1.

1.1) Program to remove all repeated elements from an array

import java.util.\*;

public class RemoveDuplicatesFromArray {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

// Get the size of the array from the user

System.out.print("Enter the size of the array: ");

int size = input.nextInt();

int[] arr = new int[size];

// Get array elements from the user

System.out.println("Enter the elements of the array:");

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

arr[i] = input.nextInt();

}

input.close();

// Remove duplicates from the array

int[] result = removeDuplicates(arr);

// Print the array without duplicates

System.out.println("Array without duplicates:");

for (int i = 0; i < result.length; i++) {

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

}

}

public static int[] removeDuplicates(int[] arr) {

// Create a HashSet to store unique elements

Set<Integer> uniqueSet = new HashSet<>();

// Iterate over the array and add elements to the HashSet

for (int i = 0; i < arr.length; i++) {

uniqueSet.add(arr[i]);

}

// Convert the HashSet back to an array

int[] result = new int[uniqueSet.size()];

int index = 0;

for (int element : uniqueSet) {

result[index++] = element;

}

return result;

}

}

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

import java.util.\*;

public class CommonElements {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

// Get the size of the first array from the user

System.out.print("Enter the size of the first array: ");

int size1 = input.nextInt();

int[] arr1 = new int[size1];

// Get elements of the first array from the user

System.out.println("Enter the elements of the first array:");

for (int i = 0; i < size1; i++) {

arr1[i] = input.nextInt();

}

// Get the size of the second array from the user

System.out.print("Enter the size of the second array: ");

int size2 = input.nextInt();

int[] arr2 = new int[size2];

// Get elements of the second array from the user

System.out.println("Enter the elements of the second array:");

for (int i = 0; i < size2; i++) {

arr2[i] = input.nextInt();

}

input.close();

// Find common elements between the two arrays

int[] commonElements = findCommonElements(arr1, arr2);

// Print the common elements

System.out.println("Common elements between the two arrays:");

for (int element : commonElements) {

System.out.print(element + " ");

}

}

public static int[] findCommonElements(int[] arr1, int[] arr2) {

// Create a HashSet to store unique elements from the first array

Set<Integer> uniqueElements = new HashSet<>();

for (int element : arr1) {

uniqueElements.add(element);

}

// Create a list to store common elements

List<Integer> commonElementsList = new ArrayList<>();

for (int element : arr2) {

if (uniqueElements.contains(element)) {

commonElementsList.add(element);

}

}

// Convert the list to an array

int[] commonElementsArray = new int[commonElementsList.size()];

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

commonElementsArray[i] = commonElementsList.get(i);

}

return commonElementsArray;

}

}

2.

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

import java.util.\*;

public class CountDuplicateWords {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

// Get the input string from the user

System.out.print("Enter a string: ");

String str = input.nextLine();

input.close();

// Count the number of duplicate words

int count = countDuplicateWords(str);

// Print the result

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

}

public static int countDuplicateWords(String str) {

// Convert the string to lowercase and split into words

String[] words = str.toLowerCase().split("\\s+");

// Create a HashMap to store word frequencies

Map<String, Integer> wordFreq = new HashMap<>();

// Count the frequency of each word

for (String word : words) {

wordFreq.put(word, wordFreq.getOrDefault(word, 0) + 1);

}

// Count the number of duplicate words

int count = 0;

for (int freq : wordFreq.values()) {

if (freq > 1) {

count++;

}

}

return count;

}

}

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

public class StringContainsExample {

public static void main(String[] args) {

String str = "umbrella";

// Check if the string contains 'e'

boolean containsE = str.contains("e");

// Print the result

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.\*;

public class ReverseString {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

// Get the input string from the user

System.out.print("Enter a string: ");

String str = input.nextLine();

input.close();

// Reverse the string

String reversedStr = reverseString(str);

// Print the reversed string

System.out.println("Reversed string: " + reversedStr);

}

public static String reverseString(String str) {

// Convert the string to a character array

char[] charArray = str.toCharArray();

// Reverse the character array

int left = 0;

int right = charArray.length - 1;

while (left < right) {

// Swap characters at left and right indices

char temp = charArray[left];

charArray[left] = charArray[right];

charArray[right] = temp;

// Move the left and right indices towards the center

left++;

right--;

}

// Convert the character array back to a string

String reversedStr = new String(charArray);

return reversedStr;

}

}

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

import java.util.\*;

class PalindromeChecker {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

// Get the input string from the user

System.out.print("Enter a string: ");

