**MODULE------6**

**• W.A.J.P to Take three numbers from the user and print the greatest number.**

**------------------------------------------------**

**import java.util.Scanner;**

**public class GreatestNumber**

**{**

**public static void main(String[] args)**

**{**

**Scanner scanner = new Scanner(System.in);**

**// Taking three numbers from the user**

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

**int num1 = scanner.nextInt();**

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

**int num2 = scanner.nextInt();**

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

**int num3 = scanner.nextInt();**

**// Finding the greatest number**

**int greatest;**

**if (num1 >= num2 && num1 >= num3)**

**{**

**greatest = num1;**

**}**

**else if (num2 >= num1 && num2 >= num3)**

**{**

**greatest = num2;**

**}**

**else**

**{**

**greatest = num3;**

**}**

**// Printing the greatest number**

**System.out.println("The greatest number is: " + greatest);**

**scanner.close();**

**}**

**}**

**• W.A.J.P in Java to display the first 10 natural numbers using while loop.**

**-----------------------------------------------------------------------------**

**public class First10NaturalNumbers**

**{**

**public static void main(String[] args)**

**{**

**int number = 1; // Start from the first natural number**

**System.out.println("The first 10 natural numbers are:");**

**while (number <= 10)**

**{**

**System.out.println(number);**

**number++; // Increment the number by 1**

**}**

**}**

**}**

**• W.A.J.P to find factorial for Given Number.**

**-----------------------------------------------------------**

**import java.util.Scanner;**

**public class Factorial {**

**public static void main(String[] args) {**

**Scanner scanner = new Scanner(System.in);**

**// Taking input from the user**

**System.out.print("Enter a number to find its factorial: ");**

**int number = scanner.nextInt();**

**// Initializing the result to 1 (as factorial of 0 and 1 is 1)**

**long factorial = 1;**

**// Calculating the factorial using a for loop**

**for (int i = 1; i <= number; i++) {**

**factorial \*= i; // Multiply the current number to the result**

**}**

**// Printing the factorial of the given number**

**System.out.println("The factorial of " + number + " is: " + factorial);**

**scanner.close();**

**}**

**}**

**• W.A.J.P to check given number is Prime or not?**

**-----------------------------------------------------------**

**import java.util.Scanner;**

**public class PrimeCheck {**

**public static void main(String[] args) {**

**Scanner scanner = new Scanner(System.in);**

**// Taking input from the user**

**System.out.print("Enter a number to check if it's prime: ");**

**int number = scanner.nextInt();**

**// Variable to store if the number is prime**

**boolean isPrime = true;**

**// A prime number is greater than 1 and divisible only by 1 and itself**

**if (number <= 1) {**

**isPrime = false; // Numbers less than or equal to 1 are not prime**

**} else**

**{**

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

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

**if (number % i == 0) {**

**isPrime = false; // If divisible by any number other than 1 and itself, it's not prime**

**break;**

**break;**

**}**

**}**

**}**

**// Output the result**

**if (isPrime) {**

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

**} else {**

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

**}**

**scanner.close();**

**}**

**}**

**• W.A.J.P to check given number is Armstrong or not?**

**-------------------------------------------------------------**

**import java.util.Scanner;**

**public class ArmstrongNumber**

**{**

**public static void main(String[] args)**

**{**

**Scanner scanner = new Scanner(System.in);**

**// Taking input from the user**

**System.out.print("Enter a number to check if it's an Armstrong number: ");**

**int number = scanner.nextInt();**

**int originalNumber = number;**

**int sum = 0;**

**// Calculate the number of digits in the number**

**int digits = String.valueOf(number).length();**

**// Calculate the sum of the digits raised to the power of the number of digits**

**while (number > 0)**

**{**

**int digit = number % 10;**

**// Extract the last digit**

**sum += Math.pow(digit, digits);**

**// Add the digit raised to the power of the number of digits to sum**

**number /= 10;**

**// Remove the last digit**

**}**

**// Check if the sum is equal to the original number**

**if (sum == originalNumber) {**

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

**} else {**

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

**}**

**scanner.close();**

**}**

**}**

**• W.A.J.P for create Fibonacci Series.**

**---------------------------------------------**

**import java.util.Scanner;**

**public class FibonacciSeries {**

**public static void main(String[] args) {**

**Scanner scanner = new Scanner(System.in);**

**// Taking input from the user**

**System.out.print("Enter the number of terms in the Fibonacci series: ");**

**int n = scanner.nextInt();**

**// Variables to store the first two Fibonacci numbers**

**int first = 0, second = 1;**

**System.out.println("The Fibonacci series is:");**

**// Displaying the first term**

**System.out.print(first + " ");**

**// Displaying the remaining terms**

**for (int i = 1; i < n; i++) {**

**System.out.print(second + " ");**

**int next = first + second; // Calculating the next term in the series**

**first = second; // Updating first to second**

**second = next; // Updating second to next**

**}**

**scanner.close();**

**}**

**}**

**• W.A.J.P to Print pattern Given Below.