**1.**

**1.1) Program to remove all repeated elements from an array**

//Program to remove all repeated elements from an array  
import java.util.\***;**public class prblm1 {  
 public static <**T**> ArrayList<**T**> removeDuplicates(ArrayList<**T**> list) {  
 ArrayList<**T**> newList = new ArrayList<**T**>()**;** for (**T** element : list) {  
 if (!newList.contains(element)) {  
 newList.add(element)**;** }  
 }  
 return newList**;** }  
  
 public static void main(String args[]) {  
 Scanner scanner = new Scanner(System.*in*)**;** ArrayList<Integer> list = new ArrayList<>()**;** System.*out*.print("Enter the number of elements in the array: ")**;** int length = scanner.nextInt()**;** System.*out*.println("Enter the elements of the array:")**;** for (int i = **0;** i < length**;** i++) {  
 int element = scanner.nextInt()**;** list.add(element)**;** }  
 scanner.close()**;** System.*out*.println("ArrayList with duplicates: " + list)**;** ArrayList<Integer> newList = *removeDuplicates*(list)**;** System.*out*.println("ArrayList with duplicates removed: " + newList)**;** }  
}

**1.2) Write a Java program to find the common elements between two arrays of integers.**

//Write a Java program to find the common elements between two arrays of integers.  
import java.util.\***;**public class prblm2 {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*)**;** System.*out*.print("Enter the number of elements in the first array: ")**;** int length1 = scanner.nextInt()**;** int[] array1 = new int[length1]**;** System.*out*.println("Enter the elements of the first array:")**;** for (int i = **0;** i < length1**;** i++) {  
 array1[i] = scanner.nextInt()**;** }  
  
 System.*out*.print("Enter the number of elements in the second array: ")**;** int length2 = scanner.nextInt()**;** int[] array2 = new int[length2]**;** System.*out*.println("Enter the elements of the second array:")**;** for (int i = **0;** i < length2**;** i++) {  
 array2[i] = scanner.nextInt()**;** }  
  
 scanner.close()**;** List<Integer> commonElements = new ArrayList<>()**;** for (int num : array1) {  
 if (Arrays.*binarySearch*(array2**,** num) >= **0**) {  
 commonElements.add(num)**;** }  
 }  
  
 System.*out*.println("Common Elements: " + commonElements)**;** }  
}

**2.**

**2. 1) Java Program to Count Number of Duplicate Words in String**

import java.util.Scanner**;**public class StringDuplicateWords {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*)**;** System.*out*.print("Enter a string: ")**;** String string = scanner.nextLine()**;** int count**;** // Converts the string into lowercase  
 string = string.toLowerCase()**;** // Split the string into words using built-in function  
 String words[] = string.split(" ")**;** System.*out*.println("Duplicate words in the given string: ")**;** for (int i = **0;** i < words.length**;** i++) {  
 count = **1;** for (int j = i + **1;** j < words.length**;** j++) {  
 if (words[i].equals(words[j])) {  
 count++**;** // Set words[j] to "0" to avoid printing visited word  
 words[j] = "0"**;** }  
 }  
  
 // Displays the duplicate word if count is greater than 1  
 if (count > **1** && !words[i].equals("0"))  
 System.*out*.println(words[i])**;** }  
  
 scanner.close()**;** }  
}

**2.2) How to Check if the String Contains 'e' in umbrella**

//How to Check if the String Contains 'e' in umbrella  
  
public class containsE {  
 public static void main(String[] args) {  
   
 String str = "umbrella"**;** // Check if the string contains 'e'  
 boolean containsE = str.contains("e")**;** if (containsE) {  
 System.*out*.println("The string contains 'e'.")**;** } else {  
 System.*out*.println("The string does not contain 'e'.")**;** }  
 }  
}

**3.**

**3.1)Java Program to Reverse a String.**

import java.util.Scanner**;**public class StringReverse {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*)**;** System.*out*.print("Enter a string: ")**;** String input = scanner.nextLine()**;** String reversed = *reverseString*(input)**;** System.*out*.println("Reversed string: " + reversed)**;** scanner.close()**;** }  
  
 public static String reverseString(String str) {  
 StringBuilder reversed = new StringBuilder()**;** for (int i = str.length() - **1;** i >= **0;** i--) {  
 reversed.append(str.charAt(i))**;** }  
 return reversed.toString()**;** }  
}

**3.2) Write a Java program to check that String is palindrome or not.**

import java.util.Scanner**;**public class PalindromeString{  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*)**;** System.*out*.print("Enter a string: ")**;** String input = scanner.nextLine()**;** boolean isPalindrome = *checkPalindrome*(input)**;** if (isPalindrome) {  
 System.*out*.println("The string is a palindrome.")**;** } else {  
 System.*out*.println("The string is not a palindrome.")**;** }  
  
  
 scanner.close()**;** }  
  
 public static boolean checkPalindrome(String str) {  
 int left = **0;** int right = str.length() - **1;** while (left < right) {  
 if (str.charAt(left) != str.charAt(right)) {  
 return false**;** // Characters don't match, not a palindrome  
 }  
 left++**;** right--**;** }  
  
 return true**;** // All characters matched, it is a palindrome  
 }  
}

**4. A Company manufactures Vehicles, which could be a Helicopter, a Car, or a Train depending on the customer’s demand. Each Vehicle instance has a method called move, which prints on the console the nature of movement of the vehicle. For example, the Helicopter Flies in Air, the Car Drives on Road and the Train Runs on Track. Write a program that accepts input from the user on the kind of vehicle the user wants to order, and the system should print out nature of movement. Implement all Java coding best practices to implement this program.**

//A Company manufactures Vehicles, which could be a Helicopter, a Car, or a Train depending on the customer’s demand. Each Vehicle instance has a method called move, which prints on the console the nature of movement of the vehicle. For example, the Helicopter Flies in Air, the Car Drives on Road and the Train Runs on Track. Write a program that accepts input from the user on the kind of vehicle the user wants to order, and the system should print out nature of movement. Implement all Java coding best practices to implement this program.  
  
