# Day 11 – File Handling and Streams in Java

## Objective:

To learn how to perform file input and output operations in Java using streams and understand how data can be stored and retrieved from external files.

---

## Content:

Today, I studied **File Handling** in Java, which allows programs to read from and write to files. Java provides the **java.io** package for handling files, enabling persistent data storage beyond program execution.

---

### 1. File Handling Overview

File handling is used to perform operations like creating, reading, writing, and deleting files.

**Common Classes Used:**

- `File` – represents a file or directory
- `FileReader` and `BufferedReader` – used for reading data
- `FileWriter` and `BufferedWriter` – used for writing data

---

### 2. Writing to a File:

```
import java.io.*;
public class WriteExample {
    public static void main(String[] args) {
        try {
            BufferedWriter bw = new BufferedWriter(new
FileWriter("output.txt"));
            bw.write("Hello, Java File Handling!");
            bw.close();
```

```
            System.out.println("Data written successfully.");
        } catch (IOException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

**Output:**

```
Data written successfully.
```

---

**3. Reading from a File:**

```
import java.io.*;
public class ReadExample {
    public static void main(String[] args) {
        try {
            BufferedReader br = new BufferedReader(new
FileReader("output.txt"));
            String line;
            while ((line = br.readLine()) != null) {
                System.out.println(line);
            }
            br.close();
        } catch (IOException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

---

**4. Stream Types**

- **Byte Streams:** Used for binary data (e.g., `FileInputStream`, `FileOutputStream`)

- **Character Streams:** Used for text data (e.g., `FileReader`, `FileWriter`)

---

## Learning Outcome:

Learned how to create, read, and write files using streams in Java.
Understood the difference between byte and character streams.
Gained the ability to handle file-related exceptions and work with external data storage effectively.