**

**-----------------------------**

**1**

**12**

**123**

**1234**

**12345**

**public class NumberPattern {**

**public static void main(String[] args) {**

**int rows = 5; // Number of rows**

**for (int i = 1; i <= rows; i++) { // Loop through rows**

**for (int j = 1; j <= i; j++) { // Loop to print numbers in each row**

**System.out.print(j);**

**}**

**System.out.println(); // Move to the next line after each row**

**}**

**}**

**}**

**output**

**1**

**12**

**123**

**1234**

**12345**

**-------------------------------------------------------------------**

**1**

**01**

**101**

**0101**

**10101**

**public class BinaryPattern {**

**public static void main(String[] args) {**

**int rows = 5; // Number of rows**

**for (int i = 1; i <= rows; i++) { // Loop through rows**

**for (int j = 1; j <= i; j++) { // Loop to print numbers in each row**

**// Print 1 if the sum of i and j is even, else print 0**

**if ((i + j) % 2 == 0) {**

**System.out.print("1");**

**} else {**

**System.out.print("0");**

**}**

**}**

**System.out.println(); // Move to the next line after each row**

**}**

**}**

**}**

**----------------------------------------------------------**

**1**

**2 2**

**3 3 3**

**4 4 4 4**

**public class PyramidPattern {**

**public static void main(String[] args) {**

**int rows = 4; // Number of rows**

**for (int i = 1; i <= rows; i++) { // Loop through rows**

**// Print leading spaces**

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

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

**}**

**// Print the numbers with a space between them**

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

**System.out.print(i + " ");**

**}**

**// Move to the next line after each row**

**System.out.println();**

**}**

**}**

**}**

**\***

**\* \* \***

**\* \* \* \* \***

**\* \* \***

**\***

**public class DiamondPattern {**

**public static void main(String[] args) {**

**int rows = 4; // Number of rows in the widest part of the diamond**

**// Print the upper part of the diamond**

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

**// Print leading spaces**

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

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

**}**

**// Print stars with spaces in between**

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

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

**}**

**System.out.println(); // Move to the next line**

**}**

**// Print the lower part of the diamond**

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

**// Print leading spaces**

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

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

**}**

**// Print stars with spaces in between**

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

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

**}**

**System.out.println(); // Move to the next line**

**}**

**}**

**}**

**• WAP to compute the sum of the first 100 prime numbers**

**---------------------------------------------------------**

**public class SumOfPrimes {**

**public static void main(String[] args) {**

**int count = 0; // To count the number of prime numbers found**

**long sum = 0; // To store the sum of prime numbers**

**int number = 2; // The number to be checked for primality**

**while (count < 100) { // Continue until we find 100 prime numbers**

**if (isPrime(number)) {**

**sum += number; // Add the prime number to the sum**

**count++; // Increment the count of prime numbers**

**}**

**number++; // Move to the next number**

**}**

**// Print the sum of the first 100 prime numbers**

**System.out.println("The sum of the first 100 prime numbers is: " + sum);**

**}**

**// Method to check if a number is prime**

**public static boolean isPrime(int num) {**

**if (num <= 1) return false; // Numbers less than or equal to 1 are not prime**

**if (num <= 3) return true; // 2 and 3 are prime numbers**

**if (num % 2 == 0 || num % 3 == 0) return false; // Check divisibility by 2 and 3**

**// Check divisibility from 5 to sqrt(num)**

**for (int i = 5; i \* i <= num; i += 6) {**

**if (num % i == 0 || num % (i + 2) == 0) {**

**return false;**

**}**

**}**

**return true;**

**}**

**}**

**• WAP to sum values of an array.**

**-------------------------------------------**

**public class ArraySum {**

**public static void main(String[] args) {**

**// Initialize the array with some values**

**int[] numbers = {1, 2, 3, 4, 5}; // You can change these values or size as needed**

**// Variable to store the sum of the array elements**

**int sum = 0;**

**// Loop through the array and add each element to the sum**

**for (int number : numbers) {**

**sum += number;**

**}**

**// Print the sum of the array elements**

**System.out.println("The sum of the array values is: " + sum);**

**}**

**}**

**• WAP to calculate the average value of array elements.**

**-------------------------------------------------**

**public class ArrayAverage {**

**public static void main(String[] args) {**

**// Initialize the array with some values**

**int[] numbers = {10, 20, 30, 40, 50}; // You can change these values or size as needed**

**// Variable to store the sum of the array elements**

**int sum = 0;**

**// Loop through the array and add each element to the sum**

**for (int number : numbers) {**

**sum += number;**

**}**

**// Calculate the average**

**double average = (double) sum / numbers.length;**

**// Print the average of the array elements**

**System.out.println("The average value of the array elements is: " + average);**

**}**

**}**

**• WAP to find the index of an array element.**

**-----------------------------------------------------**

**public class FindElementIndex {**

**public static void main(String[] args) {**

**// Initialize the array with some values**

**int[] numbers = {10, 20, 30, 40, 50}; // You can change these values as needed**

**// Element to find in the array**

**int target = 30; // Change this value to the element you want to find**

**// Variable to store the index of the target element**

**int index = -1; // -1 indicates that the element was not found**

**// Loop through the array to find the target element**

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