import java.util.Scanner**;**// Vehicle class (parent class)  
abstract class Vehicle {  
 public abstract void move()**;**}  
  
// Helicopter class (subclass of Vehicle)  
class Helicopter extends Vehicle {  
 @Override  
 public void move() {  
 System.*out*.println("The Helicopter flies in the air.")**;** }  
}  
  
// Car class (subclass of Vehicle)  
class Car extends Vehicle {  
 @Override  
 public void move() {  
 System.*out*.println("The Car drives on the road.")**;** }  
}  
  
// Train class (subclass of Vehicle)  
class Train extends Vehicle {  
 @Override  
 public void move() {  
 System.*out*.println("The Train runs on the track.")**;** }  
}  
  
public class VehicleType {  
 public static void main(String[] args) {  
 // Accept input from the user  
 Scanner scanner = new Scanner(System.*in*)**;** System.*out*.print("Enter the type of vehicle (Helicopter/Car/Train): ")**;** String vehicleType = scanner.nextLine()**;** scanner.close()**;** // Create the corresponding vehicle object based on user input  
 Vehicle vehicle**;** switch (vehicleType.toLowerCase()) {  
 case "helicopter":  
 vehicle = new Helicopter()**;** break**;** case "car":  
 vehicle = new Car()**;** break**;** case "train":  
 vehicle = new Train()**;** break**;** default:  
 System.*out*.println("Invalid vehicle type.")**;** return**;** // Exit the program if an invalid vehicle type is entered  
 }  
  
 // Print the nature of movement of the selected vehicle  
 vehicle.move()**;** }  
}

import java.util.Scanner;

// Define the Vehicle interface

interface Vehicle {

void move();

}

// Implement the Helicopter class

class Helicopter implements Vehicle {

@Override

public void move() {

System.out.println("The Helicopter flies in the air.");

}

}

// Implement the Car class

class Car implements Vehicle {

@Override

public void move() {

System.out.println("The Car drives on the road.");

}

}

// Implement the Train class

class Train implements Vehicle {

@Override

public void move() {

System.out.println("The Train runs on the track.");

}

}

// Main class

class VehicleFactory {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Welcome to the Vehicle Factory!");

while (true) {

System.out.println("What kind of vehicle would you like to order?");

System.out.println("1. Helicopter");

System.out.println("2. Car");

System.out.println("3. Train");

System.out.println("4. Exit");

int choice = scanner.nextInt();

if (choice == 4) {

System.out.println("Thank you for using the Vehicle Factory. Goodbye!");

break;

}

Vehicle vehicle = createVehicle(choice);

if (vehicle != null) {

vehicle.move();

} else {

System.out.println("Invalid choice. Please try again.");

}

}

scanner.close();

}

// Create the appropriate vehicle based on user choice

private static Vehicle createVehicle(int choice) {

switch (choice) {

case 1:

return new Helicopter();

case 2:

return new Car();

case 3:

return new Train();

default:

return null;

}

}

}

**5. We have to calculate the percentage of marks obtained in three subjects (each out of 100) by student A and in four subjects (each out of 100) by student B. Create an abstract class 'Marks' with an abstract method 'getPercentage'. It is inherited by two other classes 'A' and 'B' each having a method with the same name which returns the percentage of the students. The constructor of student A takes the marks in three subjects as its parameters and the marks in four subjects as its parameters for student B. Create an object for each of the two classes and print the percentage of marks for both the students.**

// We have to calculate the percentage of marks obtained in three subjects  
// (each out of 100) by student A and in four subjects (each out of 100) by student B.  
// Create an abstract class 'Marks' with an abstract method 'getPercentage'.  
// It is inherited by two other classes 'A' and 'B' each having a method with the  
// same name which returns the percentage of the students. The constructor of student A  
// takes the marks in three subjects as its parameters and the marks in four subjects as  
// its parameters for student B. Create an object for each of the two classes and print  
// the percentage of marks for both the students.  
  
import java.util.\***;**abstract class Marks {  
 public abstract double getPercentage()**;**}  
  
class A extends Marks {  
  
 private double subject1Marks**;** private double subject2Marks**;** private double subject3Marks**;** public A(double subject1Marks**,** double subject2Marks**,** double subject3Marks) {  
 this.subject1Marks = subject1Marks**;** this.subject2Marks = subject2Marks**;** this.subject3Marks = subject3Marks**;** }  
  
 @Override  
 public double getPercentage() {  
 double totalMarks = subject1Marks + subject2Marks + subject3Marks**;** return (totalMarks / **300**) \* **100;** }  
}  
  
class B extends Marks {  
 private double subject1Marks**;** private double subject2Marks**;** private double subject3Marks**;** private double subject4Marks**;** public B(double subject1Marks**,** double subject2Marks**,** double subject3Marks**,** double subject4Marks) {  
 this.subject1Marks = subject1Marks**;** this.subject2Marks = subject2Marks**;** this.subject3Marks = subject3Marks**;** this.subject4Marks = subject4Marks**;** }  
  
 @Override  
 public double getPercentage() {  
 double totalMarks = subject1Marks + subject2Marks + subject3Marks + subject4Marks**;** return (totalMarks / **400**) \* **100;** }  
}  
  
public class CalculatePercentage {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*)**;** System.*out*.println("Enter Marks for A: ")**;** int a = scanner.nextInt()**;** int b = scanner.nextInt()**;** int c = scanner.nextInt()**;** System.*out*.println("Enter Marks for B :")**;** int d = scanner.nextInt()**;** int e = scanner.nextInt()**;** int f = scanner.nextInt()**;** int g = scanner.nextInt()**;** A studentA = new A(a**,** b**,** c)**;** B studentB = new B(d**,** e**,** f**,** g)**;** System.*out*.println("Percentage of marks for Student A: " + studentA.getPercentage())**;** System.*out*.println("Percentage of marks for Student B: " + studentB.getPercentage())**;** }  
}

**6. Write the following code in your editor below:  
A class named Arithmetic with a method named add that takes 2 integers as parameters and returns an integer denoting their sum.  
A class named Adder that inherits from a superclass named Arithmetic. The main method in the Tester class should print the following: SAMPLE O/P:My superclass is: Arithmetic  
42 13 20**

//A class named Arithmetic with a method named add that takes 2 integers as parameters and returns an integer denoting their sum.