String str = input.nextLine();

input.close();

// Check if the string is a palindrome

boolean isPalindrome = isPalindrome(str);

// Print the result

if (isPalindrome) {

System.out.println("The string is a palindrome.");

} else {

System.out.println("The string is not a palindrome.");

}

}

public static boolean isPalindrome(String str) {

// Remove non-alphanumeric characters and convert to lowercase

String cleanedStr = str.replaceAll("[^a-zA-Z0-9]", "").toLowerCase();

// Check if the cleaned string is a palindrome

int left = 0;

int right = cleanedStr.length() - 1;

while (left < right) {

if (cleanedStr.charAt(left) != cleanedStr.charAt(right)) {

return false;

}

left++;

right--;

}

return true;

}

}

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.

import java.util.Scanner;

abstract class Vehicle {

abstract void move();

}

class Helicopter extends Vehicle {

@Override

void move() {

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

}

}

class Car extends Vehicle {

@Override

void move() {

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

}

}

class Train extends Vehicle {

@Override

void move() {

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

}

}

class VehicleFactory {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

// Get the input from the user

System.out.print("Enter the type of vehicle (Helicopter/Car/Train): ");

String vehicleType = input.nextLine();

input.close();

// Create the corresponding vehicle object based on user input

Vehicle vehicle;

if (vehicleType.equalsIgnoreCase("Helicopter")) {

vehicle = new Helicopter();

} else if (vehicleType.equalsIgnoreCase("Car")) {

vehicle = new Car();

} else if (vehicleType.equalsIgnoreCase("Train")) {

vehicle = new Train();

} else {

System.out.println("Invalid vehicle type entered.");

return;

}

// Call the move method of the selected vehicle

vehicle.move();

}

}

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 eac of the two classes and print the percentage of marks for both the students.

import java.util.Scanner;

abstract class Marks {

abstract double getPercentage();

}

class A extends Marks {

private int subject1;

private int subject2;

private int subject3;

public A(int subject1, int subject2, int subject3) {

this.subject1 = subject1;

this.subject2 = subject2;

this.subject3 = subject3;

}

@Override

double getPercentage() {

double totalMarks = subject1 + subject2 + subject3;

return (totalMarks / 300) \* 100;

}

}

class B extends Marks {

private int subject1;

private int subject2;

private int subject3;

private int subject4;

public B(int subject1, int subject2, int subject3, int subject4) {

this.subject1 = subject1;

this.subject2 = subject2;

this.subject3 = subject3;

this.subject4 = subject4;

}

@Override

double getPercentage() {

double totalMarks = subject1 + subject2 + subject3 + subject4;

return (totalMarks / 400) \* 100;

}

}

class MarksPercentage {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

// Get marks for student A

System.out.print("Enter marks for student A (Subject 1): ");

int marksA1 = input.nextInt();

System.out.print("Enter marks for student A (Subject 2): ");

int marksA2 = input.nextInt();

System.out.print("Enter marks for student A (Subject 3): ");

int marksA3 = input.nextInt();

// Create object for student A

A studentA = new A(marksA1, marksA2, marksA3);

// Get marks for student B

System.out.print("Enter marks for student B (Subject 1): ");

int marksB1 = input.nextInt();

System.out.print("Enter marks for student B (Subject 2): ");

int marksB2 = input.nextInt();

System.out.print("Enter marks for student B (Subject 3): ");

int marksB3 = input.nextInt();

System.out.print("Enter marks for student B (Subject 4): ");

int marksB4 = input.nextInt();

// Create object for student B

B studentB = new B(marksB1, marksB2, marksB3, marksB4);

input.close();

// Print the percentage of marks for both students

System.out.println("Percentage of student A: " + studentA.getPercentage() + "%");

System.out.println("Percentage of 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**

class Arithmetic {

public int add(int a, int b) {

return a + b;

}

}

class Adder extends Arithmetic {

// No additional methods or fields needed in this case

}

class Tester {

public static void main(String[] args) {

Adder adder = new Adder();

// Print the superclass name

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

// Perform addition using the inherited add() method

int num1 = 42;

int num2 = 13;

int sum = adder.add(num1, num2);

// Print the numbers and their sum

System.out.println(num1 + " " + num2 + " " + sum);

}

}

**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”.

import java.util.Scanner;

class MyCalculator {

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);

}

}

}

class CalculatorTester {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

MyCalculator calculator = new MyCalculator();

try {

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

int n = input.nextInt();

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

int p = input.nextInt();

long result = calculator.power(n, p);