**if (numbers[i] == target) {**

**index = i; // Store the index of the target element**

**break; // Exit the loop once the element is found**

**}**

**}**

**// Print the result**

**if (index != -1) {**

**System.out.println("The index of element " + target + " is: " + index);**

**} else {**

**System.out.println("Element " + target + " is not found in the array.");**

**}**

**}**

**}**

**• WAP to find the maximum and minimum value of an array.**

**---------------------------------------------------------**

**public class MinMaxArray {**

**public static void main(String[] args) {**

**// Initialize the array with some values**

**int[] numbers = {45, 22, 89, 34, 67, 23, 90}; // You can change these values as needed**

**// Check if the array is not empty**

**if (numbers.length == 0) {**

**System.out.println("The array is empty.");**

**return;**

**}**

**// Initialize min and max with the first element of the array**

**int min = numbers[0];**

**int max = numbers[0];**

**// Loop through the array to find the min and max values**

**for (int i = 1; i < numbers.length; i++) {**

**if (numbers[i] < min) {**

**min = numbers[i]; // Update min if a smaller value is found**

**}**

**if (numbers[i] > max) {**

**max = numbers[i]; // Update max if a larger value is found**

**}**

**}**

**// Print the result**

**System.out.println("The minimum value in the array is: " + min);**

**System.out.println("The maximum value in the array is: " + max);**

**}**

**}**

**• WAP to Compare Two String.**

**--------------------------------------**

**import java.util.Scanner;**

**public class CompareStrings {**

**public static void main(String[] args) {**

**Scanner scanner = new Scanner(System.in);**

**// Prompt the user to enter the first string**

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

**String string1 = scanner.nextLine();**

**// Prompt the user to enter the second string**

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

**String string2 = scanner.nextLine();**

**// Compare strings using equals()**

**if (string1.equals(string2)) {**

**System.out.println("The strings are equal.");**

**} else {**

**System.out.println("The strings are not equal.");**

**}**

**// Compare strings using compareTo()**

**int comparisonResult = string1.compareTo(string2);**

**if (comparisonResult < 0) {**

**System.out.println("The first string is less than the second string.");**

**} else if (comparisonResult > 0) {**

**System.out.println("The first string is greater than the second string.");**

**} else {**

**System.out.println("The strings are equal.");**

**}**

**scanner.close();**

**}**

**}**

**• WAP to concatenate a given string to the end of another string.**

**----------------------------------------------------------------**

**import java.util.Scanner;**

**public class StringConcatenation {**

**public static void main(String[] args) {**

**Scanner scanner = new Scanner(System.in);**

**// Prompt the user to enter the first string**

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

**String firstString = scanner.nextLine();**

**// Prompt the user to enter the second string**

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

**String secondString = scanner.nextLine();**

**// Concatenate using concat() method**

**String concatenatedString1 = firstString.concat(secondString);**

**// Concatenate using + operator**

**String concatenatedString2 = firstString + secondString;**

**// Print the results**

**System.out.println("Concatenated using concat(): " + concatenatedString1);**

**System.out.println("Concatenated using + operator: " + concatenatedString2);**

**scanner.close();**

**}**

**}**

**• WAP to demonstrate try catch block.**

**---------------------------------------**

**import java.util.Scanner;**

**public class TryCatchDemo {**

**public static void main(String[] args) {**

**Scanner scanner = new Scanner(System.in);**

**// Prompt the user to enter two numbers**

**System.out.print("Enter the numerator: ");**

**int numerator = scanner.nextInt();**

**System.out.print("Enter the denominator: ");**

**int denominator = scanner.nextInt();**

**try {**

**// Attempt to perform division**

**int result = numerator / denominator;**

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

**} catch (ArithmeticException e) {**

**// Handle division by zero error**

**System.out.println("Error: Cannot divide by zero.");**

**} catch (Exception e) {**

**// Handle any other unexpected exceptions**

**System.out.println("An unexpected error occurred: " + e.getMessage());**

**} finally {**

**// This block will always execute, whether an exception occurs or not**

**System.out.println("Execution completed.");**

**scanner.close();**

**}**

**}**

**}**

**• WAP to demonstrate multiple catch blocks**

**------------------------------------------------------**

**import java.util.InputMismatchException;**

**import java.util.Scanner;**

**public class MultipleCatchDemo {**

**public static void main(String[] args) {**

**Scanner scanner = new Scanner(System.in);**

**try {**

**// Prompt the user to enter two numbers**

**System.out.print("Enter the numerator: ");**

**int numerator = scanner.nextInt();**

**System.out.print("Enter the denominator: ");**

**int denominator = scanner.nextInt();**

**// Attempt to perform division**

**int result = numerator / denominator;**

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

**} catch (ArithmeticException e) {**

**// Handle division by zero error**

**System.out.println("Error: Cannot divide by zero.");**

**} catch (InputMismatchException e) {**

**// Handle invalid input type (e.g., if user enters a non-integer value)**

**System.out.println("Error: Invalid input. Please enter integers only.");**

**} catch (Exception e) {**

**// Handle any other unexpected exceptions**

**System.out.println("An unexpected error occurred: " + e.getMessage());**

**} finally {**

**// This block will always execute, whether an exception occurs or not**

**System.out.println("Execution completed.");**

**scanner.close();**

**}**

**}**

**}**

**• WAP to create one thread by implementing Runnable interface in Class.**

**--------------------------------------------------------------------------**

**To create a thread in Java using the Runnable interface, you need to follow these steps:**

1. **Implement the Runnable interface: Define a class that implements Runnable and override its run() method with the code that should be executed by the thread.**
2. **Create a Thread object: Instantiate a Thread object with an instance of the Runnable class.**
3. **Start the thread: Call the start() method on the Thread object to begin execution.**

