1. Find interest using function or constructor (Assume required values at your own)

```
import java.util.Scanner;

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

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

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

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

      // Calculate simple interest using the function
      double interest = calculateSimpleInterest(principal, rate, time);

      // Print the result
      System.out.println("Simple Interest: " + interest);

      scanner.close();
   }

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

2. Implementation of method overloading and overriding

Method Overloading:

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

   // Overloaded method to add three integers
   public int add(int a, int b, int c) {
      return a + b + c;
   }
```

```
    // Overloaded method to add two doubles
    public double add(double a, double b) {
        return a + b;
    }
}

public class MethodOverloadingExample {
    public static void main(String[] args) {
        Calculator calc = new Calculator();

        // Calls the add method with two integers
        System.out.println("Sum of 3 and 5: " + calc.add(3, 5));

        // Calls the add method with three integers
        System.out.println("Sum of 3, 5, and 7: " + calc.add(3, 5, 7));

        // Calls the add method with two doubles
        System.out.println("Sum of 3.5 and 2.5: " + calc.add(3.5, 2.5));
    }
}
```

Method Overriding:

```
class Animal {
    void makeSound() {
        System.out.println("Animal makes a sound");
    }
}

class Dog extends Animal {
    @Override
    void makeSound() {
        System.out.println("Dog barks");
    }
}

public class MethodOverridingExample {
    public static void main(String[] args) {
        Animal animal = new Animal();
        animal.makeSound(); // Output: Animal makes a sound

        Dog dog = new Dog();
        dog.makeSound(); // Output: Dog barks
    }
}
```

3. Implementation of single and multilevel Inheritance.✅

Single Inheritance:

```
// Parent class
class Animal {
   void eat() {
      System.out.println("Animal is eating");
   }
}

// Child class inheriting from Animal
class Dog extends Animal {
   void bark() {
      System.out.println("Dog is barking");
   }
}

public class SingleInheritanceExample {
   public static void main(String[] args) {
      Dog dog = new Dog();
      dog.eat(); // Inherited method from Animal class
      dog.bark(); // Method from Dog class
   }
}
```

Multilevel Inheritance:

```
// Grandparent class
class Animal {
   void eat() {
      System.out.println("Animal is eating");
   }
}

// Parent class inheriting from Animal
class Dog extends Animal {
   void bark() {
      System.out.println("Dog is barking");
   }
}

// Child class inheriting from Dog
class Puppy extends Dog {
   void sleep() {
      System.out.println("Puppy is sleeping");
   }
}
```

```
public class MultilevelInheritanceExample {
    public static void main(String[] args) {
        Puppy puppy = new Puppy();
        puppy.eat(); // Inherited method from Animal class
        puppy.bark(); // Inherited method from Dog class
        puppy.sleep(); // Method from Puppy class
    }
}
```

4. Implementation of Inner Class✅

```
public class OuterClass {
    private int outerData;

    // Constructor
    public OuterClass(int data) {
        outerData = data;
    }

    // Inner class
    public class InnerClass {
        private int innerData;

        // Constructor
        public InnerClass(int data) {
            innerData = data;
        }

        // Method to display inner data
        public void displayInnerData() {
            System.out.println("Inner data: " + innerData);
        }
    }

    public static void main(String[] args) {
        // Create an instance of OuterClass
        OuterClass outer = new OuterClass(10);

        // Create an instance of InnerClass
        OuterClass.InnerClass inner = outer.new InnerClass(20);

        // Display inner data
        inner.displayInnerData();
    }
}
```

5. Implementation 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);
   }
}
```

6. Create your own package and use properties of it in another program 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**.

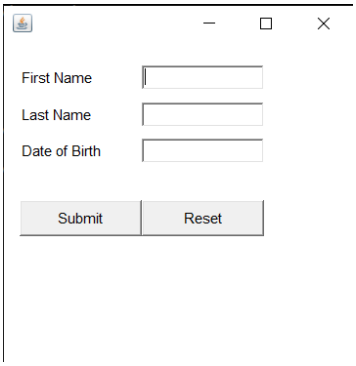
```
import data.*;

// 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();
   }
}
```

7. Implementation 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..");
      }
   }
}
```

8. Implementation of GUI using AWT ✓

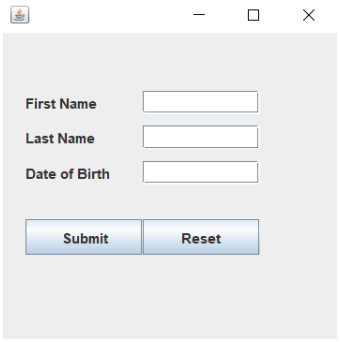
```
import java.awt.*;
public class AwtApp extends Frame  {
AwtApp(){
Label firstName = new Label("First Name");
firstName.setBounds(20, 50, 80, 20);

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();

} }
```



