**PART-V Exception Handling**

|  |  |
| --- | --- |
| **No.** | **Aim of the Practical** |
| 24. | AIM :-Write a java program which takes two integers x & y asinput, you have to compute x/y. If x and y are not integersor if y is zero, exception will occur and you have toreport it. **PROGRAM CODE :**  import java.util.InputMismatchException;  import java.util.Scanner;  public class jprac\_24 {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.in);  try {  System.out.print("Enter the first integer (x): ");  int x = scanner.nextInt();  System.out.print("Enter the second integer (y): ");  int y = scanner.nextInt();  int result = x / y;  System.out.println("Result of " + x + " / " + y + " = " + result);  } catch (InputMismatchException e) {  System.out.println("Invalid input.--> Please enter integers only.");  } catch (ArithmeticException e) {  System.out.println(e.getMessage());  } finally {  scanner.close();  }  System.out.print("\nName : DHAVAL DESAI \nID : 23DCS020 ");  }  }  **OUTPUT:**    **CONCLUSION:**  In this practical, we have demonstrated how to handle exceptions in Java using try-catch blocks. The program takes two integers as input and attempts to compute their division. It handles two types of exceptions:  InputMismatchException - This occurs when the input is not an integer.  ArithmeticException - This occurs when there is an attempt to divide by zero. |
| 25. | AIM : Write a Java program that throws an exception and catchit using a try-catch block.PROGRAM CODE : import java.util.Scanner;  public class jprac\_25 {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  try {  System.out.println("Enter two numbers : ");  int a = sc.nextInt();  int b = sc.nextInt();  int c = a / b;  System.out.println("Result : " + c);  } catch (ArithmeticException e) {  System.out.println("Arithmetic Exception: " + e);  }  sc.close();  System.out.print("\nName : DHAVAL DESAI \nID : 23DCS020 ");  }  }  **OUTPUT:**    **CONCLUSION:**  In this practical, we have demonstrated how to handle exceptions in Java using a try-catch block. The program takes two integers as input and attempts to compute their division. It specifically handles the ArithmeticException that occurs when there is an attempt to divide by zero. By catching this exception, the program can provide a meaningful error message to the user instead of terminating abruptly. |
| 26. | AIM : Write a java program to generate user defined exceptionusing “throw” and “throws” keyword.Also Write a java that differentiates checked andunchecked exceptions. (Mention at least two checked and two unchecked exceptions in program).PROGRAM CODE : import java.util.Scanner;  public class jprac\_26 {  public static void main(String[] args) {  Scanner sc= new Scanner(System.in);  try {  System.out.println("Enter two numbers : ");  int a = sc.nextInt();  int b = sc.nextInt();  if (b == 0) {  throw new ArithmeticException("Division by zero is not allowed.");  }  int c = a/b;  System.out.println("Result : " + c);  } catch (ArithmeticException e) {  System.out.println("Arithmetic Exception: " + e);  }  System.out.print("\nName : DHAVAL DESAI \nID : 23DCS020 ");  }  }  **OUTPUT:**    **CONCLUSION:**  In this practical, we demonstrated the use of throw and throws keywords to handle exceptions in Java. The program differentiates between checked and unchecked exceptions by handling ArithmeticException (unchecked) and simulating other exceptions. Additionally, it shows how to generate and handle user-defined exceptions, ensuring robust error handling and improving program reliability. |