System.out.println(n + " raised to the power of " + p + " is: " + result);

} catch (Exception e) {

System.out.println(e.getMessage());

} finally {

input.close();

}

}

}

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.HashMap;

import java.util.Scanner;

class PhoneBook {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

// Create a HashMap to store the phone book entries

HashMap<String, String> phoneBook = new HashMap<>();

// Read the number of entries in the phone book

int numEntries = input.nextInt();

input.nextLine(); // Consume the newline character after reading the integer

// Read the phone book entries and populate the HashMap

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

String name = input.nextLine();

String phoneNumber = input.nextLine();

phoneBook.put(name, phoneNumber);

}

// Read the queries until the end-of-file

while (input.hasNextLine()) {

String query = input.nextLine();

// Check if the query is present in the phone book

if (phoneBook.containsKey(query)) {

String phoneNumber = phoneBook.get(query);

System.out.println(query + " " + phoneNumber);

} else {

System.out.println("Not found");

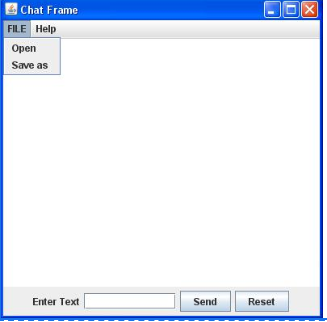
}

}

input.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 javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class AdditionGUI {

public static void main(String args[]) {

Abc obj = new Abc();

}

}

class Abc implements ActionListener {

JLabel l1;

JTextField t1;

JLabel l2;

JTextField t2;

JButton b1, b2, b3;

JLabel l3;

JTextField t3;

public Abc() {

JFrame j = new JFrame("Number Addition");

l1 = new JLabel("First Number:");

t1 = new JTextField(20);

l2 = new JLabel("Second Number:");

t2 = new JTextField(20);

l3 = new JLabel("Result");

t3 = new JTextField(20);

b1 = new JButton("Add");

b2 = new JButton("Clear");

b3 = new JButton("Exit");

j.add(l1);

j.add(t1);

j.add(l2);

j.add(t2);

j.add(l3);

j.add(t3);

j.add(b1);

j.add(b2);

j.add(b3);

l1.setBounds(100, 50, 180, 50);

t1.setBounds(250, 50, 180, 50);

l2.setBounds(100, 150, 180, 50);

t2.setBounds(250, 150, 180, 50);

l3.setBounds(100, 250, 180, 50);

t3.setBounds(250, 250, 180, 50);

b1.setBounds(150, 350, 180, 50);

b2.setBounds(300, 350, 180, 50);

b3.setBounds(400, 450, 180, 50);

b1.addActionListener(this);

b2.addActionListener(this);

b3.addActionListener(this);

j.setLayout(null);

j.setVisible(true);

j.setSize(600, 600);

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

}

public void actionPerformed(ActionEvent ae) {

if (ae.getSource() == b1) {

int num1 = Integer.parseInt(t1.getText());

int num2 = Integer.parseInt(t2.getText());

int value = num1 + num2;

t3.setText(" " + value);

}

if (ae.getSource() == b2) {

t1.setText("");

t2.setText("");

t3.setText("");

}

if (ae.getSource() == b3) {

System.exit(0);

}

}

}

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;

class MultiplicationTable {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

System.out.print("Input a number: ");

int number = input.nextInt();

for (int i = 1; i <= 10; i++) {

int result = number \* i;

System.out.println(number + " x " + i + " = " + result);

}

input.close();

}

}

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

import java.util.Scanner;

class PrimeNumberChecker {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

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

int number = input.nextInt();

boolean isPrime = isPrimeNumber(number);

if (isPrime) {

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

} else {

System.out.println(number + " is not a prime number.");

}

input.close();

}

private static boolean isPrimeNumber(int number) {

if (number <= 1) {

return false;

}

for (int i = 2; i <= Math.sqrt(number); i++) {

if (number % 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;

class HalfDiamondPattern {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

System.out.print("Enter the number of rows (half of the diamond): ");

int rows = input.nextInt();

// Upper half of the diamond

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

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

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

}

System.out.println();

}

// Lower half of the diamond

for (int i = rows - 1; i >= 1; i--) {

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

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

}

System.out.println();

}

input.close();

}

}

13.