9. 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);
```

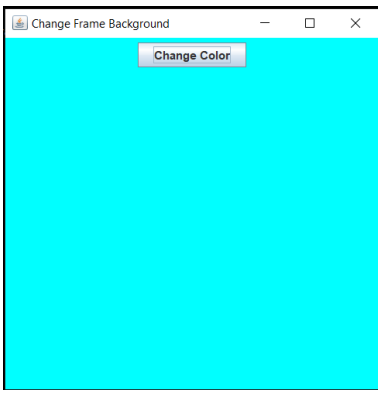
```
f.add(sbmt);
f.add(reset);

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();
}
}
```



10. Open color dialog and assign selected color to any component✓

```
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);
    }
}
```



11. Implementing the concept of Mouse Event

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

public class MouseEventExample extends JFrame implements MouseListener {
    JLabel label;

    public MouseEventExample() {
        label = new JLabel();
        label.setBounds(20, 20, 200, 50);
        label.addMouseListener(this);

        add(label);

        setSize(300, 200);
        setLayout(null);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setVisible(true);
    }
```

```
    public void mouseClicked(MouseEvent e) {
      label.setText("Mouse Clicked");
    }

    public void mouseEntered(MouseEvent e) {
      label.setText("Mouse Entered");
    }

    public void mouseExited(MouseEvent e) {
      label.setText("Mouse Exited");
    }

    public void mousePressed(MouseEvent e) {
      label.setText("Mouse Pressed");
    }

    public void mouseReleased(MouseEvent e) {
      label.setText("Mouse Released");
    }

    public static void main(String[] args) {
      new MouseEventExample();
    }
}
```

12. Display IP address and MAC address✅

```
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());
  }
}
```

13. Implementation of Database Programming (select data from table)

```
package test11;
import java.sql.*;

public class MysqlCon {

        public static void main(String[] args) {try{
                Class.forName("com.mysql.jdbc.Driver");
                Connection con=DriverManager.getConnection(
                "jdbc:mysql://127.0.0.1:Port/dbname","root","");
                //here sonoo is database name, root is username and password
                Statement stmt=con.createStatement();
                ResultSet rs=stmt.executeQuery("select * from sample");
                System.out.println("hi");
                while(rs.next())
                System.out.println(rs.getInt(1)+""+rs.getString(2));
                con.close();
                }catch(Exception e){ System.out.println(e);}
                // TODO Auto-generated method stub

        }

}
```

14. Implementation of Database Programming (Insert data into table)

```
public class JdbcAccessTest {

   public static void main(String[] args) {

      String databaseURL = "jdbc:mydsn:ucanaccess://D:/Java/Contacts.accdb";

      try (Connection connection = DriverManager.getConnection(databaseURL)) {


         /*String sql = "INSERT INTO Contacts (Full_Name, Email, Phone) VALUES (?, ?, ?)";

         PreparedStatement preparedStatement = connection.prepareStatement(sql);
         preparedStatement.setString(1, "Jim Rohn");
         preparedStatement.setString(2, "rohnj@herbalife.com");
         preparedStatement.setString(3, "0919989998");

         int row = preparedStatement.executeUpdate();

         if (row > 0) {
            System.out.println("A row has been inserted successfully.");
         }
```

```
            */
            String sql = "SELECT * FROM Contacts";

            Statement statement = connection.createStatement();
            ResultSet result = statement.executeQuery(sql);

            while (result.next()) {
                int id = result.getInt("Contact_Id");
                String fullname = result.getString("Full_Name");
                String email = result.getString("Email");
                String phone = result.getString("Phone");

                System.out.println(id + ", " + fullname + ", " + email + ", " + phone);
            }

        } catch (SQLException ex) {
            ex.printStackTrace();
        }
    }
}
```

15. Implementation of Collection Classes (any three classes)✅

```
import java.util.ArrayList;
import java.util.HashMap;
import java.util.HashSet;
import java.util.Map;
import java.util.Set;

public class CollectionClasses {
    public static void main(String[] args) {
        // ArrayList
        ArrayList<String> arrayList = new ArrayList<>();
        arrayList.add("Apple");
        arrayList.add("Banana");
        arrayList.add("Orange");
        System.out.println("ArrayList: " + arrayList);

        // HashMap
        HashMap<Integer, String> hashMap = new HashMap<>();
        hashMap.put(1, "One");
        hashMap.put(2, "Two");
        hashMap.put(3, "Three");
        System.out.println("HashMap: " + hashMap);

        // HashSet
        HashSet<String> hashSet = new HashSet<>();
        hashSet.add("Apple");
```

```
        hashSet.add("Banana");
        hashSet.add("Orange");
        System.out.println("HashSet: " + hashSet);
    }
}
```

16. File handling: Create file and write content into file✓

```
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();
            }


        }
}
```

17. Implementation of string functions (Any four functions)

```
public class StringFunctions {
   public static void main(String[] args) {
      // Example string
      String str = "Hello, World!";

      // Length of the string
      int length = str.length();
      System.out.println("Length of the string: " + length);

      // Character at index
      char charAtIndex = str.charAt(7);
      System.out.println("Character at index 7: " + charAtIndex);

      // Substring
      String substring = str.substring(7, 12);
      System.out.println("Substring from index 7 to 12: " + substring);

      // Uppercase
      String uppercase = str.toUpperCase();
      System.out.println("Uppercase string: " + uppercase);
   }
}
```