1. **Declaration and Initialization:**

int num1, num2, result; Scanner sc = new Scanner(System.in);

Here, it declares three integers (**num1**, **num2**, and **result**) and creates a **Scanner** object (**sc**) for user input.

2. **User Input:**

System.out.println("Enter First number:"); num1 = Integer.parseInt(sc.next()); System.out.println("Enter Second number:"); num2 = Integer.parseInt(sc.next());

Asks the user to input two numbers, converts the input to integers, and stores them in **num1** and **num2**.

3. **Division:**

result = num1 / num2; System.out.println("Division is:" + result);

Performs division of **num1** by **num2** and prints the result. This section may throw an **ArithmeticException** if the user inputs 0 as the second number.

4. **Array Declaration and Access:**

int x[] = new int[12]; x[15] = 60;

Declares an integer array **x** of size 12 and tries to assign a value to the 16th element (index 15). This may throw an **ArrayIndexOutOfBoundsException** since arrays in Java are zero-indexed, and attempting to access an element beyond the array size results in an error.

5. **Exception Handling:**

catch (ArithmeticException e) { System.out.println(e); } catch (NumberFormatException e) { System.out.println(e); } catch (ArrayIndexOutOfBoundsException e) { System.out.println(e); }

Catches and handles specific types of exceptions. If there is an arithmetic error (division by zero), a number format error (invalid input), or an array index error, it prints the corresponding exception.

In summary, the program takes two numbers from the user, performs division, attempts to access an invalid array index, and uses exception handling to manage potential errors