1.

1.1) Program to remove all repeated elements from an array

import java.util.ArrayList;

import java.util.HashSet;

import java.util.List;

import java.util.Set;

public class RemoveRepeatedElements {

public static void main(String[] args) {

int[] array = {1, 2, 3, 4, 3, 2, 1, 5, 6, 7, 7, 8, 9, 9, 10};

int[] result = removeRepeatedElements(array);

System.out.println("Original Array:");

printArray(array);

System.out.println("\nArray without Repeated Elements:");

printArray(result);

}

public static int[] removeRepeatedElements(int[] array) {

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

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

for (int element : array) {

if (!uniqueElements.contains(element)) {

uniqueElements.add(element);

resultList.add(element);

}

}

int[] resultArray = new int[resultList.size()];

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

resultArray[i] = resultList.get(i);

}

return resultArray;

}

public static void printArray(int[] array) {

for (int element : array) {

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

}

System.out.println();

}

}

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

import java.util.\*;

public class Main {

public static void main(String[] args) {

int[] array1 = {1};

int[] array2 = {1, 5, 2, 1};

int[] commonElements = findCommonElements(array1, array2);

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

printArray(array1);

System.out.println("\nArray 2:");

printArray(array2);

System.out.println("\nCommon Elements:");

printArray(commonElements);

}

public static int[] findCommonElements(int[] array1, int[] array2) {

Set<Integer> set1 = new HashSet<>();

Set<Integer> set2 = new HashSet<>();

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

for (int element : array1) {

set1.add(element);

}

for (int element : array2) {

if (set1.contains(element)) {

set2.add(element);

}

}

Iterator<Integer> itr = set2.iterator();

while (itr.hasNext()) {

resultList.add(itr.next());

}

int[] resultArray = new int[resultList.size()];

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

resultArray[i] = resultList.get(i);

}

return resultArray;

}

public static void printArray(int[] array) {

for (int element : array) {

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

}

System.out.println();

}

}

2.

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

public class Main {

public static void main(String[] args) {

String str = "the quick brown fox jumps over the lazy dog the";

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

int count = 0;

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

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

if (words[i].equals(words[j])) {

count++;

System.out.println(words[j]);

break;//(here the last won’t be printed because break terminates the loop)

}

}

}

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

}

}

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

public class containselement {

public static void main(String[] args){

String s="umbrella";

if(s.contains("e")){

System.out.println("The word contains 'e'.");

}

else{

System.out.println("The word does not containt 'e'.");

}

}

}

3.

3.1) Java Program to Reverse a String.

public class ReverseString {

public static void main(String[] args) {

String str = "Hello, World!";

String reversedStr = reverseString(str);

System.out.println("Original String: " + str);

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

}

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.

public class PalindromeChecker {

public static void main(String[] args) {

String str = "madam"; // Change this string to test different cases

if (isPalindrome(str)) {

System.out.println(str + " is a palindrome.");

} else {

System.out.println(str + " is not a palindrome.");

}

}

public static boolean isPalindrome(String str) {

int i = 0;

int j = str.length() - 1;

while (i < j) {

if (str.charAt(i) != str.charAt(j)) {

return false;

}

i++;

j--;

}

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;

class Vehicle {

public void move() {

System.out.println("Vehicle moves");

}

}

class Helicopter extends Vehicle {

@Override

public void move() {

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

}

}

class Car extends Vehicle {

@Override

public void move() {

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

}

}

class Train extends Vehicle {

@Override

public void move() {

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

}

}

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Enter the type of vehicle you want to order (helicopter/car/train):");

String vehicleType = scanner.nextLine().toLowerCase();

Vehicle vehicle;

switch (vehicleType) {

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 entered.");

return;

}

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

abstract class Marks {

public abstract double getPercentage();

}

class A extends Marks {

private double subject1;

private double subject2;

private double subject3;

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

this.subject1 = subject1;

this.subject2 = subject2;

this.subject3 = subject3;

}

@Override

public double getPercentage() {

double totalMarks = subject1 + subject2 + subject3;

return (totalMarks / 300) \* 100;

}

}

class B extends Marks {

private double subject1;

private double subject2;

private double subject3;

private double subject4;

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

this.subject1 = subject1;

this.subject2 = subject2;

this.subject3 = subject3;

this.subject4 = subject4;

}

@Override

public double getPercentage() {

double totalMarks = subject1 + subject2 + subject3 + subject4;

return (totalMarks / 400) \* 100;

}

}

public class Main {

public static void main(String[] args) {

A studentA = new A(85, 90, 92);

B studentB = new B(78, 85, 90, 88);

System.out.println("Student A percentage: " + studentA.getPercentage() + "%");

System.out.println("Student B percentage: " + 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 {

// Empty implementation as Adder inherits the 'add' method from Arithmetic

}

public class Tester {

public static void main(String[] args) {

Adder adder = new Adder();

System.out.println("My superclass is: Arithmetic");

int sum = adder.add(42, 13);

System.out.println(sum);

sum = adder.add(13, 20);

System.out.println(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”.

