1.Quadratic Equation

import java.util.\*;

class QuadraticEquationExample

{

public static void main(String[] Strings)

{

Scanner sc = new Scanner(System.in);

System.out.print("Enter the value of a: ");

double a = sc.nextDouble();

System.out.print("Enter the value of b: ");

double b = sc.nextDouble();

System.out.print("Enter the value of c: ");

double c = sc.nextDouble();

double d= b \* b - 4.0 \* a \* c;

if (d> 0.0)

{

double r1 = (-b + Math.pow(d, 0.5)) / (2.0 \* a);

double r2 = (-b - Math.pow(d, 0.5)) / (2.0 \* a);

System.out.println("The roots are " + r1 + " and " + r2);

}

else if (d == 0.0)

{

double r1 = -b / (2.0 \* a);

System.out.println("The root is " + r1);

}

else

{

System.out.println("Roots are not real.");

}

}

}

2.Mouse Events

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class MouseEventsDemo extends JFrame implements MouseListener, MouseMotionListener {

    private JLabel statusLabel;

    public MouseEventsDemo() {

        setTitle("Mouse Events Demo");

        setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

        setLayout(new FlowLayout());

        statusLabel = new JLabel("No mouse event");

        add(statusLabel);

        addMouseListener(this);

        addMouseMotionListener(this);

        setSize(300, 200);

        setVisible(true);

    }

    public static void main(String[] args) {

        SwingUtilities.invokeLater(new Runnable() {

            public void run() {

                new MouseEventsDemo();

            }

        });

    }

    @Override

    public void mouseClicked(MouseEvent e) {

        statusLabel.setText("Mouse Clicked at (" + e.getX() + ", " + e.getY() + ")");

    }

    @Override

    public void mousePressed(MouseEvent e) {

        statusLabel.setText("Mouse Pressed at (" + e.getX() + ", " + e.getY() + ")");

    }

    @Override

    public void mouseReleased(MouseEvent e) {

        statusLabel.setText("Mouse Released at (" + e.getX() + ", " + e.getY() + ")");

    }

    @Override

    public void mouseEntered(MouseEvent e) {

        statusLabel.setText("Mouse Entered at (" + e.getX() + ", " + e.getY() + ")");

    }

    @Override

    public void mouseExited(MouseEvent e) {

        statusLabel.setText("Mouse Exited at (" + e.getX() + ", " + e.getY() + ")");

    }

    @Override

    public void mouseDragged(MouseEvent e) {

        statusLabel.setText("Mouse Dragged at (" + e.getX() + ", " + e.getY() + ")");

    }

    @Override

    public void mouseMoved(MouseEvent e) {

        statusLabel.setText("Mouse Moved at (" + e.getX() + ", " + e.getY() + ")");

    }

}

3.Marks Grading System

public class GradingSystem {

public static void main(String[] args) {

    int marks=65;

    if(marks<50){

        System.out.println("fail");

   }

    else if(marks>=50 && marks<60){

        System.out.println("D grade");

    }

    else if(marks>=60 && marks<70){

        System.out.println("C grade");

    }

    else if(marks>=70 && marks<80){

        System.out.println("B grade");

    }

    else if(marks>=80 && marks<90){

        System.out.println("A grade");

    }else if(marks>=90 && marks<100){

        System.out.println("A+ grade");

    }else{

        System.out.println("Invalid!");

    }

}

}

4.More than one thread

public class MultiThreadDemo {

    public static void main(String[] args) {

        // Create and start the first thread

        Thread thread1 = new Thread(new MyRunnable("Thread 1"));

        thread1.start();

        // Create and start the second thread

        Thread thread2 = new Thread(new MyRunnable("Thread 2"));

        thread2.start();

        // Create and start the third thread

        Thread thread3 = new Thread(new MyRunnable("Thread 3"));

        thread3.start();

    }

}

class MyRunnable implements Runnable {

    private String threadName;

    public MyRunnable(String threadName) {

        this.threadName = threadName;

    }

    public void run() {

        System.out.println("Thread " + threadName + " is running.");

        try {

            // Simulate some work

            Thread.sleep(2000);

        } catch (InterruptedException e) {

            e.printStackTrace();

        }

        System.out.println("Thread " + threadName + " has finished.");

    }

}

5.Armstrong Number

public class Armstrong {

    public static void main(String[] args) {

        int number = 371, originalNumber, remainder, result = 0;

        originalNumber = number;

        while (originalNumber != 0)

        {

            remainder = originalNumber % 10;

            result += Math.pow(remainder, 3);

            originalNumber /= 10;

        }

        if(result == number)

            System.out.println(number + " is an Armstrong number.");

        else

            System.out.println(number + " is not an Armstrong number.");