// A

// class named

// Adder that

// inherits from

// a superclass

// named Arithmetic.

// The main

// method in

// the Tester

// class should

// print the following:

// SAMPLE O/P:

// My superclass is:Arithmetic 42 13 20

class Arithmetic {

    public int add(int a, int b) {

        return a + b;

    }

}

class Adder extends Arithmetic {

}

public class ClassSuperclass {

    public static void main(String[] args) {

        Adder adder = new Adder();

        // Print superclass name

        System.out.println("My superclass is: " + adder.getClass().getSuperclass().getName());

        // Perform addition using the add() method

        int sum = adder.add(42, 13);

        System.out.println("42 + 13 = " + sum);

        // Perform addition using the add() method from the superclass

        int sum2 = adder.add(20, 0);

        System.out.println("20 + 0 = " + sum2);

    }

}

**7. You are required to compute the power of a number by implementing a calculator. Create a class My Calculator which consists of a single method long power (int, int). This method takes two integers n and p, as parameters and finds (n)p. If either or is negative, then the method must throw an exception which says " n or p should not be negative”. Also, if both and are zero, then the method must throw an exception which says "n or p should not be negative”.**

class MyCalculator {

    public long power(int n, int p) throws Exception {

        if (n < 0 || p < 0) {

            throw new Exception("n or p should not be negative");

        } else if (n == 0 && p == 0) {

            throw new Exception("n and p should not be zero");

        } else {

            return (long) Math.pow(n, p);

        }

    }

}

public class CalculatorException {

    public static void main(String[] args) {

        MyCalculator calculator = new MyCalculator();

        try {

            long result1 = calculator.power(2, 3);

            System.out.println("Result 1: " + result1); // 2^3 = 8

            long result2 = calculator.power(0, 0);

            System.out.println("Result 2: " + result2);

        } catch (Exception e) {

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

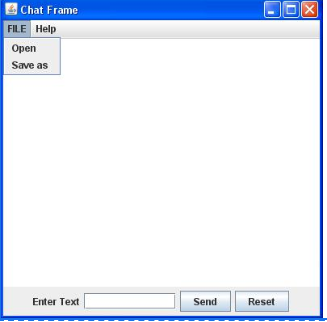
        }

    }

}

**8. You are given a phone book that consists of people's names and their phone number. After that you will be given some person's name as query. For each query, print the phone number of that person. Use HashMap to implement it.The first line will have an integer denoting the number of entries in the phone book. Each entry consists of two lines: a name and the corresponding phone number.  
After these, there will be some queries. Each query will contain a person's name. Read the queries until end-of-file.  
Constraints:  
A person's name consists of only lower-case English letters and it may be in the format 'first-name last-name' or in the format 'first-name'. Each phone number has exactly 8 digits without any leading zeros.For each case, print "Not found" if the person has no entry in the phone book. Otherwise, print the person's name and phone number.**

import java.util.\***;**public class map {  
  
 public static void main(String args[])  
 {  
 Scanner sc = new Scanner(System.*in*)**;** System.*out*.println("enter nubmer of entries : ")**;** int totalEntries = sc.nextInt()**;** sc.nextLine()**;** //// Consume the remaining newline character after reading the integer input  
  
 HashMap <String **,** Integer> map = new HashMap <String **,** Integer>()**;** for(int i=**0 ;** i<totalEntries **;** i++){  
  
 System.*out*.println("enter name of "+ i+**1** + " person : ")**;** String name = sc.nextLine()**;** System.*out*.println("enter phone number of "+ i+**1** + " person : ")**;** int phoneno = sc.nextInt()**;** sc.nextLine()**;** map.put(name **,** phoneno)**;** }  
  
 System.*out*.println("Enter the queries (press Ctrl+D or Ctrl+Z to exit):")**;** while(sc.hasNextLine())  
 {  
 String query = sc.nextLine()**;** if(map.containsKey(query))  
 {  
 int phoneNo = map.get(query)**;** System.*out*.println("Phone number of the query you fired is "+ phoneNo)**;** }else{  
 System.*out*.println("Not Found \n")**;** }  
 }  
  
 sc.close()**;** }  
}

9. 

import javax.swing.\***;**import java.awt.\***;**class ChatFrame {  
 public static void main(String args[]) {  
  
 // Creating the Frame  
 JFrame frame = new JFrame("Chat Frame")**;** frame.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*)**;** frame.setSize(**400, 400**)**;** // Creating the MenuBar and adding components  
 JMenuBar mb = new JMenuBar()**;** JMenu m1 = new JMenu("File")**;** JMenu m2 = new JMenu("Help")**;** mb.add(m1)**;** mb.add(m2)**;** JMenuItem m11 = new JMenuItem("Open")**;** JMenuItem m22 = new JMenuItem("Save as")**;** m1.add(m11)**;** m1.add(m22)**;** // Creating the panel at bottom and adding components  
 JPanel panel = new JPanel()**;** // the panel is not visible in output  
 JLabel label = new JLabel("Enter Text")**;** JTextField tf = new JTextField(**10**)**;** // accepts upto 10 characters  
 JButton send = new JButton("Send")**;** JButton reset = new JButton("Reset")**;** panel.add(label)**;** // Components Added using Flow Layout  
 panel.add(tf)**;** panel.add(send)**;** panel.add(reset)**;** // Text Area at the Center  
 JTextArea ta = new JTextArea()**;** // Adding Components to the frame.  
 frame.getContentPane().add(BorderLayout.*SOUTH***,** panel)**;** frame.getContentPane().add(BorderLayout.*NORTH***,** mb)**;** frame.getContentPane().add(BorderLayout.*CENTER***,** ta)**;** frame.setVisible(true)**;** }  
}

