**xCHAROTAR UNIVERSITY OF SCIENCE & TECHNOLOGY**

**DEVANG PATEL INSTITUTE OF ADVANCE TECHNOLOGY & RESEARCH**

Department of Computer Engineering/Computer Science & Engineering/Information Technology

**Subject Name: JAVA PROGRAMMING**

**Semester: III**

**Subject Code: CSE201**

**Academic year: 2024-25**

Part - 1

|  |  |
| --- | --- |
| **No.** | **Aim of the Practical** |
| 1. | AIM : Demonstration of installation steps of Java,Introduction to Object Oriented Concepts, comparison of Java with other object-oriented programming languages. Introduction to JDK, JRE, JVM, Javadoc, command line argument. Introduction to Eclipse or NetBeans IDE,or BlueJ and Console Programming. **CONCLUSION:**  The comparison of Java with other object-oriented languages highlighted Java's unique features and advantages, such as platform independence and robust API. The introduction to Java's core components—JDK, JRE, JVM, and Javadoc—offered insights into the architecture and tools that make Java applications efficient and portable. Additionally, the discussion on command line arguments emphasized Java's versatility in handling input. The exploration of IDEs like Eclipse, NetBeans, or BlueJ, along with Console Programming, showcased the diverse development environments available for Java, catering to different preferences and project requirements. |

|  |  |
| --- | --- |
| 2. | AIM : Imagine you are developing a simple banking application where you need to display the current balance of a user account. For simplicity, let's say the current balance is $20. Write a java program to store this balance in a variable and then display it to the user. **PROGRAM CODE :**  import java.util.Scanner;  public class jprac\_2 {  public static void main(String[] args) {  int balance=20,no;  System.out.print("Enter your account number : ");  Scanner s = new Scanner(System.in);  no =s.nextInt();  if(no==1)  {  System.out.print("Your curent balance is : "+ balance+"$");  }  else {System.out.print("Enter valid account number");}  System.out.print("\nName : DHAVAL DESAI \nID : 23DCS020 ");  }  }  **OUTPUT:**    **CONCLUSION:**  This code demonstrates a simple Java program that prompts the user for an account number, checks if it matches a predefined value (1), and then displays the current balance if the match is successful. If the account number does not match, it prompts the user to enter a valid account number. The program uses basic control flow with an if-else statement to determine the output based on the user's input. It also illustrates the use of the Scanner class for reading user input from the console. Additionally, the program prints the name and ID of the author or a placeholder individual, showcasing basic string concatenation and output in Java. This code serves as a fundamental example of conditional logic, user input |

|  |  |
| --- | --- |
| 3. | AIM : Write a program to take the user for a distance (in meters) and the time taken (as three numbers: hours, minutes, seconds), and display the speed, in meters per second, kilometers per hour and miles per hour (hint:1 mile = 1609 meters).PROGRAM CODE : import java.util.Scanner;  public class jprac\_3  {  public static void main(String[] args) {  try (Scanner sc = new Scanner(System.in)) {  System.out.println("Enter the time taken in Hours : ");  int hr=sc.nextInt();  System.out.println("Enter the time taken in Minutes : ");  int min =sc.nextInt();  System.out.println("Enter the time taken in second : ");  int sec=sc.nextInt();  System.out.println("Enter the Distance coverd in meter : ");  int Distance=sc.nextInt();  int totalSec=( hr\*3600 )+( min\*60)+sec;  double kmdis=Distance/1000;  double miledis= Distance/1609;  double total\_hours=hr+(min/60) + (sec/3600);  double velocity1 = Distance/totalSec;  double velocity2=kmdis/total\_hours;  double velocity3=miledis/total\_hours;  System.out.println("Velocity int the Different units : ");  System.out.println("The velocity is in ( Meter/Sec) : "+velocity1);  System.out.println("The velocity is in ( KM/Hour) : "+velocity2);  System.out.println("The velocity is in ( mile/Hour) : "+velocity3);  }  }  }  **OUTPUT:**    **CONCLUSION:**  This program calculates and displays the speed of an object in meters per second, kilometers per hour, and miles per hour based on the user-provided distance (in meters) and time (in hours, minutes, and seconds). It utilizes the Scanner class for input, demonstrating basic Java input/output operations and arithmetic calculations. The conversion of distance to kilometers and miles, along with the conversion of time into seconds and hours, showcases practical applications of mathematical concepts in programming. The program outputs the calculated speeds in different units, providing a clear example of unit conversion and formatting output in Java. It concludes with the author's name and ID, personalizing the code output. This code snippet is a straightforward example of applying mathematical formulas and user input in Java to solve real-world problems. |

