## 1. Implementation of overloading.

```
public class OverloadingExample {
  // Method to add two integers
  public static int add(int a, int b) {
    return a + b;
  }

  // Method to add three integers
  public static int add(int a, int b, int c) {
    return a + b + c;
  }

  public static void main(String[] args) {
    // Call the add method with two integers
    int sum1 = add(10, 20);
    System.out.println("Sum of 10 and 20: " + sum1);

    // Call the add method with three integers
    int sum2 = add(10, 20, 30);
    System.out.println("Sum of 10, 20, and 30: " + sum2);
  }
}
```

## 2. Inner class and two types of inheritance.

```
// Base class Employee
class Employee {
  private String name;
  private int id;

  // Constructor
  public Employee(String name, int id) {
    this.name = name;
    this.id = id;
  }

  // Method to display employee information
  public void displayInfo() {
    System.out.println("Name: " + name);
    System.out.println("ID: " + id);
  }
```

```
}

// Manager class inheriting from Employee (single inheritance)
class Manager extends Employee {
   private String department;

   // Constructor
   public Manager(String name, int id, String department) {
      super(name, id); // Calling superclass constructor
      this.department = department;
   }

   // Method to display manager information
   public void displayManagerInfo() {
      System.out.println("Department: " + department);
   }
}

// Worker class at the same level of hierarchy as Manager (hierarchical inheritance)
class Worker extends Employee {
   private String role;

   // Constructor
   public Worker(String name, int id, String role) {
      super(name, id); // Calling superclass constructor
      this.role = role;
   }

   // Method to display worker information
   public void displayWorkerInfo() {
      System.out.println("Role: " + role);
   }
}

public class EmployeeExample {
   public static void main(String[] args) {
      // Creating instances of Manager and Worker
      Manager manager = new Manager("John", 101, "Sales");
      Worker worker = new Worker("Alice", 201, "Developer");

      // Displaying information about Manager
```

```
        System.out.println("Manager information:");
        manager.displayInfo();
        manager.displayManagerInfo();

        // Displaying information about Worker
        System.out.println("\nWorker information:");
        worker.displayInfo();
        worker.displayWorkerInfo();
    }
}
```

**3. Implementing the concept of Interface.**

```
import java.io.*;

interface In1 {

    final int a = 10;
    void display();
}

class Interf implements In1 {
    public void display(){
     System.out.println("Hello");
    }

    public static void main(String[] args)
    {
        Interf t = new Interf();
        t.display();
        System.out.println(a);
    }
}
```

**4. Implementing the concept of package.**

```
package data;

// Class to which the above package belongs
public class Demo {

    // Member functions of the class- 'Demo'
```

```
  // Method 1 - To show()
  public void show()
  {

    // Print message
    System.out.println("Hi Everyone");
  }

  // Method 2 - To show()
  public void view()
  {
    // Print message
    System.out.println("Hello");
  }
}
```

**Procedure:**

**1.** To generate the output from the above program

**Command**: javac Demo.java

**2.** This Command Will Give Us a Class File

**Command**: javac -d . Demo.java

**3.** So This Command Will Create a New Folder Called **data**.

```
import data.*;

// Class to which the package belongs
class ncj {

  // main driver method
  public static void main(String arg[])
  {

    // Creating an object of Demo class
    Demo d = new Demo();

    // Calling the functions show() and view()
    // using the object of Demo class
    d.show();
    d.view();
  }
```

```
}
```

**5. Implementing the concept of Exception handling.**

```
public class SimpleException {
   public static void main(String[] args) {
      try {
         int result = 10/0;
         System.out.println("Result iss: " + result);
      } catch (ArithmeticException e) {
         System.err.println("Error: Division by zero is not allowed..");
      }
   }
}
```

