatus(); System.out.println("*****5*****"); Borrower b4 = new Borrower("Oishi"); b4.borrowBook("Anandmath"); b4.borrowBook("Durgesh Nandini"); b4.borrowerDetails(); } } public class Borrower{ public static int book_count[] = {3, 3, 3}; public static String book_name[] = {"Pather Panchali", "Durgesh Nandini", "Anandmath"}; // Your Code here }</pre>	<pre>Available Books: Pather Panchali: 3 Durgesh Nandini: 3 Anandmath: 3 *****1***** Name: Nabila Books Borrowed: Pather Panchali Anandmath *****2***** Name: Sadia Books Borrowed: Anandmath Durgesh Nandini Pather Panchali *****3***** 1 copies of Anandmath is remaining. *****4***** Available Books: Pather Panchali: 1 Durgesh Nandini: 2 Anandmath: 0 *****5***** This book is not available. Name: Oishi Books Borrowed: Durgesh Nandini</pre>

Task 5

For this task, you need to design the **Cargo** class with appropriate static and non-static variables and methods to produce this given output for the given tester code.

Note: .load() method marks an object as selected for transport, and .unload() method unmarked it. At a time, the transport capacity is 10.0 Tonnes. Each Cargo object is initialized with 2 attributes from the constructor - the contents and the weight. Carefully observe the outputs to identify the other attributes and design the class.

Given Code	Expected Output
<pre> public class CargoTester { public static void main(String[] args) { System.out.println("Cargo Capacity: " + Cargo.capacity()); System.out.println("1====="); Cargo a = new Cargo("Industrial Machinery", 4.5); a.details(); System.out.println("2====="); a.load(); System.out.println("3====="); Cargo b = new Cargo("Steel Ingot", 2.7); b.details(); System.out.println("4====="); System.out.println("Cargo Capacity: " + Cargo.capacity()); System.out.println("5====="); b.load(); System.out.println("Cargo Capacity: " + Cargo.capacity()); System.out.println("6====="); Cargo c = new Cargo("Tree Trunks", 3.6); c.load(); System.out.println("7====="); c.details(); b.details(); System.out.println("8====="); Cargo d = new Cargo("Processed Goods", 1.8); d.load(); System.out.println("Cargo Capacity: " + Cargo.capacity()); System.out.println("9====="); b.unload(); System.out.println("Cargo Capacity: " + Cargo.capacity()); System.out.println("10====="); c.load(); System.out.println("11====="); b.details(); System.out.println("Cargo Capacity: " + Cargo.capacity()); } } </pre>	<pre> Cargo Capacity: 10.0 1===== Cargo ID: 1, Contents: Industrial Machinery, Weight: 4.5, Loaded: false 2===== Cargo 1 loaded for transport. 3===== Cargo ID: 2, Contents: Steel Ingot, Weight: 2.7, Loaded: false 4===== Cargo Capacity: 5.5 5===== Cargo 2 loaded for transport. Cargo Capacity: 2.8 6===== Cannot load cargo, exceeds weight capacity. 7===== Cargo ID: 3, Contents: Tree Trunks, Weight: 3.6, Loaded: false Cargo ID: 2, Contents: Steel Ingot, Weight: 2.7, Loaded: true 8===== Cargo 4 loaded for transport. Cargo Capacity: 1.0 9===== Cargo 2 unloaded. Cargo Capacity: 3.7 10===== Cargo 3 loaded for transport. 11===== Cargo ID: 2, Contents: Steel Ingot, Weight: 2.7, Loaded: false Cargo Capacity: 0.09999999999999964 </pre>

Task 6

Complete the class **Circle** so that the desired outputs are generated properly.

Given Code	Expected Output
<pre>public class shapeTester { public static void main(String[] args) { Circle c = new Circle(); System.out.println("====="); c.name = "Circle"; c.color = "Red"; c.radius = 5; c.displayInfo(); System.out.println("====="); c.area(); } } public class Shape { public String name; public String color; public void displayInfo() { System.out.printf("Name: %s\nColor: %s\n", name, color); } } public class Circle extends Shape { //Your Code Here }</pre>	<pre>===== Name: Circle Color: Red ===== Area of Red Circle: 78.54</pre>

Task 7

Complete the class **Dog** so that the desired outputs are generated properly.

