Minimum is: 2

EXPERIMENT-3

OBJECTIVE

Write Object Oriented programs in Java: Objects, Classes constructors, returning and passing objects as parameter, Inheritance, Access Control, using super, final with inheritance Overloading and overriding methods, Abstract classes, Extended classes

EXPERIMENT-3.1

OBJECTIVE

Implementation of Objects. Classes and Constructors in Java.

PROGRAM

```
class Goeduhub // creating a class
{
int id;
String name;
Goeduhub()  // creating default constructor
{
  System.out.println("Default constructor called!!");
}
//creating a parameterized constructor
Goeduhub(int i,String n)
{
  id = i;
  name = n;
  System.out.println("Parameterized constructor called!!");
}
Goeduhub(Goeduhub g)
{
```

```
    id=g.id;
    name=g.name;
    System.out.println("Copy constructor called!!");
  }
  //method to display the values
  void display()
  {
   System.out.println(id+" "+name);
   }
  public static void main(String args[])
  {
    Goeduhub g1 = new Goeduhub(); //creating objects and passing values
    g1.display(); //calling method to display the values of object
    Goeduhub g2 = new Goeduhub(123,"Ankit");
    g2.display();
    Goeduhub g3 = new Goeduhub(456,"Rohan");
    g3.display();
    Goeduhub g4 = new Goeduhub(g2); //passing object as parameter
    g4.display();
  }
}
```

OUTPUT

Default constructor called!!

0 null

Parameterized constructor called!!

123 Ankit

Parameterized constructor called!!

456 Rohan

Copy constructor called!!

123 Ankit

EXPERIMENT-3.2

OBJECTIVE

Implementation of Inheritance and Access control in Java

Program:

```
class lname{
void fun1() {  System.out.println("Technologies !");
} }
class fname extends lname{
void fun2() {
System.out.print("Poornima ");}
}
class Greet extends fname{
void fun3(){
System.out.print("Welcome to ");}
}
class Test1{
public static void main(String args[]){
Greet d=new Greet();
 d.fun3();
 d.fun2();
 d.fun1();
 }}
```

OUTPUT

Welcome to Poornima Technologies!

EXPERIMENT-3.3

OBJECTIVE

Implementation of super and final keywords in Java

Program: // superclass Person

```
class Person
{
   int id=111;
   void message()
   {
    System.out.println("Welcome to Goeduhub!");
   }
  Person()
   {
     System.out.println("Person class Constructor");
   }
}
// subclass Student extending the Person class
class Student extends Person // Inheritance
{
   Student()
    {
   super(); // invoke or call parent class constructor
 System.out.println("Student class Constructor");
 }
 void message()
```

```
    {
     System.out.println("Technologies");
    }
    void display()
    {
   super.message(); // calling super class method
   message();
  System.out.println("Student Id: "+super.id); //accessing super class variable
  }
}
class Test
{
 public static void main(String[] args)
 {
  Student s = new Student();
  s.display();
 }
}
```

OUTPUT

```
Person class Constructor
Student class Constructor
Welcome to Poornima!
Technologies
Student Id: 111
```

EXPERIMENT-3.4

OBJECTIVE

Implementation of overloading and overriding methods in Java

PROGRAM:

```
class Test1
 {
  static int add(int a,int b){return a+b;}
  static int add(int a,int b,int c){return a+b+c;}  //changing no. of arguments
  static double add(double a, double b){return a+b;} //changing data types
}
 class Test2
 {
 public static void main(String[] args)
  {
    System.out.println(Test1.add(24,41));
    System.out.println(Test1.add(24,41,11));
    System.out.println(Test1.add(13.2,14.6));
  }
}
```

OUTPUT

65

76

27.799999999999997

EXPERIMENT-3.5

OBJECTIVE

Implementation of Abstract classes in Java

Program:

```
// Abstract class
abstract class Animal {
 // Abstract method (does not have a body)
 public abstract void speak();
 // Regular method
 public void sleep() {
  System.out.println("Zzzzz!!!");
 }
}
// Subclass (inherit from Animal)
class Dog extends Animal {
 public void speak() {
  // The body of speak() is provided here
  System.out.println("Dog barks!");
 }
}
class Test3 {
 public static void main(String[] args) {
  Dog myDog = new Dog(); // Create a Dog object
  myDog.speak();
  myDog.sleep();
 }}
```

OUTPUT

```
Dog barks!
```