**Here's an example of how to do this:**

**// Implement the Runnable interface**

**class MyRunnable implements Runnable {**

**@Override**

**public void run() {**

**// Code to be executed by the thread**

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

**System.out.println("Thread is running: " + i);**

**try {**

**Thread.sleep(1000); // Sleep for 1 second**

**} catch (InterruptedException e) {**

**System.out.println("Thread interrupted.");**

**}**

**}**

**}**

**}**

**public class RunnableExample {**

**public static void main(String[] args) {**

**// Create an instance of MyRunnable**

**MyRunnable myRunnable = new MyRunnable();**

**// Create a Thread object and pass the MyRunnable instance**

**Thread thread = new Thread(myRunnable);**

**// Start the thread**

**thread.start();**

**// Main thread continues to run in parallel**

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

**System.out.println("Main thread: " + i);**

**try {**

**Thread.sleep(500); // Sleep for 0.5 seconds**

**} catch (InterruptedException e) {**

**System.out.println("Main thread interrupted.");**

**}**

**}**

**}**

**}**

**• WAP to create one thread by extending Thread class in another Class.**

**-----------------------------------------------**

**// Extend the Thread class**

**class MyThread extends Thread {**

**@Override**

**public void run() {**

**// Code to be executed by the thread**

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

**System.out.println("Thread is running: " + i);**

**try {**

**Thread.sleep(1000); // Sleep for 1 second**

**} catch (InterruptedException e) {**

**System.out.println("Thread interrupted.");**

**}**

**}**

**}**

**}**

**public class ThreadExample {**

**public static void main(String[] args) {**

**// Create an instance of MyThread**

**MyThread myThread = new MyThread();**

**// Start the thread**

**myThread.start();**

**// Main thread continues to run in parallel**

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

**System.out.println("Main thread: " + i);**

**try {**

**Thread.sleep(500); // Sleep for 0.5 seconds**

**} catch (InterruptedException e) {**

**System.out.println("Main thread interrupted.");**

**}**

**}**

**}**

**}**

**• WAP to iterate through all elements in an array list.**

**-----------------------------------------**

**To iterate through all elements in an ArrayList in Java, you can use several methods:**

1. **Using a for-each loop.**
2. **Using an Iterator.**
3. **Using a for loop with an index.**
4. **Using Java 8's forEach method with a lambda expression.**