10. 

import java.awt.EventQueue**;**import javax.swing.\***;**import java.awt.\***;**public class NumberFrame {  
 private JFrame frame**;** */\*\*  
 \* Launch the application.  
 \*/* public static void main(String[] args) {  
 EventQueue.*invokeLater*(new Runnable() {  
 public void run() {  
 try {  
 NumberFrame window = new NumberFrame()**;** window.frame.setVisible(true)**;** } catch (Exception e) {  
 e.printStackTrace()**;** }  
 }  
 })**;** }  
 */\*\*  
 \* Create the application.  
 \*/* public NumberFrame() {  
 initialize()**;** }  
 */\*\*  
 \* Initialize the contents of the frame.  
 \*/* private void initialize() {  
 frame = new JFrame()**;** frame.setBounds(**100, 100, 450, 300**)**;** frame.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*)**;** frame.getContentPane().setLayout(null)**;** JLabel label1=new JLabel("Number Addition")**;** label1.setForeground(Color.*BLUE*)**;** label1.setBounds(**20, 10, 100, 30**)**;** frame.getContentPane().add(label1)**;** //for number 1  
 JLabel first=new JLabel("First Number: ")**;** first.setBounds(**10, 40, 120, 20**)**;** frame.getContentPane().add(first)**;** JTextField f1=new JTextField()**;** f1.setBounds(**140, 40, 140, 20**)**;** frame.getContentPane().add(f1)**;** //for number 2  
 JLabel second=new JLabel("Second Number:")**;** second.setBounds(**10, 65, 120, 20**)**;** frame.getContentPane().add(second)**;** JTextField s1=new JTextField()**;** s1.setBounds(**140, 65, 140, 20**)**;** frame.getContentPane().add(s1)**;** //for result  
 JLabel result=new JLabel("Result:")**;** result.setBounds(**10, 90, 120, 20**)**;** frame.getContentPane().add(result)**;** JTextField r1=new JTextField()**;** r1.setBounds(**140, 90, 140, 20**)**;** frame.getContentPane().add(r1)**;** //add button  
 JButton add=new JButton("ADD")**;** add.setBounds(**90,120, 80,20**)**;** frame.getContentPane().add(add)**;** //clear button  
 JButton clear=new JButton("CLEAR")**;** clear.setBounds(**200,120, 80,20**)**;** frame.getContentPane().add(clear)**;** //exit button  
 JButton exit=new JButton("Exit")**;** exit.setBounds(**250, 200, 80, 20**)**;** frame.add(exit)**;** }  
}

**11.**

**11.1) Write a Java program that takes a number as input and prints its multiplication table up to 10. Test Data:  
Input a number: 8  
Expected Output :  
8 x 1 = 8  
8 x 2 = 16  
8 x 3 = 24  
...  
8 x 10 = 80**

import java.util.Scanner**;**public class Multiplication  
{  
 public static void main(String args[])  
 {  
 Scanner sc = new Scanner(System.*in*)**;** System.*out*.print("Enter number: ")**;**//reading a number whose table is to be print  
 int num=sc.nextInt()**;**//loop start execution form and execute until the condition i<=10 becomes false  
 for(int i=**1;** i <= **10;** i++)  
 {  
//prints table of the entered number  
 System.*out*.println(num+" \* "+i+" = "+num\*i)**;** }  
 }  
}

**11.2) Write a java program to check that given number is prime or not.**

//Write a java program to check that given number is prime or not.

import java.util.Scanner;

public class PrimeNumber {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        System.out.println("Enter a number to check if it is truly prime number or not: ");

        int number = sc.nextInt();

        if (isPrime(number)) {

            System.out.println(number + " is prime number");

        } else {

            System.out.println(number + " is a non-prime number");

        }

    }

    static boolean isPrime(int num) {

        if (num <= 1) {

            return false;

        }

        for (int i = 2; i <= num / 2; i++) {

            if ((num % i) == 0)

                return false;

        }

        return true;

    }

}

**12. Write a Java program to display the pattern like a diamond.  
Input number of rows (half of the diamond) :7 Expected Output :  
  
  
\*   
\*\*\*   
\*\*\*\*\*   
\*\*\*\*\*\*\*   
\*\*\*\*\*\*\*\*\*   
\*\*\*\*\*\*\*\*\*\*\*   
\*\*\*\*\*\*\*\*\*\*\*\*\*   
\*\*\*\*\*\*\*\*\*\*\*   
\*\*\*\*\*\*\*\*\*   
\*\*\*\*\*\*\*   
\*\*\*\*\*   
\*\*\*   
\***

import java.util.Scanner;

public class DiamondPattern {

    public static void main(String args[]) {

        Scanner sc = new Scanner(System.in);

        System.out.println("Enter a number: ");

        int n = sc.nextInt();

        n = n \* 2;

        int a, b;

        for (a = 0; a < n; a++) {

            for (b = 0; b <= a; b++) {

                System.out.print("\* ");

            }

            System.out.println();

        }

        for (a = n-1; a > 0; a--) {

            for (b = a-1; b >= 0; b--) {

                System.out.print("\* ");

            }

            System.out.println();

        }

    }

}

**13.**

**13.1) Write Java Program to find the transpose of a given matrix .**

public class MatrixTranspose {

    public static void main(String args[]) {

        // creating a matrix

        int original[][] = { { 1, 3, 4 }, { 2, 4, 3 }, { 3, 4, 5 } };

        // creating another matrix to store transpose of a matrix

        int transpose[][] = new int[3][3]; // 3 rows and 3 columns

        // Code to transpose a matrix

        for (int i = 0; i < 3; i++) {

            for (int j = 0; j < 3; j++) {

                transpose[i][j] = original[j][i];

            }

        }

        System.out.println("Printing Matrix without transpose:");

        for (int i = 0; i < 3; i++) {

            for (int j = 0; j < 3; j++) {

                System.out.print(original[i][j] + " ");