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 Main {

public static void main(String[] args) {

MyCalculator myCalculator = new MyCalculator();

try {

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

System.out.println("Result 1: " + result1);

long result2 = myCalculator.power(-1, 5);

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

long result3 = myCalculator.power(0, 0);

System.out.println("Result 3: " + result3);

} catch (Exception e) {

System.out.println("Exception caught: " + 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.HashMap;

import java.util.Map;

import java.util.Scanner;

public class Solution {

/\* input Scanner for reading input for the program \*/

private static Scanner inputScanner;

public static void main(String[] args) {

/\* Create a phone book Map \*/

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

/\* Create the input Scanner instance \*/

inputScanner = new Scanner(System.in);

/\* Read the number of input phone numbers \*/

int numFriends = inputScanner.nextInt();

/\* Go to the next line of input \*/

inputScanner.nextLine();

/\* Loop thru the number of friends, reading the name and phone number, and adding to the phoneBook \*/

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

/\* Read the name of the friend \*/

String name = inputScanner.nextLine();

/\* Read the phone number \*/

String phone = inputScanner.nextLine();

/\* Put the name and phone number into the phoneBook \*/

phoneBook.put(name, phone);

}

/\* Loop while there is still more input data \*/

while (inputScanner.hasNext()) {

/\* Read the person to search for a phone number \*/

String inputName = inputScanner.nextLine();

/\* Does the name exist in the phone Book \*/

if (phoneBook.containsKey(inputName)) {

/\* Yes, print out the name and phone number \*/

System.out.println(inputName + "=" + phoneBook.get(inputName));

} else {

/\* Print out error message if not found \*/

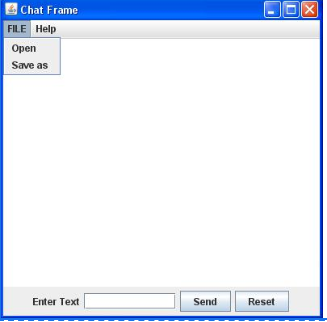
System.out.println("Not found");

}

}

}

}

9. 

import javax.swing.\*;

import java.awt.\*;

class gui {

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 NumberAddition {

private JFrame frame;

/\*\*

\* Launch the application.

\*/

public static void main(String[] args) {

EventQueue.invokeLater(new Runnable() {

public void run() {

try {

NumberAddition window = new NumberAddition();

window.frame.setVisible(true);

} catch (Exception e) {

e.printStackTrace();

}

}

});

}

/\*\*

\* Create the application.

\*/

public NumberAddition() {

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 \_11 {

public static void main(String [] args){

Scanner sc = new Scanner(System.in);

int n = sc.nextInt();

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

System.out.println(n +" \* "+i+" = "+i\*n);

}

}

}

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

import java.util.Scanner;

public class PrimeChecker {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

int number = scanner.nextInt();

if (isPrime(number)) {

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

} else {

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

}

}

public static boolean isPrime(int number) {

if (number <= 1) {

return false;

}

// Check for divisibility from 2 to square root of the number

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

if (number % i == 0) {

return false; // Number is divisible, not prime

}

}

return true; // Number is prime

}

}

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 Exercise21 {

public static void main(String[] args)

{

int i,j,r;

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

Scanner in = new Scanner(System.in);

r = in.nextInt();

for(i=0;i<=r;i++)

{

for(j=1;j<=r-i;j++)

System.out.print(" ");

for(j=1;j<=2\*i-1;j++)

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

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

}

for(i=r-1;i>=1;i--)

{

for(j=1;j<=r-i;j++)

System.out.print(" ");

for(j=1;j<=2\*i-1;j++)

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

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

}

}

}

13.