    }

}

6. Set Using HashSet

import java.util.HashSet;

import java.util.Set;

public class Hashset {

    public static void main(String[] args) {

        // Create a HashSet

        Set<String> set = new HashSet<>();

        // Add elements to the set

        set.add("Apple");

        set.add("Banana");

        set.add("Orange");

        set.add("Grapes");

        set.add("Apple"); // Adding a duplicate element

        // Print the set

        System.out.println("Set elements: " + set);

        // Check if an element exists in the set

        String element = "Banana";

        if (set.contains(element)) {

            System.out.println(element + " is present in the set.");

        } else {

            System.out.println(element + " is not present in the set.");

        }

        // Remove an element from the set

        String removeElement = "Orange";

        boolean removed = set.remove(removeElement);

        if (removed) {

            System.out.println(removeElement + " was removed from the set.");

        } else {

            System.out.println(removeElement + " is not present in the set.");

        }

        // Print the updated set

        System.out.println("Updated set elements: " + set);

        // Iterate over the set using enhanced for loop

        System.out.println("Iterating over the set using enhanced for loop:");

        for (String item : set) {

            System.out.println(item);

        }

        // Clear the set

        set.clear();

        System.out.println("Set cleared. Size: " + set.size());

    }

}

7.Constructor Overloading

public class ConstructorOverloading {

    private int number;

    private String name;

    // Default constructor

    public ConstructorOverloading() {

        number = 0;

        name = "";

    }

    // Constructor with one parameter

    public ConstructorOverloading(int number) {

        this.number = number;

        name = "";

    }

    // Constructor with two parameters

    public ConstructorOverloading(int number, String name) {

        this.number = number;

        this.name = name;

    }

    // Method to display the values

    public void display() {

        System.out.println("Number: " + number);

        System.out.println("Name: " + name);

    }

    public static void main(String[] args) {

        // Create objects using different constructors

        ConstructorOverloading obj1 = new ConstructorOverloading();

        ConstructorOverloading obj2 = new ConstructorOverloading(10);

        ConstructorOverloading obj3 = new ConstructorOverloading(20, "John");

        // Display the values

        System.out.println("Object 1:");

        obj1.display();

        System.out.println();

        System.out.println("Object 2:");

        obj2.display();

        System.out.println();

        System.out.println("Object 3:");

        obj3.display();

    }

}

8.String Operations

public class StringOperations {

    public static void main(String[] args) {

        String str1 = "Hello";

        String str2 = "World";

        // Concatenation

        String concat = str1 + " " + str2;

        System.out.println("Concatenation: " + concat);

        // Length

        int length = str1.length();

        System.out.println("Length of str1: " + length);

        // Substring

        String substring = str2.substring(0, 3);

        System.out.println("Substring of str2: " + substring);

        // Uppercase and lowercase

        String uppercase = str1.toUpperCase();

        String lowercase = str2.toLowerCase();

        System.out.println("Uppercase: " + uppercase);

        System.out.println("Lowercase: " + lowercase);

        // Comparison

        boolean isEqual = str1.equals(str2);

        boolean isEqualIgnoreCase = str1.equalsIgnoreCase(str2);

        System.out.println("Comparison using equals: " + isEqual);

        System.out.println("Comparison using equalsIgnoreCase: " + isEqualIgnoreCase);

        // IndexOf

        int index = str1.indexOf('o');

        System.out.println("Index of 'o' in str1: " + index);

        // Replace

        String replaced = str1.replace('l', 'x');

        System.out.println("Replacement in str1: " + replaced);

        // Split

        String sentence = "This is a sample sentence.";

        String[] words = sentence.split(" ");

        System.out.println("Splitting sentence:");

        for (String word : words) {

            System.out.println(word);

        }

        // Trim

        String spacedString = "   Hello, World!    ";

        String trimmed = spacedString.trim();

        System.out.println("Trimmed string: " + trimmed);

    }

}

9.Multi-level Inheritance

class Animal {

    public void eat() {

        System.out.println("Animal is eating.");

    }

}

class Dog extends Animal {

    public void bark() {

        System.out.println("Dog is barking.");

    }

}

class Labrador extends Dog {

    public void run() {

        System.out.println("Labrador is running.");

    }

}

public class MultiLevelInheritance {

    public static void main(String[] args) {

        Labrador labrador = new Labrador();

        labrador.eat();  // Inherited from Animal class

        labrador.bark(); // Inherited from Dog class

        labrador.run();  // Defined in Labrador class

    }

}

10.Menu Bar

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class MenuBarDemo extends JFrame {

    public MenuBarDemo() {

        setTitle("Menu Bar Demo");

        setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

        // Create the menu bar

        JMenuBar menuBar = new JMenuBar();

        // Create the menus

        JMenu fileMenu = new JMenu("File");