            }

            System.out.println();// new line

        }

        System.out.println("Printing Matrix After Transpose:");

        for (int i = 0; i < 3; i++) {

            for (int j = 0; j < 3; j++) {

                System.out.print(transpose[i][j] + " ");

            }

            System.out.println();// new line

        }

    }

}

**13.2) Write Java Program to find the number of the words in the given text file.**

//Write Java Program to find the number of the words in the given text file.

import java.io.File;

import java.io.FileNotFoundException;

import java.util.Scanner;

public class WordCount {

    public static void main(String[] args) {

        try {

            // Specify the path to the text file

            String filePath = "new.txt";

            // Create a File object

            File file = new File(filePath);

            // Create a Scanner to read the file

            Scanner scanner = new Scanner(file);

            int wordCount = 0;

            // Iterate through each word in the file

            while (scanner.hasNext()) {

                scanner.next();

                wordCount++;

            }

            // Close the Scanner

            scanner.close();

            System.out.println("Number of words: " + wordCount);

        } catch (FileNotFoundException e) {

            System.out.println("File not found: " + e.getMessage());

        }

    }

}

14. 

import java.awt.BorderLayout**;**import java.awt.GridLayout**;**import javax.swing.\***;**public class SwingCalculator extends JFrame {  
 JTextField tf**;** JButton b1**,** b2**,** b3**,** b4**,** b5**,** b6**,** b7**,** b8**,** b9**,** b10**,** b11**,** b12**,** b13**,** b14**,** b15**,** b16**;** JPanel p1**;** public SwingCalculator() {  
 tf = new JTextField()**;** p1 = new JPanel()**;** b1 = new JButton("1")**;** b2 = new JButton("2")**;** b3 = new JButton("3")**;** b4 = new JButton("4")**;** b5 = new JButton("5")**;** b6 = new JButton("6")**;** b7 = new JButton("7")**;** b8 = new JButton("8")**;** b9 = new JButton("9")**;** b10 = new JButton("0")**;** b11 = new JButton("C")**;** b12 = new JButton("=")**;** b13 = new JButton("+")**;** b14 = new JButton("-")**;** b15 = new JButton("\*")**;** b16 = new JButton("/")**;** p1.setLayout(new GridLayout(**4, 4**))**;** p1.add(b7)**;** p1.add(b8)**;** p1.add(b9)**;** p1.add(b13)**;** p1.add(b4)**;** p1.add(b5)**;** p1.add(b6)**;** p1.add(b14)**;** p1.add(b1)**;** p1.add(b2)**;** p1.add(b3)**;** p1.add(b15)**;** p1.add(b11)**;** p1.add(b10)**;** p1.add(b12)**;** p1.add(b16)**;** add(tf**,** BorderLayout.*NORTH*)**;** add(p1**,** BorderLayout.*CENTER*)**;** // setLayout(new BorderLayout());  
 setVisible(true)**;** setSize(**300, 300**)**;** setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*)**;** }  
  
 public static void main(String[] args) {  
 new SwingCalculator()**;** }  
}

workin cal

// Java program to create a simple calculator  
// with basic +, -, /, \* using java swing elements  
  
import java.awt.event.\***;**import javax.swing.\***;**import java.awt.\***;**class cal extends JFrame implements ActionListener {  
 // create a frame  
 static JFrame *f***;** // create a textfield  
 static JTextField *l***;** // store operator and operands  
 String s0**,** s1**,** s2**;** // default constructor  
 cal()  
 {  
 s0 = s1 = s2 = ""**;** }  
  
 // main function  
 public static void main(String args[])  
 {  
 // create a frame  
 *f* = new JFrame("calculator")**;** try {  
 // set look and feel  
 UIManager.*setLookAndFeel*(UIManager.*getSystemLookAndFeelClassName*())**;** }  
 catch (Exception e) {  
 System.*err*.println(e.getMessage())**;** }  
  
 // create a object of class  
 cal c = new cal()**;** // create a textfield  
 *l* = new JTextField(**16**)**;** // set the textfield to non editable  
 *l*.setEditable(false)**;** // create number buttons and some operators  
 JButton b0**,** b1**,** b2**,** b3**,** b4**,** b5**,** b6**,** b7**,** b8**,** b9**,** ba**,** bs**,** bd**,** bm**,** be**,** beq**,** beq1**;** // create number buttons  
 b0 = new JButton("0")**;** b1 = new JButton("1")**;** b2 = new JButton("2")**;** b3 = new JButton("3")**;** b4 = new JButton("4")**;** b5 = new JButton("5")**;** b6 = new JButton("6")**;** b7 = new JButton("7")**;** b8 = new JButton("8")**;** b9 = new JButton("9")**;** // equals button  
 beq1 = new JButton("=")**;** // create operator buttons  
 ba = new JButton("+")**;** bs = new JButton("-")**;** bd = new JButton("/")**;** bm = new JButton("\*")**;** beq = new JButton("C")**;** // create . button  
 be = new JButton(".")**;** // create a panel  
 JPanel p = new JPanel()**;** // add action listeners  
 bm.addActionListener(c)**;** bd.addActionListener(c)**;** bs.addActionListener(c)**;** ba.addActionListener(c)**;** b9.addActionListener(c)**;** b8.addActionListener(c)**;** b7.addActionListener(c)**;** b6.addActionListener(c)**;** b5.addActionListener(c)**;** b4.addActionListener(c)**;** b3.addActionListener(c)**;** b2.addActionListener(c)**;** b1.addActionListener(c)**;** b0.addActionListener(c)**;** be.addActionListener(c)**;** beq.addActionListener(c)**;** beq1.addActionListener(c)**;** // add elements to panel  
 p.add(*l*)**;** p.add(ba)**;** p.add(b1)**;** p.add(b2)**;** p.add(b3)**;** p.add(bs)**;** p.add(b4)**;** p.add(b5)**;** p.add(b6)**;** p.add(bm)**;** p.add(b7)**;** p.add(b8)**;** p.add(b9)**;** p.add(bd)**;** p.add(be)**;** p.add(b0)**;** p.add(beq)**;** p.add(beq1)**;** // set Background of panel  
 p.setBackground(Color.*blue*)**;** // add panel to frame  
 *f*.add(p)**;** *f*.setSize(**200, 220**)**;** *f*.show()**;** }  
  
