12.Write GUI program to find factorial of given number using applet.

```
import javax.swing.*;

import java.awt.*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;


public class FactorialCalculatorSwing extends JFrame implements ActionListener {

    private JTextField inputField;

    private JLabel resultLabel;


    public FactorialCalculatorSwing() {

        setTitle("Factorial Calculator");

        setSize(300, 150);

        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);


        // Create and configure components

        inputField = new JTextField(10);

        JButton calculateButton = new JButton("Calculate Factorial");

        calculateButton.addActionListener(this);


        resultLabel = new JLabel("Factorial: ");


        // Create and configure a JPanel to hold components

        JPanel panel = new JPanel();

        panel.add(new JLabel("Enter a number: "));

        panel.add(inputField);

        panel.add(calculateButton);

        panel.add(resultLabel);
        // Add the panel to the frame
        add(panel);

        // Set the frame to be visible
        setVisible(true);
    }

    public void actionPerformed(ActionEvent e) {
        if (e.getActionCommand().equals("Calculate Factorial")) {
            try {
                int number = Integer.parseInt(inputField.getText());
                long factorial = calculateFactorial(number);
                resultLabel.setText("Factorial: " + factorial);
            } catch (NumberFormatException ex) {
                resultLabel.setText("Invalid input. Enter a valid number.");
            }
        }
    }

    private long calculateFactorial(int n) {
        if (n < 0) {
            return -1; // Factorial is not defined for negative numbers
        } else if (n == 0 || n == 1) {
            return 1;
        } else {
            long result = 1;
            for (int i = 2; i <= n; i++) {
                result *= i;
            }
            return result;
        }
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> new FactorialCalculatorSwing());
    }
}
```

13.Extending the Thread class

```
class MyThread extends Thread {
    @Override
    public void run() {
        // Code to be executed in the new thread
        for (int i = 1; i <= 5; i++) {
            System.out.println("Thread: " + i);
            try {
                Thread.sleep(1000); // Sleep for 1 second
            } catch (InterruptedException e) {
                System.out.println("Thread interrupted");
            }
        }
    }
}

public class thread {
    public static void main(String[] args) {
        MyThread myThread = new MyThread(); // Create an instance of the custom thread class
        myThread.start(); // Start the thread

        // Code in the main thread
        for (int i = 1; i <= 5; i++) {
            System.out.println("Main: " + i);
            try {
                Thread.sleep(1000); // Sleep for 1 second
            } catch (InterruptedException e) {
                System.out.println("Main thread interrupted");
            }
        }
    }
}
```

14.Write a program to perform union, intersect and difference of two sets.

```
import java.util.HashSet;
import java.util.Set;

public class set_functions {
    public static void main(String[] args) {
        // Create two sets
        Set<Integer> set1 = new HashSet<>();
        Set<Integer> set2 = new HashSet<>();

        // Add elements to the first set
        set1.add(1);
        set1.add(2);
        set1.add(3);
        set1.add(4);

        // Add elements to the second set
        set2.add(3);
        set2.add(4);
        set2.add(5);
        set2.add(6);

        // Perform union
        Set<Integer> union = new HashSet<>(set1);
        union.addAll(set2);
        System.out.println("Union: " + union);

        // Perform intersection
        Set<Integer> intersection = new HashSet<>(set1);
        intersection.retainAll(set2);
        System.out.println("Intersection: " + intersection);

        // Perform difference (set1 - set2)
        Set<Integer> difference1 = new HashSet<>(set1);
        difference1.removeAll(set2);
        System.out.println("Difference (set1 - set2): " + difference1);

        // Perform difference (set2 - set1)
        Set<Integer> difference2 = new HashSet<>(set2);
        difference2.removeAll(set1);
        System.out.println("Difference (set2 - set1): " + difference2);
    }
}
```

```
}
```

15.Write java program to demonstrate Hierarchical inheritance.

```
class Student {
    int rollNumber;

    Student(int rollNumber) {
        this.rollNumber = rollNumber;
    }
}

class Test extends Student {
    int sub1;
    int sub2;

    Test(int rollNumber, int sub1, int sub2) {
        super(rollNumber);
        this.sub1 = sub1;
        this.sub2 = sub2;
    }
}

class Result extends Test {
    Result(int rollNumber, int sub1, int sub2) {
        super(rollNumber, sub1, sub2);
    }

    void displayResult() {
        int totalMarks = sub1 + sub2;
        System.out.println("Roll Number: " + rollNumber);

        System.out.println("Subject 1 Marks: " + sub1);

        System.out.println("Subject 2 Marks: " + sub2);

        System.out.println("Total Marks: " + totalMarks);

    }

}


public class Main {

    public static void main(String[] args) {

        Result result = new Result(101, 85, 90);

        result.displayResult();

    }

}
```