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

import java.util.Scanner;

public class MatrixTransposeExample2

{

public static void main(String args[])

{

int i, j;

System.out.println("Enter total rows and columns: ");

Scanner s = new Scanner(System.in);

int row = s.nextInt();

int column = s.nextInt();

int array[][] = new int[row][column];

System.out.println("Enter matrix:");

for(i = 0; i < row; i++)

{

for(j = 0; j < column; j++)

{

array[i][j] = s.nextInt();

System.out.print(" ");

}

}

System.out.println("The above matrix before Transpose is ");

for(i = 0; i < row; i++)

{

for(j = 0; j < column; j++)

{

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

}

System.out.println(" ");

}

System.out.println("The above matrix after Transpose is ");

for(i = 0; i < column; i++)

{

for(j = 0; j < row; j++)

{

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

}

System.out.println(" ");

}

}

}

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

import java.io.BufferedReader;

import java.io.FileReader;

public class CountWordFile

{

public static void main(String[] args) throws Exception {

String line;

int count = 0;

//Opens a file in read mode

FileReader file = new FileReader("data.txt ");

BufferedReader br = new BufferedReader(file);

//Gets each line till end of file is reached

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

//Splits each line into words

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

//Counts each word

count = count + words.length;

}

System.out.println("Number of words present in given file: " + count);

br.close();

}

}

14. 

import java.awt.\*;

import java.awt.event.\*;

import javax.swing.\*; // Using Swing's components and containers

// A Swing application extends from javax.swing.JFrame

public class SwingCalculator extends JFrame {

private JTextField tfDisplay;

private int result = 0; // the result so far

private String numberInStr = ""; // the number entered as String

private char previousOpr = ' '; // the previous operator

private char currentOpr = ' '; // the current operator

// Constructor to setup the UI components and event handlers

public SwingCalculator() {

// TODO: Setup the UI

// ......

// Number buttons listener (inner class)

class NumberBtnListener implements ActionListener {

@Override

public void actionPerformed(ActionEvent evt) {

numberInStr += evt.getActionCommand();

tfDisplay.setText(numberInStr);

}

}

// Operator buttons listener (inner class)

class OprBtnListener implements ActionListener {

@Override

public void actionPerformed(ActionEvent evt) {

previousOpr = currentOpr; // save

currentOpr = evt.getActionCommand().charAt(0);

// TODO: Processing logic

// ......

}

}

}

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;

public class ArrayListIteration {

public static void main(String[] args) {

// Create an ArrayList

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

// Insert elements into the ArrayList

list.add(20);

list.add(30);

list.add(40);

// Iterate over the ArrayList using a for-loop

System.out.println("Using for-loop:");

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

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

}

// Iterate over the ArrayList using an iterator

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

Iterator<Integer> itr = list.iterator();

while (itr.hasNext()) {

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

}

// Iterate over the ArrayList using an enhanced for-loop

System.out.println("Using enhanced for-loop:");

for (int num : list) {

System.out.println(num);

}

}

}

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.Map;

import java.util.Scanner;

public class WordCount {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

String input = scanner.nextLine();

Map<String, Integer> wordCountMap = countWords(input);

System.out.println(wordCountMap);

}

public static Map<String, Integer> countWords(String input) {

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

// Split the input string into words

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

// Count the occurrence of each word

for (String word : words) {

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

}

return wordCountMap;

}

}

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 scanner = new Scanner(System.in);

String[] strings = new String[10];

// Read 10 strings from the console

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

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

strings[i] = scanner.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 combinedString = strings[0] + strings[1];

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

// Reverse the first string

String reversedString = reverseString(strings[0]);

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

}

public static String reverseString(String str) {

StringBuilder reversed = new StringBuilder(str);

reversed.reverse();

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 {

protected String name;

protected int age;

public Person(String name, int age) {

this.name = name;

this.age = 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 boolean hasSalaryGreaterThan(double amount) {

return salary > amount;

}

}

public class InheritanceExample {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

Employee[] employees = new Employee[5];

System.out.println("Enter details of 5 persons:");

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

System.out.println("Person " + (i + 1) + ":");

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

String name = scanner.nextLine();

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

int age = scanner.nextInt();

scanner.nextLine(); // Consume the newline character

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

String designation = scanner.nextLine();

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

double salary = scanner.nextDouble();

scanner.nextLine(); // Consume the 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.hasSalaryGreaterThan(5000)) {

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

}

}

}

}

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 Person, Account {

private String name;

private String accountNumber;

private double balance;

@Override

public void set(String accountNumber, double balance) {

this.accountNumber = accountNumber;

this.balance = balance;

}

@Override

public void display() {

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

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

System.out.println("Interest: $" + (balance \* 0.05));

}

@Override

public void store(String name) {

this.name = name;

}

@Override

public void disp() {

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

}

}

public class MultipleInheritanceExample {

public static void main(String[] args) {

Customer customer = new Customer();

customer.store("John Doe");

customer.set("123456", 5000.0);

customer.disp();

customer.display();