 public void actionPerformed(ActionEvent e)  
 {  
 String s = e.getActionCommand()**;** // if the value is a number  
 if ((s.charAt(**0**) >= '0' && s.charAt(**0**) <= '9') || s.charAt(**0**) == '.') {  
 // if operand is present then add to second no  
 if (!s1.equals(""))  
 s2 = s2 + s**;** else  
 s0 = s0 + s**;** // set the value of text  
 *l*.setText(s0 + s1 + s2)**;** }  
 else if (s.charAt(**0**) == 'C') {  
 // clear the one letter  
 s0 = s1 = s2 = ""**;** // set the value of text  
 *l*.setText(s0 + s1 + s2)**;** }  
 else if (s.charAt(**0**) == '=') {  
  
 double te**;** // store the value in 1st  
 if (s1.equals("+"))  
 te = (Double.*parseDouble*(s0) + Double.*parseDouble*(s2))**;** else if (s1.equals("-"))  
 te = (Double.*parseDouble*(s0) - Double.*parseDouble*(s2))**;** else if (s1.equals("/"))  
 te = (Double.*parseDouble*(s0) / Double.*parseDouble*(s2))**;** else  
 te = (Double.*parseDouble*(s0) \* Double.*parseDouble*(s2))**;** // set the value of text  
 *l*.setText(s0 + s1 + s2 + "=" + te)**;** // convert it to string  
 s0 = Double.*toString*(te)**;** s1 = s2 = ""**;** }  
 else {  
 // if there was no operand  
 if (s1.equals("") || s2.equals(""))  
 s1 = s**;** // else evaluate  
 else {  
 double te**;** // store the value in 1st  
 if (s1.equals("+"))  
 te = (Double.*parseDouble*(s0) + Double.*parseDouble*(s2))**;** else if (s1.equals("-"))  
 te = (Double.*parseDouble*(s0) - Double.*parseDouble*(s2))**;** else if (s1.equals("/"))  
 te = (Double.*parseDouble*(s0) / Double.*parseDouble*(s2))**;** else  
 te = (Double.*parseDouble*(s0) \* Double.*parseDouble*(s2))**;** // convert it to string  
 s0 = Double.*toString*(te)**;** // place the operator  
 s1 = s**;** // make the operand blank  
 s2 = ""**;** }  
  
 // set the value of text  
 *l*.setText(s0 + s1 + s2)**;** }  
 }  
}

**15. Write a Java Program to iterate ArrayList using for-loop, iterator, and advance for-loop. Insert 3 Array List. Input 20 30 40 Output:  
  
  
iterator Loop:  
20  
30  
40  
Advanced For Loop:  
20  
30  
40  
For Loop:  
20  
30  
40**

import java.util.ArrayList;

import java.util.Iterator;

public class ArrayListIteration {

    public static void main(String[] args) {

        // Create an ArrayList and add elements

        ArrayList<Integer> arrayList = new ArrayList<>();

        arrayList.add(20);

        arrayList.add(30);

        arrayList.add(40);

        System.out.println("iterator Loop:");

        // Iterate using Iterator

        Iterator<Integer> iterator = arrayList.iterator();

        while (iterator.hasNext()) {

            System.out.println(iterator.next());

        }

        System.out.println("Advanced For Loop:");

        // Iterate using Advanced For Loop

        for (int num : arrayList) {

            System.out.println(num);

        }

        System.out.println("For Loop:");

        // Iterate using regular for loop

        for (int i = 0; i < arrayList.size(); i++) {

            System.out.println(arrayList.get(i));

        }

    }

}

**16. Write a Java Program to count the number of words in a string using HashMap.Output:  
Input :Enter String: "This this is is done by Saket Saket";  
{Saket=2, by=1, this=1, This=1, is=2, done=1}**

import java.util.HashMap;

public class StringWordCountHashmap {

    public static void main(String[] args) {

        String str = "This this is is done by Saket Saket";

        String[] split = str.split(" ");

        HashMap<String, Integer> map = new HashMap<String, Integer>();

        for (int i = 0; i < split.length - 1; i++) {

            if (map.containsKey(split[i])) {

                int count = map.get(split[i]);

                map.put(split[i], count + 1);

            }

            else {

                map.put(split[i], 1);

            }

        }

        System.out.println(map);

    }

}

**17. Write a program to read 10 string from console and then print the sorted strings on console (Use String Class). 2) combine two string 3)reverse first string and display it .**

import java.util.Arrays;

import java.util.Scanner;

public class StringOperations {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        System.out.println("Enter 10 strings: ");

        String arr[] = new String[10];

        for (int i = 0; i < 10; i++) {

            String str = sc.next();

            arr[i] = str;

        }

        Arrays.sort(arr);

        System.out.println("\nSorted Array: ");

        for (int i = 0; i < 10; i++) {

            System.out.println(arr[i]);

        }

        System.out.println("\nCombination of 2 strings: ");

        String newstr = arr[0] + " " + arr[1];

        System.out.println(newstr);

        System.out.println("\nReverse of first element: ");

        StringBuilder strrev = new StringBuilder(arr[0]);

        strrev.reverse();

        System.out.println(strrev);

    }

}

**18. Write a program to implement following inheritance. Accept data for 5 persons and display the name of employee having salary greater than 5000.  
  
Class Name: Person  
Member variables:  
Name, age  
  
Class Name: Employee  
Member variables:  
Designation, salary**

//Write a program to implement following inheritance. Accept data for 5 persons and display the name of employee having salary greater than 5000.