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

import java.util.Scanner;

class MatrixTranspose {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

System.out.print("Enter the number of rows in the matrix: ");

int rows = input.nextInt();

System.out.print("Enter the number of columns in the matrix: ");

int columns = input.nextInt();

int[][] matrix = new int[rows][columns];

System.out.println("Enter the elements of the matrix:");

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

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

matrix[i][j] = input.nextInt();

}

}

int[][] transpose = getTranspose(matrix);

System.out.println("The transpose of the matrix is:");

displayMatrix(transpose);

input.close();

}

private static int[][] getTranspose(int[][] matrix) {

int rows = matrix.length;

int columns = matrix[0].length;

int[][] transpose = new int[columns][rows];

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

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

transpose[j][i] = matrix[i][j];

}

}

return transpose;

}

private static void displayMatrix(int[][] matrix) {

int rows = matrix.length;

int columns = matrix[0].length;

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

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

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

}

System.out.println();

}

}

}

13.2) 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;

class WordCount {

public static void main(String[] args) {

try {

// Specify the file path

String filePath = "C:\\Users\\Sarika\\OneDrive\\Documents\\homepage.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;

// Read each word in the file

while (scanner.hasNext()) {

scanner.next();

wordCount++;

} // Close the scanner

scanner.close();

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

} catch (FileNotFoundException e) {

System.out.println("File not found.");

}

}

}

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();

}

}

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

import java.util.ArrayList;

import java.util.Iterator;

class ArrayListIteration {

public static void main(String[] args) {

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

numbers.add(20);

numbers.add(30);

numbers.add(40);

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

iterateWithIterator(numbers);

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

iterateWithAdvancedForLoop(numbers);

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

iterateWithForLoop(numbers);

}

private static void iterateWithIterator(ArrayList<Integer> numbers) {

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

while (iterator.hasNext()) {

int number = iterator.next();

System.out.println(number);

}

}

private static void iterateWithAdvancedForLoop(ArrayList<Integer> numbers) {

for (int number : numbers) {

System.out.println(number);

}

}

private static void iterateWithForLoop(ArrayList<Integer> numbers) {

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

int number = numbers.get(i);

System.out.println(number);

}

}

}

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;

import java.util.Scanner;

class WordCount {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

System.out.print("Enter a string: ");

String str = input.nextLine();

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

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

for (String word : words) {

wordCountMap.put(word, wordCountMap.getOrDefault(word, 0) + 1);

}

System.out.println(wordCountMap);

input.close();

}

}

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 nd dispaly it .

import java.util.Arrays;

import java.util.Scanner;

public class StringOperations {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

// Read 10 strings from the console

String[] strings = new String[10];

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

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

strings[i] = input.nextLine();

}

// Sort the strings

Arrays.sort(strings);

// Print the sorted strings

System.out.println("Sorted strings:");

for (String str : strings) {

System.out.println(str);

}

// Combine two strings

String combined = strings[0] + strings[1];

System.out.println("Combined string: " + combined);

// Reverse the first string

String reversed = reverseString(strings[0]);

System.out.println("Reversed first string: " + reversed);

input.close();

}

private 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();

}

}

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

import java.util.Scanner;

class Person {

String name;

int age;

public Person(String name, int age) {

this.name = name;

this.age = age;

}

}

class Employee extends Person {

String designation;

double salary;

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

super(name, age);

this.designation = designation;

this.salary = salary;

}

}

class InheritanceExample {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

Employee[] employees = new Employee[5];

for (int i = 0; i < 5; i++) {

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

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

String name = input.nextLine();

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

int age = input.nextInt();

input.nextLine(); // Consume the remaining newline character

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

String designation = input.nextLine();

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

double salary = input.nextDouble();

input.nextLine(); // Consume the remaining newline character

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

}

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

for (Employee employee : employees) {

if (employee.salary > 5000) {

System.out.println(employee.name);

}

}

input.close();

}

}

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.

import java.util.Scanner;

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 a 5% interest rate

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

}

}

class MultipleInheritanceExample {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

Customer customer = new Customer();

System.out.print("Enter name: ");

String name = input.nextLine();

customer.store(name);

System.out.print("Enter account number: ");

String accountNumber = input.nextLine();

System.out.print("Enter balance: ");

double balance = input.nextDouble();

input.nextLine(); // Consume the remaining newline character

customer.set(accountNumber, balance);

System.out.println("\nAccount Information:");

customer.disp();

customer.display();

customer.calculateInterest();

input.close();

}

}

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.  
"



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;

import java.util.Scanner;

class Student {

private String name;

private String grade;

private List<String> courses;

public Student(String name, String grade) {

this.name = name;

this.grade = grade;

this.courses = new ArrayList<>();