}

}

20. class VideoTape {

protected String title;

protected int duration;

public VideoTape(String title, int duration) {

this.title = title;

this.duration = duration;

}

public void play() {

System.out.println("Playing " + title);

}

public void stop() {

System.out.println("Stopped " + title);

}

}

class Movie extends VideoTape {

private String director;

public Movie(String title, int duration, String director) {

super(title, duration);

this.director = director;

}

public void displayInfo() {

System.out.println("Title: " + title);

System.out.println("Duration: " + duration + " minutes");

System.out.println("Director: " + director);

}

}

class MusicVideo extends VideoTape {

private String artist;

public MusicVideo(String title, int duration, String artist) {

super(title, duration);

this.artist = artist;

}

public void displayInfo() {

System.out.println("Title: " + title);

System.out.println("Duration: " + duration + " minutes");

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

}

}

public class ClassHierarchyExample {

public static void main(String[] args) {

Movie movie = new Movie("Inception", 148, "Christopher Nolan");

movie.displayInfo();

movie.play();

System.out.println();

MusicVideo musicVideo = new MusicVideo("Shape of You", 234, "Ed Sheeran");

musicVideo.displayInfo();

musicVideo.play();

}

}  
"



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 Student {

private String name;

private int grade;

private List<String> courses;

public Student(String name, int 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 String getName() {

return name;

}

public int getGrade() {

return grade;

}

public List<String> getCourses() {

return courses;

}

public static void main(String[] args) {

// Create a Student object

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

// Add courses

student.addCourse("Math");

student.addCourse("Science");

student.addCourse("English");

// Print student information

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

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

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

// Remove a course

student.removeCourse("Science");

// Print updated courses

System.out.println("Updated 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.

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 + ")";

}

}

public class PersonAndEmployeeExample {

public static void main(String[] args) {

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

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

Employee employee = new Employee("Jane", "Smith", "E1234", "Software Engineer");

System.out.println("Employee Name: " + 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 LongestConsecutiveSequence {

public static int longestConsecutive(int[] nums) {

// Create a hash set to store all the numbers in the array

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

for (int num : nums) {

set.add(num);

}

int longestSequence = 0;

// Iterate through each number 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 currentSequence = 1;

// Increment the current number to find the consecutive elements

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

currentNum++;

currentSequence++;

}

// Update the longest sequence if necessary

longestSequence = Math.max(longestSequence, currentSequence);

}

}

return longestSequence;

}

public static void main(String[] args) {

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

int longestSequence = longestConsecutive(nums);

System.out.println("Length of the longest consecutive elements sequence: " + longestSequence);

}

}

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.

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 {

this.rollNo = rollNo;

// Validate and set the name

if (!isValidName(name)) {

throw new NameNotValidException("Invalid name. Name should not contain numbers or special symbols.");

}

this.name = name;

// Validate and set the age

if (!isValidAge(age)) {

throw new AgeNotWithinRangeException("Invalid age. Age should be between 15 and 21.");

}

this.age = age;

this.course = course;

}

private boolean isValidName(String name) {

// Check if the name contains any numbers or special symbols

return name.matches("[a-zA-Z\\s]+");

}

private boolean isValidAge(int age) {

// Check if the age is within the range of 15 to 21 (inclusive)

return age >= 15 && age <= 21;

}

public int getRollNo() {

return rollNo;

}

public String getName() {

return name;

}

public int getAge() {

return age;

}

public String getCourse() {

return course;

}

}

public class StudentExample {

public static void main(String[] args) {

try {

// Create a student object with valid data

Student student1 = new Student(1, "John Doe", 18, "Computer Science");

System.out.println("Student 1: " + student1.getName() + " (Roll No: " + student1.getRollNo() + ", Age: " + student1.getAge() + ", Course: " + student1.getCourse() + ")");

// Create a student object with an invalid name

Student student2 = new Student(2, "Jane123", 20, "Mathematics");

System.out.println("Student 2: " + student2.getName() + " (Roll No: " + student2.getRollNo() + ", Age: " + student2.getAge() + ", Course: " + student2.getCourse() + ")");

} catch (NameNotValidException | AgeNotWithinRangeException e) {

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

}

}

}

25. 

import java.awt.\*;

import java.awt.event.\*;

import javax.swing.\*;

class Color\_Demo extends Frame

{

Label lbl1,lbl2,lbl3,lbl4,lbl5;

public Color\_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 Color\_Demo();

}

}