        JMenu editMenu = new JMenu("Edit");

        JMenu helpMenu = new JMenu("Help");

        // Create the menu items

        JMenuItem newItem = new JMenuItem("New");

        JMenuItem openItem = new JMenuItem("Open");

        JMenuItem saveItem = new JMenuItem("Save");

        JMenuItem exitItem = new JMenuItem("Exit");

        JMenuItem cutItem = new JMenuItem("Cut");

        JMenuItem copyItem = new JMenuItem("Copy");

        JMenuItem pasteItem = new JMenuItem("Paste");

        JMenuItem aboutItem = new JMenuItem("About");

        // Add menu items to the menus

        fileMenu.add(newItem);

        fileMenu.add(openItem);

        fileMenu.add(saveItem);

        fileMenu.addSeparator();

        fileMenu.add(exitItem);

        editMenu.add(cutItem);

        editMenu.add(copyItem);

        editMenu.add(pasteItem);

        helpMenu.add(aboutItem);

        // Add the menus to the menu bar

        menuBar.add(fileMenu);

        menuBar.add(editMenu);

        menuBar.add(helpMenu);

        // Set the menu bar for the frame

        setJMenuBar(menuBar);

        // Create a label to display selected menu item

        JLabel label = new JLabel("Selected menu item");

        // Add the label to the frame

        add(label, BorderLayout.CENTER);

        // Add action listeners to menu items

        exitItem.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent e) {

                System.exit(0);

            }

        });

        aboutItem.addActionListener(new ActionListener() {

            public void actionPerformed(ActionEvent e) {

                label.setText("About menu item selected");

            }

        });

        setSize(300, 200);

        setVisible(true);

    }

    public static void main(String[] args) {

        SwingUtilities.invokeLater(new Runnable() {

            public void run() {

                new MenuBarDemo();

            }

        });

    }

}

11.Runtime Polymorphism

class Animal {

    public void makeSound() {

        System.out.println("Animal is making a sound");

    }

}

class Dog extends Animal {

    public void makeSound() {

        System.out.println("Dog is barking");

    }

}

class Cat extends Animal {

    public void makeSound() {

        System.out.println("Cat is meowing");

    }

}

public class RuntimePolymorphism {

    public static void main(String[] args) {

        Animal animal1 = new Animal();

        Animal animal2 = new Dog();

        Animal animal3 = new Cat();

        animal1.makeSound();

        animal2.makeSound();

        animal3.makeSound();

    }

}

12.Reading a Text File Character Stream class

import java.io.BufferedReader;

import java.io.FileReader;

import java.io.IOException;

public class ReadTextFile {

    public static void main(String[] args) {

        String filePath = "1234.txt"; // Replace with the actual file path

        try (BufferedReader reader = new BufferedReader(new FileReader(filePath))) {

            String line;

            while ((line = reader.readLine()) != null) {

                System.out.println(line);

            }

        } catch (IOException e) {

            System.out.println("An error occurred while reading the file: " + e.getMessage());

        }

    }

}

13.String Tockenizer Object

import java.util.StringTokenizer;

public class StringTokenizerDemo {

    public static void main(String[] args) {

        String sentence = "Hello, World! How are you?";

        StringTokenizer tokenizer = new StringTokenizer(sentence, ", !");

        // Count the number of tokens

        int tokenCount = tokenizer.countTokens();

        System.out.println("Number of tokens: " + tokenCount);

        // Iterate through the tokens and print them

        System.out.println("Tokens:");

        while (tokenizer.hasMoreTokens()) {

            String token = tokenizer.nextToken();

            System.out.println(token);

        }

    }

}

14.Card Layout

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.\*;

public class CardLayoutDemo extends JFrame implements ActionListener {

    private JPanel cards;

    private JButton previousButton;

    private JButton nextButton;

    private CardLayout cardLayout;

    public CardLayoutDemo() {

        setTitle("Card Layout Demo");

        setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

        // Create the cards panel

        cards = new JPanel();

        cardLayout = new CardLayout();

        cards.setLayout(cardLayout);

        // Create and add cards to the panel

        JPanel card1 = createCard("Card 1", Color.RED);

        JPanel card2 = createCard("Card 2", Color.GREEN);

        JPanel card3 = createCard("Card 3", Color.BLUE);

        cards.add(card1, "Card 1");

        cards.add(card2, "Card 2");

        cards.add(card3, "Card 3");

        // Create the navigation buttons

        previousButton = new JButton("Previous");

        nextButton = new JButton("Next");

        previousButton.addActionListener(this);

        nextButton.addActionListener(this);

        // Create the button panel

        JPanel buttonPanel = new JPanel();

        buttonPanel.add(previousButton);

        buttonPanel.add(nextButton);