}

public void addCourse(String course) {

courses.add(course);

}

public void removeCourse(String course) {

courses.remove(course);

}

public void displayCourses() {

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

for (String course : courses) {

System.out.println(course);

}

}

public void displayInfo() {

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

System.out.println("Grade: " + grade);

displayCourses();

}

}

class StudentManagement {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

System.out.print("Enter student name: ");

String name = input.nextLine();

System.out.print("Enter student grade: ");

String grade = input.nextLine();

Student student = new Student(name, grade);

System.out.print("Enter number of courses to add: ");

int numCourses = input.nextInt();

input.nextLine(); // Consume the remaining newline character

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

System.out.print("Enter course " + (i + 1) + ": ");

String course = input.nextLine();

student.addCourse(course);

}

System.out.print("Enter number of courses to remove: ");

int numCoursesToRemove = input.nextInt();

input.nextLine(); // Consume the remaining newline character

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

System.out.print("Enter course to remove " + (i + 1) + ": ");

String course = input.nextLine();

student.removeCourse(course);

}

System.out.println("\nStudent Information:");

student.displayInfo();

input.close();

}

}

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.

import java.util.Scanner;

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 String employeeId;

private String jobTitle;

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

super(firstName, lastName);

this.employeeId = employeeId;

this.jobTitle = jobTitle;

}

public String getEmployeeId() {

return employeeId;

}

@Override

public String getLastName() {

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

}

}

class PersonEmployeeTest {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

System.out.print("Enter employee's first name: ");

String firstName = input.nextLine();

System.out.print("Enter employee's last name: ");

String lastName = input.nextLine();

System.out.print("Enter employee's ID: ");

String employeeId = input.nextLine();

System.out.print("Enter employee's job title: ");

String jobTitle = input.nextLine();

Employee employee = new Employee(firstName, lastName, employeeId, jobTitle);

System.out.println("\nEmployee Information:");

System.out.println("First Name: " + employee.getFirstName());

System.out.println("Last Name: " + employee.getLastName());

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

input.close();

}

}

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;

import java.util.Scanner;

class LongestConsecutiveSequence {

public static int longestConsecutive(int[] nums) {

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

// Add all elements to the set

for (int num : nums) {

set.add(num);

}

int maxLength = 0;

// Iterate over each element in the array

for (int num : nums) {

// Check if the current number is the start of a sequence

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

int currentNum = num;

int currentLength = 1;

// Find the length of the consecutive sequence

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

currentNum++;

currentLength++;

}

// Update the maximum length if necessary

maxLength = Math.max(maxLength, currentLength);

}

}

return maxLength;

}

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

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

int n = input.nextInt();

int[] nums = new int[n];

System.out.print("Enter the elements: ");

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

nums[i] = input.nextInt();

}

int length = longestConsecutive(nums);

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

input.close();

}

}

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.util.Scanner;

class AgeNotWithinRangeException extends Exception {

public AgeNotWithinRangeException(String message) {

super(message);

}

}

class NameNotValidException extends Exception {

public NameNotValidException(String message) {

super(message);

}

}

class Student {

private int rollNo;

private String name;

private int age;

private String course;

public Student(int rollNo, String name, int age, String course) throws AgeNotWithinRangeException, NameNotValidException {

if (age < 15 || age > 21) {

throw new AgeNotWithinRangeException("Age is not within the valid range (15-21).");

}

if (!name.matches("^[a-zA-Z ]+$")) {

throw new NameNotValidException("Name contains numbers or special symbols.");

}

this.rollNo = rollNo;

this.name = name;

this.age = age;

this.course = course;

}

public void displayDetails() {

System.out.println("Roll No: " + rollNo);

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

System.out.println("Age: " + age);

System.out.println("Course: " + course);

}

}

class StudentTest {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

try {

System.out.print("Enter Roll No: ");

int rollNo = input.nextInt();

input.nextLine();

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

String name = input.nextLine();

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

int age = input.nextInt();

input.nextLine();

System.out.print("Enter Course: ");

String course = input.nextLine();

Student student = new Student(rollNo, name, age, course);

System.out.println("\nStudent Details:");

student.displayDetails();

} catch (AgeNotWithinRangeException e) {

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

} catch (NameNotValidException e) {

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

}

input.close();

}

}

25. 

import java.awt.\*;

class ColorGUI extends Frame {

Label lbl1, lbl2, lbl3, lbl4, lbl5;

public ColorGUI() {

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 ColorGUI();

}

}