**6. IP address and MAC address Displaying.**

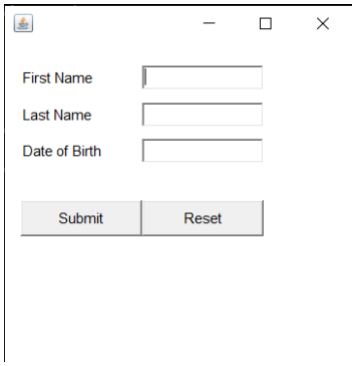
```
import java.net.*;
import java.util.*;
import java.net.InetAddress;

public class Ipmac{
  public static void main(String args[]) throws Exception {
  InetAddress address = InetAddress.getLocalHost();
  System.out.println("IP Address:" +address.getLocalHost());
  NetworkInterface networkInterface = NetworkInterface.getByInetAddress(address);
  byte[] mac = networkInterface.getHardwareAddress();
  System.out.print("MAC address  : ");

  StringBuilder stringBuilder = new StringBuilder();
  for (int i = 0; i < mac.length; i++) {
    stringBuilder.append(String.format("%02X%s", mac[i], (i < mac.length - 1) ? "-" : ""));
    }
    System.out.println(stringBuilder.toString());
  }
}
```

**7. Implementation of GUI using AWT.**

```
import java.awt.*;
public class AwtApp extends Frame  {
```

```
AwtApp(){
Label firstName = new Label("First Name");
firstName.setBounds(20, 50, 80, 20);

Label lastName = new Label("Last Name");
lastName.setBounds(20, 80, 80, 20);

Label dob = new Label("Date of Birth");
dob.setBounds(20, 110, 80, 20);

TextField firstNameTF = new TextField();
firstNameTF.setBounds(120, 50, 100, 20);

TextField lastNameTF = new TextField();
lastNameTF.setBounds(120, 80, 100, 20);

TextField dobTF = new TextField();
dobTF.setBounds(120, 110, 100, 20);

Button sbmt = new Button("Submit");
sbmt.setBounds(20, 160, 100, 30);

Button reset = new Button("Reset");
reset.setBounds(120,160,100,30);

add(firstName);
add(lastName);
add(dob);
add(firstNameTF);
add(lastNameTF);
add(dobTF);
add(sbmt);
add(reset);

setSize(300,300);
setLayout(null);
setVisible(true);
}
public static void main(String[] args) {
// TODO Auto-generated method stub
AwtApp awt = new AwtApp();
```

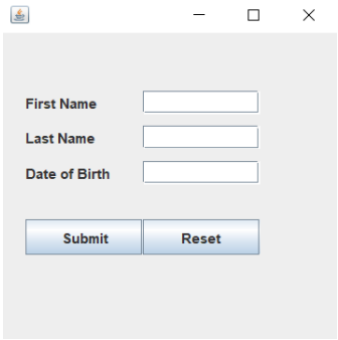
```
} }
```




## 8. Implementation of GUI using SWING.

```
import javax.swing.*;
public class SwingApp {
SwingApp(){
JFrame f = new JFrame();

JLabel firstName = new JLabel("First Name");
firstName.setBounds(20, 50, 80, 20);

JLabel lastName = new JLabel("Last Name");
lastName.setBounds(20, 80, 80, 20);

JLabel dob = new JLabel("Date of Birth");
dob.setBounds(20, 110, 80, 20);

JTextField firstNameTF = new JTextField();
firstNameTF.setBounds(120, 50, 100, 20);

JTextField lastNameTF = new JTextField();
lastNameTF.setBounds(120, 80, 100, 20);

JTextField dobTF = new JTextField();
dobTF.setBounds(120, 110, 100, 20);
```

```
JButton sbmt = new JButton("Submit");
sbmt.setBounds(20, 160, 100, 30);

JButton reset = new JButton("Reset");
reset.setBounds(120,160,100,30);

f.add(firstName);
f.add(lastName);
f.add(dob);
f.add(firstNameTF);
f.add(lastNameTF);
f.add(dobTF);
f.add(sbmt);
f.add(reset);

f.setSize(300,300);
f.setLayout(null);
f.setVisible(true);
}

public static void main(String[] args) {
// TODO Auto-generated method stub
SwingApp s = new SwingApp();
}
}
```




