Lab 30

Exception Handling in Java, try catch blocks and related concepts.

Task: Demonstrate exception handling using sample codes:

1. Handling Arithmetic Exception:

In this program, we divide two numbers and handle the `ArithmeticException` that occurs when attempting to divide by zero.

public class ArithmeticExceptionDemo {

public static void main(String[] args) {

try {

int result = divide(10, 0); // Attempting to divide by zero

System.out.println("Result: " + result);

} catch (ArithmeticException e) {

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

}

}

static int divide(int a, int b) {

return a / b;

}

}

In this program, the `divide` method divides two numbers, and the `try-catch` block in the `main` method catches the `ArithmeticException` that occurs when trying to divide by zero. It then prints an error message.

2. Handling File Not Found Exception:

In this program, we attempt to read data from a file, and we handle the `FileNotFoundException` that may occur if the file does not exist.

import java.io.BufferedReader;

import java.io.FileReader;

import java.io.IOException;

public class FileNotFoundExceptionDemo {

public static void main(String[] args) {

try {

String content = readFile("nonexistent.txt");

System.out.println("File Content: " + content);

} catch (IOException e) {

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

}

}

static String readFile(String fileName) throws IOException {

BufferedReader reader = new BufferedReader(new FileReader(fileName));

StringBuilder content = new StringBuilder();

String line;

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

content.append(line).append("\n");

}

reader.close();

return content.toString();

}

}

In this program, the `readFile` method attempts to read the content of a file specified by the `fileName`. If the file does not exist, it throws a `FileNotFoundException`, which is caught and handled in the `try-catch` block in the `main` method. It prints an error message in case of an exception.

Both of these programs demonstrate how to handle exceptions gracefully using `try-catch` blocks, ensuring that the programs do not terminate abruptly when errors occur.