**Here's an example demonstrating each method:**

**import java.util.ArrayList;**

**import java.util.Iterator;**

**public class ArrayListIteration {**

**public static void main(String[] args) {**

**// Create an ArrayList and add some elements**

**ArrayList<String> list = new ArrayList<>();**

**list.add("Apple");**

**list.add("Banana");**

**list.add("Cherry");**

**list.add("Date");**

**// 1. Using for-each loop**

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

**for (String fruit : list) {**

**System.out.println(fruit);**

**}**

**// 2. Using Iterator**

**System.out.println("\nUsing Iterator:");**

**Iterator<String> iterator = list.iterator();**

**while (iterator.hasNext()) {**

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

**}**

**// 3. Using for loop with index**

**System.out.println("\nUsing for loop with index:");**

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

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

**}**

**// 4. Using Java 8 forEach with lambda**

**System.out.println("\nUsing Java 8 forEach with lambda:");**

**list.forEach(fruit -> System.out.println(fruit));**

**}**

**}**

**• WAP to update specific array element by given element.**

**------------------------------------------------**

**import java.util.Scanner;**

**public class UpdateArrayElement {**

**public static void main(String[] args) {**

**Scanner scanner = new Scanner(System.in);**

**// Initialize the array with some values**

**int[] numbers = {10, 20, 30, 40, 50};**

**// Display the original array**

**System.out.println("Original array:");**

**for (int num : numbers) {**

**System.out.print(num + " ");**

**}**

**System.out.println();**

**// Prompt the user to enter the index of the element to update**

**System.out.print("Enter the index of the element to update (0 to " + (numbers.length - 1) + "): ");**

**int index = scanner.nextInt();**

**// Check if the index is valid**

**if (index < 0 || index >= numbers.length) {**

**System.out.println("Error: Index out of bounds.");**

**} else {**

**// Prompt the user to enter the new value**

**System.out.print("Enter the new value: ");**

**int newValue = scanner.nextInt();**

**// Update the array element**

**numbers[index] = newValue;**

**// Display the updated array**

**System.out.println("Updated array:");**

**for (int num : numbers) {**

**System.out.print(num + " ");**

**}**

**System.out.println();**

**}**

**scanner.close();**

**}**

**}**

**• WAP to remove the third element from a array list.**

**---------------------------------------------------------------**

**import java.util.ArrayList;**

**import java.util.Scanner;**

**public class RemoveThirdElement {**

**public static void main(String[] args) {**

**// Create an ArrayList and add some elements**

**ArrayList<String> list = new ArrayList<>();**

**list.add("Apple");**

**list.add("Banana");**

**list.add("Cherry");**

**list.add("Date");**

**list.add("Elderberry");**

**// Display the original ArrayList**

**System.out.println("Original ArrayList:");**

**for (String item : list) {**

**System.out.println(item);**

**}**

**// Remove the third element if it exists**

**if (list.size() >= 3) {**

**list.remove(2); // Index 2 corresponds to the third element**

**} else {**

**System.out.println("The ArrayList does not contain enough elements to remove the third one.");**

**}**

**// Display the updated ArrayList**

**System.out.println("\nUpdated ArrayList:");**

**for (String item : list) {**

**System.out.println(item);**

**}**

**}**

**}**

**• WAP to Copy one array into another**

**--------------------------------------**

**1. Using a Manual Loop**

**public class CopyArrayManual {**

**public static void main(String[] args) {**

**// Source array**

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

**// Create a destination array of the same length**

**int[] destinationArray = new int[sourceArray.length];**

**// Copy elements using a loop**

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

**destinationArray[i] = sourceArray[i];**

**}**

**// Display the destination array**

**System.out.println("Destination Array:");**

**for (int num : destinationArray) {**

**System.out.print(num + " ");**

**}**

**}**

**}**

**2.Using System.arraycopy()**

**public class CopyArraySystemArraycopy {**

**public static void main(String[] args) {**

**// Source array**

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