Given Code	Expected Output
<pre>public class AnimalTester{ public static void main(String args[]){ Animal a1 = new Animal(); System.out.println("1-----"); a1.details(); System.out.println("2-----"); Dog d1 = new Dog(); d1.name = "Pammy"; System.out.println("3-----"); System.out.println("Name: " + d1.getName()); d1.details(); System.out.println("4-----"); d1.updateSound("Bark"); System.out.println("5-----"); d1.details(); } } public class Animal{ public int legs = 4; public String sound = "Not defined"; public void details(){ System.out.println("Legs: "+legs); System.out.println("Sound: "+sound); } } public class Dog extends Animal{ //Your Code Here }</pre>	<pre>1----- Legs: 4 Sound: Not defined 2----- The dog says hello! 3----- Name: Pammy Legs: 4 Sound: Not defined 4----- 5----- Legs: 4 Sound: Bark</pre>

Task 8

1.	public class Maze{	Output	
2.	public static int x;		
3.	public void methodA(){		
4.	int m = 5;		
5.	x=11;		
6.	System.out.println(x+" "+m);		
7.	m=methodB(m-3)+x;		
8.	System.out.println(x+" "+(m));		
9.	methodB(x,m);		
10.	System.out.println(x+" "+m+x);		
11.	}		
12.	public int methodB(int y){		
13.	x=y*y;		
14.	System.out.println(x+" "+y);		
15.	return x+3;		
16.	}		
17.	public void methodB(int z, int x){		
18.	z=z-2;		
19.	x=x*1%z;		
20.	System.out.println(z+" "+x);		
21.	}		
22.	}		
23.	public class Test8{		
24.	public static void main(String [] args){		
25.	Maze c = new Maze();		
26.	c.methodA();		
27.	c.methodB(-11, 45);		
28.	}		
29.	}		

Task 9

		Output		
1.	public class Tracing {			
2.	public static int x= 0, y = 0;			
3.	public int a, b;			
4.	public Tracing(int a, int b){			
5.	this.a = a;			
6.	this.b = b;			
7.	x+=1;			
8.	y+=2;			
9.	}			
10.	public void methodA(int a){			
11.	this.a = x+a;			
12.	this.b = this.b+ this.a +this.methodB();			
13.	System.out.println(this.a+" "+this.b+" "+x);			
14.	}			
15.	public int methodB(){			
16.	this.b = y - this.b + this.a;			
17.	System.out.println(this.a+" "+this.b+" "+x);			
18.	x += this.b;			
19.	return this.b;			
20.	}			
21.	public void methodB(Tracing t1){			
22.	t1.b = this.y - t1.b + this.b;			
23.	System.out.println(t1.a+" "+t1.b+" "+x);			
24.	}			
25.	}			
26.	public class Test9{			
27.	public static void main(String [] args){			
28.	Tracing t1= new Tracing(2, 3);			
29.	t1.methodA(1);			
30.	Tracing t2= new Tracing(3, 4);			
31.	t2.methodA(2);			
32.	t1.methodB(t2);			
33.	t2.methodB(t2);			
34.	}			
35.	}			

Task 10

1	public class FinalT6A{	Outputs		
2	public static int temp = 3;			
3	public int sum;			
4	public int y = 2;			
5	public FinalT6A(int x, int p){			
6	temp+=3;			
7	y = temp - p;			
8	sum = temp + x;			
9	System.out.println(x + " " + y+ " " + sum);			
10	}			
11	public void methodA(){			
12	int x=0, y =0;			
13	y = y + this.y;			
14	x = this.y + 2 + temp;			
15	sum = x + y + methodB(temp, y);			
16	System.out.println(x + " " + y+ " " + sum);			
17	}			
18	public int methodB(int temp, int n){			
19	int x = 0;			
20	y = y + (++temp);			
21	x = x + 2 + n;			
22	sum = sum + x + y;			
23	System.out.println(x + " " + y+ " " + sum);			
24	return sum;			
25	}			
26	}			
27	public class Test10{			
28	public static void main(String [] args){			
29	FinalT6A q1 = new FinalT6A(2,1);			
30	q1.methodA();			
31	q1.methodA();			
32	}			
33	}			

Task 11

Find the outputs after running the main() method in **Test11** class.

1	public class Quiz1{	Outputs		
2	public static int temp = 4;			
3	public int sum;			
4	public int y;			
5	public Quiz1(){			
6	y = temp - 1;			
7	sum = temp + 1;			
8	temp+=2;			
9	}			
10	public Quiz1(int p){			
11	y = temp + p ;			
12	sum = p + temp + 1;			
13	temp-=1;			
14	}			
15	public void methodA(){			
16	int x=0, y =0;			
17	y = y + this.y;			
18	x = this.y + 2 + temp;			
19	sum = x + y + methodB(x, y);			
20	System.out.println(x + " " + y+ " " + sum);			
21	}			
22	public int methodB(int m, int n){			
23	int x = 0;			
24	y = y + m + (++temp);			
25	x = x + 2 + n;			
26	sum = sum + x + y;			
27	System.out.println(x + " " + y+ " " + sum);			
28	return sum;			
29	}			
30	}			
31	public class Test11{			
32	public static void main(String [] args){			
33	Quiz1 q1 = new Quiz1();			
34	q1.methodA();			
35	q1.methodA();			
36	Quiz1.temp+= 2;			

37	Quiz1 q2 = new Quiz1(2);	
38	q2.methodA();	
39	q2.methodA();	
40	}	
41	}	