|  |  |
| --- | --- |
| 4. | AIM : Imagine you are developing a budget tracking application. You need to calculate the total expenses for the month. Users will input their daily expenses, and the program should compute the sum of these expenses. Write a Java program to calculate the sum of elements in an array representing daily expenses.PROGRAM CODE : import java.util.Scanner;  public class jprac\_04 {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  System.out.print("Enter the number of days you have expenses for: ");  int numberOfDays = scanner.nextInt();  double[] dailyExpenses = new double[numberOfDays];  for (int i = 0; i < numberOfDays; i++) {  System.out.print("Enter expense for day " + (i + 1) + ": ");  dailyExpenses[i] = scanner.nextDouble();  }  double totalExpenses = calculateTotalExpenses(dailyExpenses);  System.out.println("Total Expenses for the Month: " + totalExpenses + "/- Rs.");  System.out.print("\nName : DHAVAL DESAI \nID : 23DCS020 ");  }  public static double calculateTotalExpenses(double[] expenses) {  double total = 0;  for (int i = 0; i < expenses.length; i++) {  total += expenses[i];  }  return total;  }  }  **OUTPUT:**    **CONCLUSION:**  This program is designed to calculate and display the total monthly expenses based on daily inputs from the user. It starts by asking the user for the number of days they have expense data for, then collects expense amounts for each day using a loop. These amounts are stored in an array of doubles. The program calculates the total expenses by passing this array to the calculateTotalExpenses method, which iterates over the array to sum the expenses. Finally, it displays the total expenses to the user. This program demonstrates basic Java concepts such as arrays, loops, methods, and user input handling with the Scanner class. It concludes with displaying the author's name and ID, personalizing the output. |

|  |  |
| --- | --- |
| 5. | AIM : An electric appliance shop assigns code 1 to motor,2 to fan,3 to tube and 4 for wires. All other items have code 5 or more. While selling the goods, a sales tax of 8% to motor,12% to fan,5% to tube light,7.5% to wires and 3% for all other items is charged. A list containing the product code and price in two different arrays. Write a java program using switch statement to prepare the bill.PROGRAM CODE : import java.util.Scanner;  public class jprac\_05 {  public static void main(String[] args) {  Scanner ip = new Scanner(System.in);  float[] price = {100, 100, 100, 100, 100};  float totalPrice = 0;  System.out.println("Welcome To Shop!");  float tax = 0;  int a;  do {  System.out.println("Enter product code (1 for motor, 2 for fan, 3 for tube, 4 for wires, 5 for others, 0 to finish):");  a = ip.nextInt();  switch (a) {  case 1:  tax = (price[0] \* 8) / 100;  break;  case 2:  tax = (price[1] \* 12) / 100;  break;  case 3:  tax = (price[2] \* 5) / 100;  break;  case 4:  tax = (price[3] \* 7.5f) / 100;  break;  case 5:  tax = (price[4] \* 3) / 100;  break;  default:  tax = 0;  break;  }  if (a > 0 && a <= 5) {  totalPrice += price[a - 1] + tax;  } else {  totalPrice += 0;  }  } while (a != 0);  System.out.println("Total Bill: " + totalPrice);  System.out.print("\nName : DHAVAL DESAI \nID : 23DCS020 ");  }  }**OUTPUT:**    **CONCLUSION:**  This program is a simple shopping cart calculator that computes the total bill including tax for a set of predefined products. It uses a do-while loop to continuously prompt the user for product codes until the user decides to finish by entering 0. Tax rates are applied differently based on the product selected, using a switch statement to determine the tax amount for each product. The total price, including tax, is accumulated and displayed at the end. This program demonstrates basic Java constructs such as arrays, loops, conditional statements, and user input handling with the Scanner class. It concludes by displaying the author's name and ID, adding a personal touch to the output. |

|  |  |
| --- | --- |
| 6. | AIM : Create a Java program that prompts the user to enter the number of days (n) for which they want to generate their exercise routine. The program should then calculate and display the first n terms of the Fibonacci series, representing the exercise duration for each day.PROGRAM CODE : import java.util.Scanner;  public class jprac\_06 {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  System.out.print("Enter the number of days for your exercise routine: ");  int n = scanner.nextInt();  int firstTerm = 0, secondTerm = 1;  System.out.println("Your exercise routine duration for " + n + " days:");  for (int i = 1; i <= n; ++i) {  System.out.println("Day " + i + ": " + firstTerm + " minutes");  int nextTerm = firstTerm + secondTerm;  firstTerm = secondTerm;  secondTerm = nextTerm;  }  System.out.print("\nName : DHAVAL DESAI \nID : 23DCS020 ");  }  }  **OUTPUT:**    **CONCLUSION:**  This program is designed to generate an exercise routine schedule based on the Fibonacci sequence, where the duration for each day is derived from this sequence. It prompts the user to enter the number of days they plan to follow the routine. Utilizing a simple for loop, it calculates the duration for each day, starting with 0 minutes on the first day and 1 minute on the second, then follows the Fibonacci pattern for subsequent days. The output is a daily exercise plan displaying the duration in minutes for the specified number of days. This program demonstrates the application of loops, basic arithmetic operations, and user input handling in Java. It concludes with displaying the author's name and ID, personalizing the output. |