// Class Name: Person

// Member variables:

// Name, age

// Class Name: Employee

// Member variables:

// Designation, salary

import java.util.Scanner;

class Person {

    private String name;

    private int age;

    public Person(String name, int age) {

        this.name = name;

        this.age = age;

    }

    public String getName() {

        return name;

    }

    public int getAge() {

        return age;

    }

}

class Employee extends Person {

    private String designation;

    private double salary;

    public Employee(String name, int age, String designation, double salary) {

        super(name, age);

        this.designation = designation;

        this.salary = salary;

    }

    public String getDesignation() {

        return designation;

    }

    public double getSalary() {

        return salary;

    }

}

public class EmployeeSalary {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        Employee[] employees = new Employee[5];

        // Accept data for 5 persons

        for (int i = 0; i < 5; i++) {

            System.out.println("Enter details for Employee " + (i + 1));

            System.out.print("Name: ");

            String name = scanner.nextLine();

            System.out.print("Age: ");

            int age = scanner.nextInt();

            scanner.nextLine();

            System.out.print("Designation: ");

            String designation = scanner.nextLine();

            System.out.print("Salary: ");

            double salary = scanner.nextDouble();

            scanner.nextLine();

            employees[i] = new Employee(name, age, designation, salary);

        }

        // Display the name of employees with salary greater than 5000

        System.out.println("Employees with salary greater than 5000:");

        for (Employee employee : employees) {

            if (employee.getSalary() > 5000) {

                System.out.println(employee.getName());

            }

        }

    }

}

**19. Implementing “Multiple Inheritance”. Create a two interfaces Account containing methods set() and display() And interface Person containing methods store() and disp(). Derive a class Customer from Person and Account. Accept the name, account number, balance and display all the information related to account along with the interest.**

interface Account {

    void set(String accountNumber, double balance);

    void display();

}

interface Person {

    void store(String name);

    void disp();

}

class Customer implements Account, Person {

    private String name;

    private String accountNumber;

    private double balance;

    public void set(String accountNumber, double balance) {

        this.accountNumber = accountNumber;

        this.balance = balance;

    }

    public void display() {

        System.out.println("Account Number: " + accountNumber);

        System.out.println("Balance: " + balance);

    }

    public void store(String name) {

        this.name = name;

    }

    public void disp() {

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

    }

    public void calculateInterest() {

        double interest = balance \* 0.05; // Assuming interest rate of 5%

        System.out.println("Interest: " + interest);

    }

}

public class InterestCalci {

    public static void main(String[] args) {

        Customer customer = new Customer();

        customer.store("John Doe");

        customer.set("123456789", 10000.0);

        customer.disp();

        customer.display();

        customer.calculateInterest();

    }

}

**20. "Write a program, to implement the following hierarchy. Displays information of each class the rectangle represents the classes. The classes Movie and MusicVideo inherits all the members of the class VideoTape.**  
"



class VideoTape{

    String title;

    int length;

    boolean avaialbe;

    void set(String title , int length , boolean avaialbe)

    {

        this.title = title;

        this.length = length;

        this.avaialbe = avaialbe;

    }

    void show()

    {

       System.out.println("Tittle is : "+ this.title);

       System.out.println("Length is : "+ this.length);

       System.out.println("Availbe is : "+ this.avaialbe);

    }

}

class Movie extends VideoTape{

    String Director;

    int rating;

    void set(String Director , int rating)

    {

        this.Director = Director;

        this.rating = rating;

    }

    void show()

    {

        // super.show();

         System.out.println("Director is : "+ this.Director);

         System.out.println("rating is : "+ this.rating);

    }

}

class MusicVideo extends VideoTape{

    String artist , Category;

    void set(String artist , String Category)

    {

        this.artist = artist;

        this.Category = Category;

    }

    void show()

    {

       // super.show();

        System.out.println("Artist is : "+ this.artist);

        System.out.println("Category is : "+ this.Category);

    }

}

public class hirarchy {

    public static void main(String args[])

    {

       VideoTape videoTape = new VideoTape();

       videoTape.set("Hello World", 5, true);

       videoTape.show();

       Movie movie = new Movie();

       movie.set("Ajay", 5);

       movie.show();

       MusicVideo musicVideo = new MusicVideo();

       musicVideo.set("Atul", "Comedy");

       musicVideo.show();

    }

}

**21. Write a Java program to create a class called "Student" with a name, grade, and courses attributes, and methods to add and remove courses.**

import java.util.ArrayList;

import java.util.List;

public class StudentCourses {

    private String name;

    private int grade;

    private List<String> courses;

    public StudentCourses(String name, int grade) {

        this.name = name;

        this.grade = grade;

        this.courses = new ArrayList<>();

    }

    public String getName() {

        return name;

    }

    public int getGrade() {

        return grade;

    }

    public void addCourse(String course) {

        courses.add(course);

    }

    public void removeCourse(String course) {

        courses.remove(course);

    }

    public List<String> getCourses() {

        return courses;

    }

    public static void main(String[] args) {

        // Creating a student object

        StudentCourses student = new StudentCourses("John Doe", 10);

        // Adding courses

        student.addCourse("Math");

        student.addCourse("Science");

        student.addCourse("English");

        // Removing a course

        student.removeCourse("Science");

        // Getting the student's name, grade, and courses

        System.out.println("Student Name: " + student.getName());

        System.out.println("Grade: " + student.getGrade());

        System.out.println("Courses: " + student.getCourses());

    }

}

**22. Write a Java program to create a class known as Person with methods called getFirstName() and getLastName(). Create a subclass called Employee that adds a new method named getEmployeeId() and overrides the getLastName() method to include the employee's job title.**

// Write a Java program to create a class known as Person with methods called getFirstName() and getLastName().