## 9. Assign Selected Color.

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
```

```
class FrameColor implements ActionListener
{
   static JFrame frame;
static JButton button = new JButton("Change Color");
   //Driver function
   public static void main(String args[])
   {
            //Create a frame
            frame = new JFrame("Change Frame Background");
            frame.setSize(400,400);
            frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            frame.getContentPane().setBackground(Color.white);
            frame.setLayout(new FlowLayout());
            //Create an object
            FrameColor obj = new FrameColor();
            //Create a button

            button.addActionListener(obj);
            frame.add(button);
            //Display the fame
            frame.setVisible(true);
   }
   //Function to create color dialog box and change color
   public void actionPerformed(ActionEvent e)
   {
            //Create a color dialog box
            JColorChooser color_box= new JColorChooser();
            Color color=color_box.showDialog(frame,"Select a Color",Color.white);
            //Change background color of frame
            //button.setBackground(color);
            frame.getContentPane().setBackground(color);
   }
}
```


10. Implementing the concept of Database Programming.

**11. Implementing the concept of Collection Classes.**

```
import java.util.ArrayList;
import java.util.Iterator;

public class CollectionExample {
  public static void main(String[] args) {
    // Create an ArrayList to store strings
    ArrayList<String> myList = new ArrayList<>();

    // Adding elements to the ArrayList
    myList.add("Apple");
    myList.add("Banana");
    myList.add("Orange");

    // Displaying elements using for-each loop
    System.out.println("Elements in the ArrayList:");
    for (String fruit : myList) {
      System.out.println(fruit);
    }

    // Adding an element at a specific index
    myList.add(1, "Mango");

    // Removing an element
    myList.remove("Banana");

    // Displaying elements using Iterator
    System.out.println("\nElements after modification:");
    Iterator<String> iterator = myList.iterator();
```

```
        while (iterator.hasNext()) {
            System.out.println(iterator.next());
        }

        // Checking if an element exists
        if (myList.contains("Apple")) {
            System.out.println("\nApple is present in the list.");
        } else {
            System.out.println("\nApple is not present in the list.");
        }

        // Getting the size of the ArrayList
        System.out.println("Size of the ArrayList: " + myList.size());
    }
}
```

**12. Find Interest by using function or constructor.**

**Using Func.**

```
import java.util.Scanner;

public class SiInterest {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter principal amount: ");
        double principal = scanner.nextDouble();

        System.out.print("Enter rate of interest (in percentage): ");
        double rate = scanner.nextDouble();

        System.out.print("Enter time period (in years): ");
        double time = scanner.nextDouble();

        // Call the function to calculate simple interest
        double simpleInterest = calculateSimpleInterest(principal, rate, time);

        System.out.println("Simple Interest: " + simpleInterest);

        scanner.close();
    }
```

```
    // Function to calculate simple interest
    public static double calculateSimpleInterest(double principal, double rate, double time) {
        return (principal * rate * time) / 100;
    }
}
```

**13. File Handling – Create, Read, Write.**

```
import java.io.*;
import java.util.Scanner;

public class FHandling{
        public static void main(String[] args)
        {
                //Create
                try {
                        File Obj = new File("myfile.txt");
                        if (Obj.createNewFile()) {
                                System.out.println("File created: "+ Obj.getName());
                        }
                        else {
                                System.out.println("File already exists.");
                        }
                }
                catch (IOException e) {
                        System.out.println("An error has occurred.");
                        e.printStackTrace();
                }

                //Read
                try {
                        File Obj = new File("myfile.txt");
                        Scanner Reader = new Scanner(Obj);
                        while (Reader.hasNextLine()) {
                                String data = Reader.nextLine();
                                System.out.println(data);
                        }
                        Reader.close();
                }
                catch (FileNotFoundException e) {
                        System.out.println("An error has occurred.");
```

```
                e.printStackTrace();
        }

        //Write
        try {
                FileWriter Writer
                        = new FileWriter("myfile.txt");
                Writer.write(
                        "Files in Java are seriously good!!");
                Writer.close();
                System.out.println("Successfully written.");
        }
        catch (IOException e) {
                System.out.println("An error has occurred.");
                e.printStackTrace();
        }


    }
}
```