**LockedMe**

This document contains sections for:

* Project and developer details
* Sprints planned and the tasks achieved in them
* Link to the GitHub repository to verify the project completion
* Core concepts used in the project
* Flowchart of the program.
* Process and logic of the program
* Conclusion on enhancing the application and defining the USPs (Unique Selling Points)

The code for this project is hosted at <https://github.com/minal4991/LockedMe.git>.

The project is developed by Minaldeep Cheema.

**Sprint Planning**

The project is planned to be completed in 3 sprints of 2 days each. Tasks assumed to be completed in the sprint are:

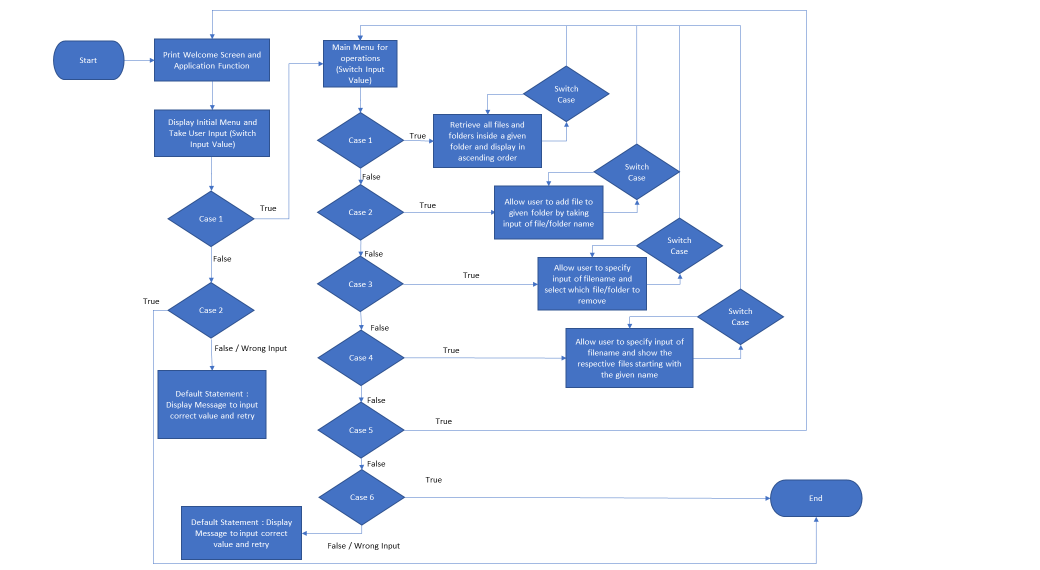
* **Sprint 1:**
  + Creating the flow of the application
  + Initializing git repository to track changes.
* **Sprint 2:**
  + Develop program in Java meeting project requirements.
  + Testing program for exceptions and breakdowns.
* **Sprint 3:**
  + Pushing code to GitHub.
  + Develop specification document.

**Core Concepts**

The concepts used in the development of this project are:

* Collections framework
* File Handling
* Sorting
* Flow Control
* Recursion
* Exception Handling

