Day 10 – Exception Handling in Java

Objective:

To understand the concept of exceptions in Java, learn how to handle runtime errors using try-catch blocks, and implement reliable and error-free programs.

Content:

Today, I studied **exception handling**, one of Java's most important mechanisms for dealing with unexpected errors that occur during program execution. It ensures smooth program flow without abrupt termination.

---

1. What is an Exception?

An **exception** is an event that disrupts the normal flow of a program.
Java provides a structured way to handle such conditions using *try-catch-finally* blocks.

Common Exception Types:

- **ArithmeticException** – division by zero
- **NullPointerException** – accessing null objects
- **ArrayIndexOutOfBoundsException** – invalid array index
- **FileNotFoundException** – file not found during I/O
- **IOException** – general input/output errors

---

2. Exception Handling Keywords

Keyword	Description
`try`	Block of code that may throw an exception
`catch`	Handles the exception
`finally`	Executes code regardless of exception
`throw`	Used to explicitly throw an exception

`throws`	Declares exceptions in method signature

Example:

```
public class ExceptionDemo {
    public static void main(String[] args) {
        try {
            int a = 10, b = 0;
            int result = a / b;
            System.out.println(result);
        } catch (ArithmeticException e) {
            System.out.println("Error: Division by zero!");
        } finally {
            System.out.println("Execution completed.");
        }
    }
}
```

Output:

```
Error: Division by zero!
Execution completed.
```

---

3. Custom Exceptions

We can create our own exceptions by extending the `Exception` class.

```
class InvalidAgeException extends Exception {
    InvalidAgeException(String msg) {
        super(msg);
    }}
```

Learning Outcome:

Understood how exceptions occur and how to manage them gracefully using try-catch-finally blocks.

Learned about predefined and user-defined exceptions.
Gained the ability to write programs that handle errors efficiently without crashing.