**// Create a destination array of the same length**

**int[] destinationArray = new int[sourceArray.length];**

**// Copy elements using System.arraycopy**

**System.arraycopy(sourceArray, 0, destinationArray, 0, sourceArray.length);**

**// Display the destination array**

**System.out.println("Destination Array:");**

**for (int num : destinationArray) {**

**System.out.print(num + " ");**

**}**

**}**

**}**

**3.Using Arrays.copyOf()**

**import java.util.Arrays;**

**public class CopyArrayArraysCopyOf {**

**public static void main(String[] args) {**

**// Source array**

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

**// Copy elements using Arrays.copyOf**

**int[] destinationArray = Arrays.copyOf(sourceArray, sourceArray.length);**

**// Display the destination array**

**System.out.println("Destination Array:");**

**for (int num : destinationArray) {**

**System.out.print(num + " ");**

**}**

**}**

**}**

**• WAP to reverse an array of integer values.**

**---------------------------------------**

**public class ReverseArray {**

**public static void main(String[] args) {**

**// Initialize the array with some values**

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

**// Display the original array**

**System.out.println("Original array:");**

**for (int num : array) {**

**System.out.print(num + " ");**

**}**

**System.out.println();**

**// Reverse the array**

**reverseArray(array);**

**// Display the reversed array**

**System.out.println("Reversed array:");**

**for (int num : array) {**

**System.out.print(num + " ");**

**}**

**}**

**// Method to reverse the array**

**public static void reverseArray(int[] array) {**

**int start = 0;**

**int end = array.length - 1;**

**while (start < end) {**

**// Swap elements at start and end indices**

**int temp = array[start];**

**array[start] = array[end];**

**array[end] = temp;**

**// Move indices towards the center**

**start++;**

**end--;**

**}**

**}**

**}**

**• WAP to find the second largest element in an array.**

**----------------------------------------------**

**To find the second largest element in an array, you can follow these steps:**

1. **Initialize Variables: Use two variables to keep track of the largest and the second largest elements.**
2. **Iterate Through the Array: Compare each element with the largest and second largest values and update them accordingly.**
3. **Handle Edge Cases: Ensure that the array has at least two distinct elements to find the second largest element.**

**Here’s a Java program to find the second largest element in an array:**

**public class SecondLargestElement {**

**public static void main(String[] args) {**

**// Initialize the array with some values**

**int[] array = {10, 5, 20, 8, 15};**

**// Call the method to find the second largest element**

**int secondLargest = findSecondLargest(array);**

**// Display the result**

**if (secondLargest != Integer.MIN\_VALUE) {**

**System.out.println("The second largest element is: " + secondLargest);**

**} else {**

**System.out.println("The array does not have enough distinct elements.");**

**}**

**}**

**// Method to find the second largest element in the array**

**public static int findSecondLargest(int[] array) {**

**if (array.length < 2) {**

**// Not enough elements to determine the second largest**

**return Integer.MIN\_VALUE;**

**}**

**int largest = Integer.MIN\_VALUE;**

**int secondLargest = Integer.MIN\_VALUE;**

**for (int num : array) {**

**if (num > largest) {**

**// Update both largest and second largest**

**secondLargest = largest;**

**largest = num;**

**} else if (num > secondLargest && num < largest) {**

**// Update only second largest**

**secondLargest = num;**

**}**

**}**

**return secondLargest;**

**}**

**}**

**• W.A.J.P. Create an abstract class 'Parent' with a method 'message'. It has two subclasses each having a method with the same name 'message' that prints "This is first subclass" and "This is second subclass" respectively. Call the methods 'message' by creating an object for each subclass.**