**Flow of the Application**



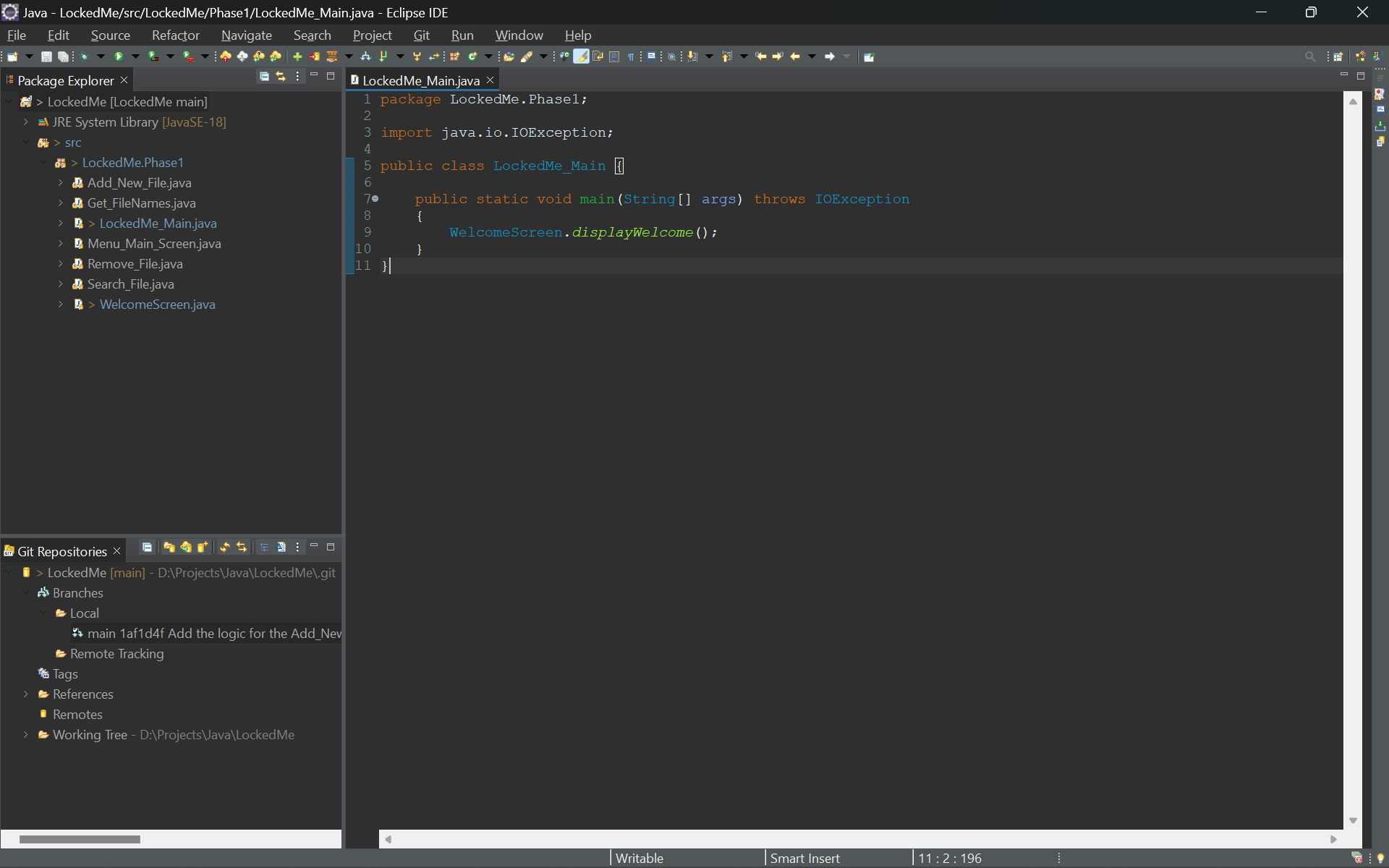
**Development Process**

**Step 1: Creating a new project in Eclipse**

* Open Eclipse
* Create a New Java Project named **LockedMe**.
* Initialize local Git repository

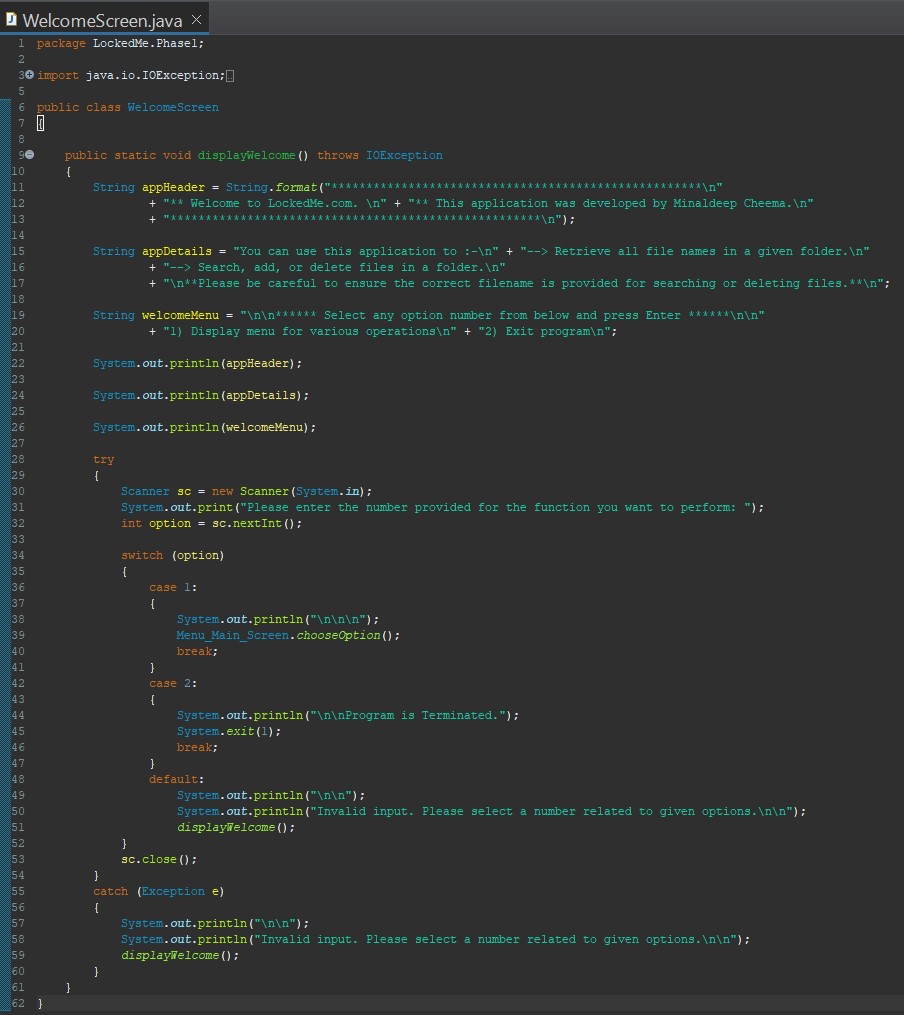
**Step 2: Writing a program in Java for the entry point of the application (LockedMe\_Main.java)**

* Create a new class **LockedMe\_Main** with the main method to start the program.
* Write the logic for this class.



