BIRBHUM INSTITUTE OF ENGINEERING & TECHNOLOGY

Name : Dipak Majumdar

Department: Computer Science & Engineering

Class Roll No : CSE-D/06-21

Subject Name: Object Oriented Programming

Subject Code: PCC-CS503

University Roll No' : 11800121031

Year : 3rd year Semester: 5th Semester

Group B (Report)

5. Consider a Box Class. Find the volume of the Box when it is not a cube and it is a cube using Method Overloading.

```
class Box {
   private double width;
   private double height;
   private double depth;

   public Box(int i, int i1, int i2) {
      width = height = depth = 0.0;
   }

   public int box(double w, double h, double d) {
      width = w;
      height = h;
      depth = d;
      return 0;
   }

   public void volume(){
      System.out.println("Volume =" + (width*height*depth));
   }
}

public class BoxDemo {
   public static void main(String[] args) {
      Box myBox1 = new Box(10, 15, 20);
      Box myBox2 = new Box(10,15,20);
      myBox1.volume();
      myBox2.volume();
   }
}
/*
Output
0
*/
```

Group A

[Assignment-2 (CA2)]

1. What is Constructor? What are the advantages of Constructor? Explain with example.

Answer: The constructor is called during object creation for instantiation of the variables. The main advantage of a constructor is that it can provide a fully initialised usable object upon object creation.

When we pass the values of the instance variables inside a constructor, during object creation itself the variables gets initialised and hence we get a usable object immediately.

Through constructor chaining, the instance variables of the super class can be initialised along with the sub class.

```
class MyMainEmployee{
   private int id;
   private String name;
   public MyMainEmployee(){
      id = 45;
      name = "Rahul";
   }

   public String getName() { return name; }
   public void setName(String n) { this.name = n;}
   public void setId(int i) { this.id = i; }
   public int getId() { return id; }
}
public class constructor {
   public static void main(String[] args) {
      MyMainEmployee Rahul = new MyMainEmployee();
      Rahul.setName("Shubham");
      Rahul.setId(34);
      System.out.println(Rahul.getId());
      System.out.println(Rahul.getName());


   }
}
/* output:
34
Shubham
*/
```

2. **Explain Recursion in Java with an example**.

Answer: Recursion in java is a process in which a method calls itself continuously. A method in java that calls itself is called recursive method.

It makes the code compact but complex to understand.

```java
public class Recursion_Example {
   static int count=0;
   static void p(){
      count++;
      if(count<=2){
         System.out.println("hello "+count);
         p();
      }
   }

   public static void main(String[] args) {
      p();
   }
}

/* Output
hello 1
hello 2
*/
```

3. What is Method Overloading? What are the basic differences between Method Overloading and Constructor Overloading?
   Explain with example.

   Answer: If a class has multiple methods having same name but different in parameters, it is known as **Method Overloading**.

➢ Constructor Overloading

2. Writing more than 1 constructor in a class with a unique set of arguments is called as Constructor Overloading
3. All constructors will have the name of the class
4. Overloaded constructor will be executed at the time of instantiating an object
5. An overloaded constructor cannot be static as a constructor relates to the creation of an object

➢ Method Overloading

1. Writing more than one method within a class with a unique set of arguments is called as method overloading
2. All methods must share the same name
3. An overloaded method if not static can only be called after instantiating the object as per the requirement
4. Overloaded method can be static, and it can be a

```
class Adder{
   static int add(int a, int b){ return a+b; }
   static int add(int a, int b, int c){ return a+b+c; }
}
public class TestOverLoading {
   public static void main(String[] args) {
      System.out.println(Adder.add(11,11));
      System.out.println(Adder.add(11,11,11));
   }
}
/* Output
22
33
*/
```

5. **How a Method can return an Object in java? Explain with proper example.**
   Answer: In java, a method can return any type of data, including objects. For example, in the following program, the incrByTen () method returns an object in which the value of an (an integer variable) is ten greater than it is in the invoking object.

```
class Test
{
   int a;

   Test(int i)
   {
      a = i;
   }

   Test incrByTen()
   {
      Test temp = new Test(a+10);
      return temp;
   }
}
public class javaProgram {
   public static void main(String args[])
   {

      Test obj1 = new Test(2);
      Test obj2;

      obj2 = obj1.incrByTen();

      System.out.println("obj1.a : " + obj1.a);
      System.out.println("obj2.a : " + obj2.a);

      obj2 = obj2.incrByTen();

      System.out.println("obj2.a after second increase : " + obj2.a);

   }
}
/* Output
Obj1.a : 2
Obj2.a : 12
Obj2.a after second increase:22
```