// Create a subclass called Employee that adds a new method named getEmployeeId() and overrides the getLastName()

// method to include the employee's job title.

class Person {

    private String firstName;

    private String lastName;

    public Person(String firstName, String lastName) {

        this.firstName = firstName;

        this.lastName = lastName;

    }

    public String getFirstName() {

        return firstName;

    }

    public String getLastName() {

        return lastName;

    }

}

class Employee extends Person {

    private int employeeId;

    private String jobTitle;

    public Employee(String firstName, String lastName, int employeeId, String jobTitle) {

        super(firstName, lastName);

        this.employeeId = employeeId;

        this.jobTitle = jobTitle;

    }

    public int getEmployeeId() {

        return employeeId;

    }

    public String getLastName() {

        return super.getLastName() + ", " + jobTitle;

    }

}

public class EmployeeName {

    public static void main(String[] args) {

        Person person = new Person("John", "Doe");

        System.out.println("Person: " + person.getFirstName() + " " + person.getLastName());

        Employee employee = new Employee("Jane", "Smith", 12345, "Manager");

        System.out.println("Employee: " + employee.getFirstName() + " " + employee.getLastName());

        System.out.println("Employee ID: " + employee.getEmployeeId());

    }

}

**23. Write a Java program to find the length of the longest consecutive elements sequence from an unsorted array of integers.  
Sample array: [49, 1, 3, 200, 2, 4, 70, 5]  
The longest consecutive elements sequence is [1, 2, 3, 4, 5], therefore the program will return its length 5.**

import java.util.HashSet;

public class ConsecutiveNumbers {

    public static int longestConsecutive(int[] nums) {

        HashSet<Integer> numSet = new HashSet<>();

        int maxLength = 0;

        for (int num : nums) {

            numSet.add(num);

        }

        for (int num : nums) {

            if (!numSet.contains(num - 1)) {

                int currentNum = num;

                int currentLength = 1;

                while (numSet.contains(currentNum + 1)) {

                    currentNum++;

                    currentLength++;

                }

                maxLength = Math.max(maxLength, currentLength);

            }

        }

        return maxLength;

    }

    public static void main(String[] args) {

        int[] nums = { 49, 1, 3, 200, 2, 4, 70, 5 };

        int length = longestConsecutive(nums);

        System.out.println("The length of the longest consecutive elements sequence is: " + length);

    }

}

**24. Create a class Student with attributes roll no, name, age and course. Initialize values through parameterized constructor. If age of student is not in between 15 and 21 then generate user-defined exception "AgeNotWithinRangeException". If name contains numbers or special symbols raise exception "NameNotValidException". Define the two exception classes.**

import java.io.\***;**import java.util.\***;**class AgeNotWithInRangeException extends Exception  
{  
 public String validage()  
 {  
 return ("Age is not between 15 and 21 … Please ReEnter the Age")**;** }  
}  
class NameNotValidException extends Exception  
{  
 public String validname()  
 {  
 return("Name is not Valid … Please ReEnter the Name")**;** }  
}  
class Student  
{  
 int roll**,**age**;** String name**,**course**;** Student()  
 {  
 roll=**0;** name=null**;** age=**0;** course=null**;** }  
 Student(int r**,**String n**,**int a**,**String c)  
 {  
 roll=r**;** course=c**;** int l**,**temp=**0;** l=n.length()**;** for(int i=**0;**i<l**;**i++)  
 {  
 char ch**;** ch=n.charAt(i)**;** if(ch<'A' || ch>'Z' && ch<'a' || ch>'z')  
 temp=**1;** }  
 /\*———-Checking Name——————–\*/  
 try  
 {  
 if(temp==**1**)  
 throw new NameNotValidException()**;** else  
 name=n**;** }  
 catch(NameNotValidException e2)  
 {  
 System.*out*.println(e2)**;** }  
 /\*———-Checking Age——————–\*/  
 try  
 {  
 if(a>=**15** && a<=**21**)  
 age=a**;** else  
 throw new AgeNotWithInRangeException()**;** }  
 catch(AgeNotWithInRangeException e1)  
 {  
 System.*out*.println(e1)**;** }  
 }  
 void display()  
 {  
 System.*out*.println("roll Name Age Course")**;** System.*out*.println("————————————————-")**;** System.*out*.println(roll+" "+name+" "+age+" "+course)**;** }  
}  
class AgeNameException  
{  
 public static void main(String args[])throws IOException  
 {  
 Scanner sc = new Scanner(System.*in*)**;** int r**,**a**;** String n**,**c**;** System.*out*.println("Enter roll,name,age,course")**;** r=sc.nextInt()**;** sc.nextLine()**;** n=sc.nextLine()**;** a=sc.nextInt()**;** sc.nextLine()**;** c=sc.nextLine()**;** Student s=new Student(r**,**n**,**a**,**c)**;** s.display()**;** }  
}

25. 

import java.awt.\***;**class demo extends Frame {  
 Label lbl1**,** lbl2**,** lbl3**,** lbl4**,** lbl5**;** public demo() {  
 lbl1 = new Label("All The Best")**;** lbl1.setForeground(Color.*red*)**;** add(lbl1)**;** lbl2 = new Label("All The Best")**;** lbl2.setForeground(Color.*magenta*)**;** add(lbl2)**;** lbl3 = new Label("All The Best")**;** lbl3.setForeground(Color.*blue*)**;** add(lbl3)**;** lbl4 = new Label("All The Best")**;** lbl4.setForeground(Color.*green*)**;** add(lbl4)**;** lbl5 = new Label("All The Best")**;** lbl5.setForeground(Color.*cyan*)**;** add(lbl5)**;** setVisible(true)**;** setSize(**400, 300**)**;** setLayout(new FlowLayout())**;** setBackground(Color.*gray*)**;** }  
  
 public void paint(Graphics g) {  
 g.setColor(Color.*magenta*)**;** g.drawString("All The Best"**, 100, 100**)**;** g.setColor(Color.*cyan*)**;** g.drawString("All The Best"**, 150, 150**)**;** g.setColor(Color.*red*)**;** g.drawString("All The Best"**, 200, 200**)**;** g.setColor(Color.*black*)**;** g.drawString("All The Best"**, 250, 250**)**;** }  
  
 public static void main(String[] args) {  
 new demo()**;** }  
}