        // Add the cards panel and button panel to the frame

        add(cards, BorderLayout.CENTER);

        add(buttonPanel, BorderLayout.SOUTH);

        pack();

        setVisible(true);

    }

    private JPanel createCard(String name, Color color) {

        JPanel card = new JPanel();

        card.setBackground(color);

        JLabel label = new JLabel(name);

        label.setFont(new Font("Arial", Font.BOLD, 24));

        card.add(label);

        return card;

    }

    public void actionPerformed(ActionEvent e) {

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

            cardLayout.previous(cards);

        } else if (e.getSource() == nextButton) {

            cardLayout.next(cards);

        }

    }

    public static void main(String[] args) {

        SwingUtilities.invokeLater(new Runnable() {

            public void run() {

                new CardLayoutDemo();

            }

        });

    }

}

15.Array List

import java.util.ArrayList;

public class ArrayListDemo {

    public static void main(String[] args) {

        // Create an ArrayList of strings

        ArrayList<String> names = new ArrayList<>();

        // Add elements to the ArrayList

        names.add("Alice");

        names.add("Bob");

        names.add("Charlie");

        // Get the size of the ArrayList

        int size = names.size();

        System.out.println("Size of ArrayList: " + size);

        // Access elements in the ArrayList

        String firstElement = names.get(0);

        System.out.println("First element: " + firstElement);

        // Update an element in the ArrayList

        names.set(1, "Eve");

        // Remove an element from the ArrayList

        names.remove(2);

        // Check if an element exists in the ArrayList

        boolean containsCharlie = names.contains("Charlie");

        System.out.println("Contains 'Charlie': " + containsCharlie);

        // Iterate over the elements in the ArrayList

        System.out.println("Elements in ArrayList:");

        for (String name : names) {

            System.out.println(name);

        }

    }

}

16.user Defined Exception

class InvalidAgeException extends Exception {

    public InvalidAgeException(String message) {

        super(message);

    }

}

class Voter {

    public void vote(int age) throws InvalidAgeException {

        if (age < 18) {

            throw new InvalidAgeException("You must be at least 18 years old to vote.");

        } else {

            System.out.println("You are eligible to vote.");

        }

    }

}

public class UserDefinedExceptionDemo {

    public static void main(String[] args) {

        Voter voter = new Voter();

        try {

            voter.vote(15);

        } catch (InvalidAgeException e) {

            System.out.println("Exception occurred: " + e.getMessage());

        }

    }

}

17.Border Layout

import java.awt.BorderLayout;

import javax.swing.JButton;

import javax.swing.JFrame;

public class BorderLayoutDemo {

    public static void main(String[] args) {

        JFrame frame = new JFrame("Border Layout Demo");

        frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

        JButton button1 = new JButton("North");

        JButton button2 = new JButton("South");

        JButton button3 = new JButton("East");

        JButton button4 = new JButton("West");

        JButton button5 = new JButton("Center");

        frame.add(button1, BorderLayout.NORTH);

        frame.add(button2, BorderLayout.SOUTH);

        frame.add(button3, BorderLayout.EAST);

        frame.add(button4, BorderLayout.WEST);

        frame.add(button5, BorderLayout.CENTER);

        frame.pack();

        frame.setVisible(true);

    }

}

18.Userdefined package

19.Simple Calculator

import java.util.Scanner;

public class SimpleCalculator {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter the first number: ");

        double num1 = scanner.nextDouble();

        System.out.print("Enter the second number: ");

        double num2 = scanner.nextDouble();

        System.out.print("Enter an operator (+, -, \*, /): ");

        char operator = scanner.next().charAt(0);

        double result;

        switch (operator) {

            case '+':

                result = num1 + num2;

                System.out.println("The sum is: " + result);

                break;

            case '-':

                result = num1 - num2;

                System.out.println("The difference is: " + result);

                break;

            case '\*':

                result = num1 \* num2;

                System.out.println("The product is: " + result);

                break;

            case '/':

                result = num1 / num2;

                System.out.println("The division result is: " + result);

                break;

            default:

                System.out.println("Invalid operator!");

        }

    }

}

20.Vector

import java.util.Vector;

public class VectorDemo {

    public static void main(String[] args) {

        Vector<String> names = new Vector<>();

        // Adding elements to the Vector

        names.add("Alice");

        names.add("Bob");

        names.add("Charlie");

        // Accessing elements in the Vector

        System.out.println("Elements in the Vector:");

        for (String name : names) {

            System.out.println(name);

        }

        // Checking if an element exists in the Vector

        boolean containsBob = names.contains("Bob");

        System.out.println("Contains 'Bob': " + containsBob);

        // Removing an element from the Vector

        names.remove("Charlie");

        // Getting the size of the Vector

        int size = names.size();

        System.out.println("Size of the Vector: " + size);

    }

}