16.Write java program to demonstrate Multilevel inheritance

```
class Animal {

    void eat() {

        System.out.println("Animals eat food.");

    }

}


class Dog extends Animal {

    void bark() {

        System.out.println("Dogs can bark.");

    }

}


class GoldenRetriever extends Dog {
    void playFetch() {
        System.out.println("Golden Retrievers can play fetch.");
    }
}

public class MultilevelInheritanceDemo {
    public static void main(String[] args) {
        GoldenRetriever dog = new GoldenRetriever();

        // Methods from the Animal class
        dog.eat();

        // Methods from the Dog class
        dog.bark();

        // Methods from the GoldenRetriever class
        dog.playFetch();
    }
}
```

17.Write a java Program to demonstrate Itemevent

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

public class ItemEventDemo extends Frame implements ItemListener {
    private Checkbox checkBox;

    public ItemEventDemo() {
```

```
    setTitle("ItemEvent Demo");

    setSize(300, 200);

    setLayout(new FlowLayout());


    checkBox = new Checkbox("Check Me");

    checkBox.addItemListener(this);


    add(checkBox);


    addWindowListener(new WindowAdapter() {

      public void windowClosing(WindowEvent we) {

        System.exit(0);

      }

    });

  }


  public void itemStateChanged(ItemEvent e) {

    if (e.getSource() == checkBox) {

      if (checkBox.getState()) {

        System.out.println("Checkbox is checked.");

      } else {

        System.out.println("Checkbox is unchecked.");

      }

    }

  }


  public static void main(String[] args) {

    ItemEventDemo demo = new ItemEventDemo();

    demo.setVisible(true);
    }
}
```

18.Write a java program to demonstrate BorderLayout() using Applet

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

public class BorderLayoutSwing extends JFrame {
    public BorderLayoutSwing() {
        setTitle("BorderLayout Example");
        setSize(400, 300);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        JPanel panel = new JPanel();
        panel.setLayout(new BorderLayout());

        panel.add(new JButton("North"), BorderLayout.NORTH);
        panel.add(new JButton("South"), BorderLayout.SOUTH);
        panel.add(new JButton("East"), BorderLayout.EAST);
        panel.add(new JButton("West"), BorderLayout.WEST);
        panel.add(new JButton("Center"), BorderLayout.CENTER);

        add(panel);
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            BorderLayoutSwing app = new BorderLayoutSwing();
            app.setVisible(true);
        });
    }
}
```

or

```
import java.applet.Applet;
import java.awt.BorderLayout;
import java.awt.Button;

public class BorderLayoutApplet extends Applet {
    public void init() {
        setLayout(new BorderLayout());

        Button northButton = new Button("North");
        Button southButton = new Button("South");
        Button eastButton = new Button("East");
        Button westButton = new Button("West");
        Button centerButton = new Button("Center");

        add(northButton, BorderLayout.NORTH);
        add(southButton, BorderLayout.SOUTH);
        add(eastButton, BorderLayout.EAST);
        add(westButton, BorderLayout.WEST);
        add(centerButton, BorderLayout.CENTER);
    }
}
```

19.Write a Program to demonstrate Grid layout ()

```
import javax.swing.*;

import java.awt.*;


public class GridLayoutDemo {

   public static void main(String[] args) {

      // Create a JFrame

      JFrame frame = new JFrame("GridLayout Example");

      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

      frame.setSize(300, 300);


      // Create a JPanel with a 3x3 grid layout

      JPanel panel = new JPanel(new GridLayout(3, 3));


      // Create buttons and add them to the panel

      for (int i = 1; i <= 9; i++) {

         JButton button = new JButton("Button " + i);

         panel.add(button);

      }


      // Add the panel to the frame

      frame.add(panel);


      // Set the frame to be visible

      frame.setVisible(true);

   }

}
```

19.Write a Program to demonstrate Grid layout ()

```
import java.awt.*;

import javax.swing.*;


public class grid_layout {

   public static void main(String[] args) {

      // Create a JFrame

      JFrame frame = new JFrame("GridLayout Demo");

      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

      frame.setSize(300, 200);


      // Create a panel with a GridLayout

      JPanel panel = new JPanel();

      panel.setLayout(new GridLayout(3, 2)); // 3 rows and 2 columns


      // Create and add components to the panel

      panel.add(new JButton("Button 1"));

      panel.add(new JButton("Button 2"));

      panel.add(new JButton("Button 3"));

      panel.add(new JButton("Button 4"));

      panel.add(new JButton("Button 5"));

      panel.add(new JButton("Button 6"));


      // Add the panel to the frame

      frame.add(panel);


      frame.setVisible(true);

   }
}
```

20Write a java program to Read contents of file using Scanner class.

```
import java.io.File;
import java.io.FileNotFoundException;
import java.util.Scanner;

public class file_scanner {
    public static void main(String[] args) {
        // Specify the path to the file you want to read
        String filePath = "sample.txt";

        try {
            // Create a File object with the specified file path
            File file = new File(filePath);

            // Create a Scanner to read from the file
            Scanner scanner = new Scanner(file);

            // Read and display the contents of the file line by line
            while (scanner.hasNextLine()) {
                String line = scanner.nextLine();
                System.out.println(line);
            }

            // Close the scanner
            scanner.close();
        } catch (FileNotFoundException e) {
            System.err.println("File not found: " + e.getMessage());
        }
    }
}
```