**----------------------------------------------------------------------------------------------**

**// Abstract class Parent**

**abstract class Parent {**

**// Abstract method**

**abstract void message();**

**}**

**// First subclass**

**class FirstSubclass extends Parent {**

**// Override the message method**

**@Override**

**void message() {**

**System.out.println("This is the first subclass");**

**}**

**}**

**// Second subclass**

**class SecondSubclass extends Parent {**

**// Override the message method**

**@Override**

**void message() {**

**System.out.println("This is the second subclass");**

**}**

**}**

**public class AbstractClassDemo {**

**public static void main(String[] args) {**

**// Create an object of FirstSubclass**

**Parent firstObject = new FirstSubclass();**

**// Call the message method**

**firstObject.message();**

**// Create an object of SecondSubclass**

**Parent secondObject = new SecondSubclass();**

**// Call the message method**

**secondObject.message();**

**}**

**}**

**• W.A.J.P. which will ask the user to enter his/her marks (out of 100). Define a method that will display grades according to the marks entered as below: Marks Grade 91-100 AA 81-90 AB 71-80 BB 61-70 BC 51-60 CD 41-50 DD <=40 Fail**

**-------------------------------------------------------**

**import java.util.Scanner;**

**public class GradeCalculator {**

**public static void main(String[] args) {**

**Scanner scanner = new Scanner(System.in);**

**// Prompt the user to enter marks**

**System.out.print("Enter your marks (out of 100): ");**

**int marks = scanner.nextInt();**

**// Validate marks input**

**if (marks < 0 || marks > 100) {**

**System.out.println("Error: Marks should be between 0 and 100.");**

**} else {**

**// Call the method to display the grade**

**displayGrade(marks);**

**}**

**scanner.close();**

**}**

**// Method to determine and display the grade based on the marks**

**public static void displayGrade(int marks) {**

**if (marks >= 91) {**

**System.out.println("Grade: AA");**

**} else if (marks >= 81) {**

**System.out.println("Grade: AB");**

**} else if (marks >= 71) {**

**System.out.println("Grade: BB");**

**} else if (marks >= 61) {**

**System.out.println("Grade: BC");**

**} else if (marks >= 51) {**

**System.out.println("Grade: CD");**

**} else if (marks >= 41) {**

**System.out.println("Grade: DD");**

**} else {**

**System.out.println("Grade: Fail");**

**}**

**}**

**}**

**• W.A.J.P. to create a custom exception if Customer withdraw amount which is greater than account balance then program will show custom exception otherwise amount will deduct from account balance. Account balance is:2000 Enter withdraw amount:2500 Sorry, insufficient balance, you need more 500 Rs.To perform this transaction.**

**-----------------------------------------------------------------------**

**1. Define the Custom Exception**

**// Custom exception class**

**class InsufficientBalanceException extends Exception {**

**public InsufficientBalanceException(String message) {**

**super(message);**

**}**

**}**

**2.Create the Account Class**

**import java.util.Scanner;**

**public class Account {**

**private int balance;**

**// Constructor to initialize account with balance**

**public Account(int initialBalance) {**

**this.balance = initialBalance;**

**}**

**// Method to withdraw amount**

**public void withdraw(int amount) throws InsufficientBalanceException {**

**if (amount > balance) {**

**// Calculate the deficit**

**int deficit = amount - balance;**

**// Throw custom exception with a message**

**throw new InsufficientBalanceException("Sorry, insufficient balance, you need more " + deficit + " Rs. to perform this transaction.");**

**} else {**

**// Deduct the amount from the balance**

**balance -= amount;**

**System.out.println("Withdrawal successful. Remaining balance: " + balance + " Rs.");**

**}**

**}**

**}**

1. **Main Program to Test the Account Class**

**import java.util.Scanner;**

**public class BankingApp {**

**public static void main(String[] args) {**

**// Initialize account with balance of 2000 Rs.**

**Account account = new Account(2000);**

**Scanner scanner = new Scanner(System.in);**

**// Prompt user to enter the withdrawal amount**

**System.out.print("Enter withdrawal amount: ");**

**int withdrawAmount = scanner.nextInt();**

**try {**

**// Attempt to withdraw the amount**

**account.withdraw(withdrawAmount);**

**} catch (InsufficientBalanceException e) {**

**// Handle custom exception**

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

**} finally {**

**// Close the scanner**

**scanner.close();**

**}**

**}**

**}**