**Step 3: Writing a program in Java for the Welcome Display of the application (WelcomeScreen.java)**

* Create a new class **WelcomeScreen.java** without the main method.
* Write the logic for this class.



* Output of the logic.

Text

Description automatically generated

**Step 4: Writing a program in Java for the Main Menu Display of the application (Menu\_Main\_Screen.java)**

* Create a new class **Menu\_Main\_Screen.java** without the main method.
* Write the logic for this class.

Text

Description automatically generated Text

Description automatically generated

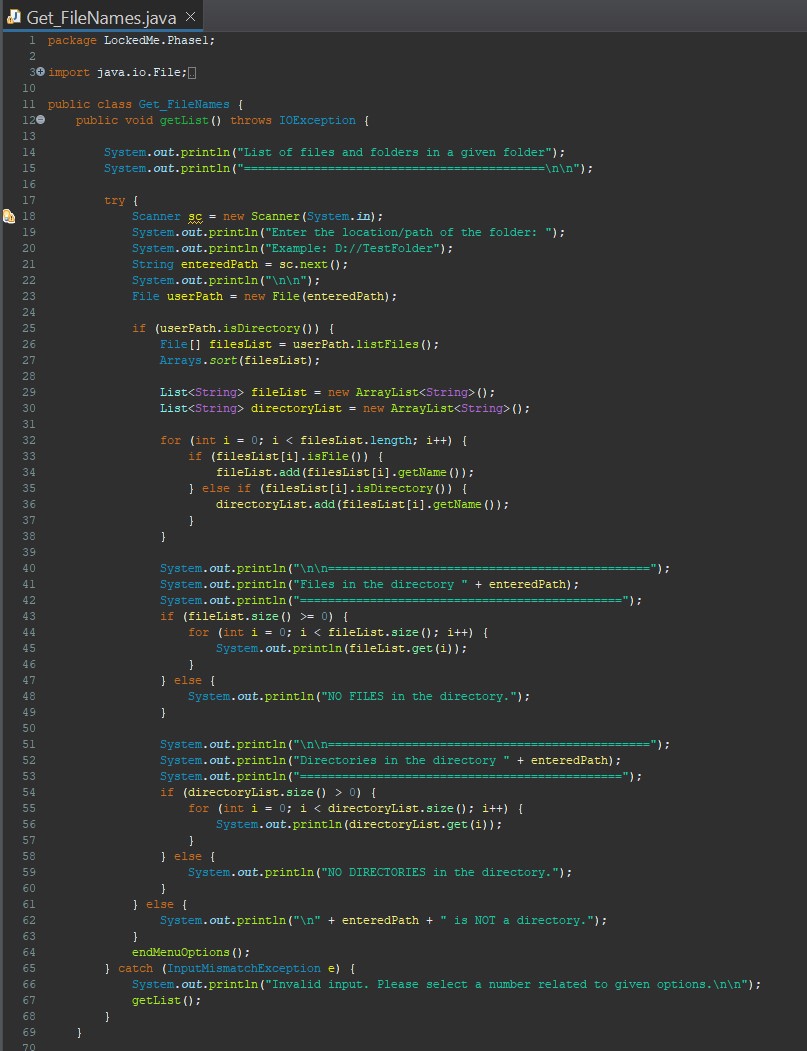
* Output of the logic

Text

Description automatically generated

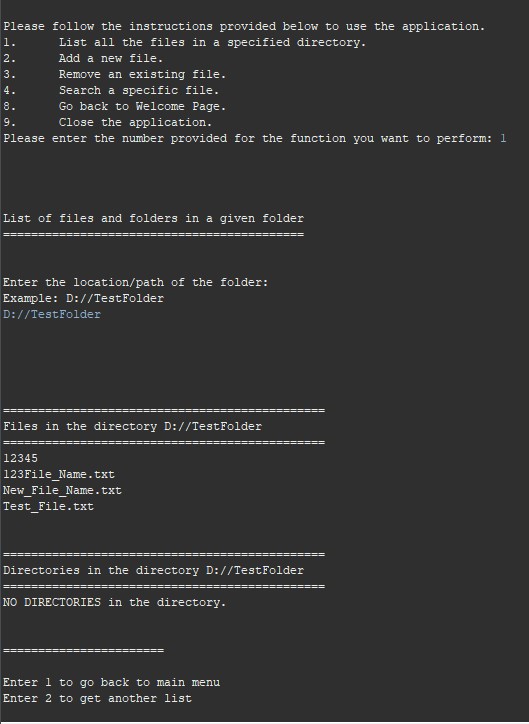
**Step 5: Writing a program in Java to Display the List of Files and Directories in a given directory (Get\_FileNames.java)**

* Create a new class **Get\_FileNames.java** without the main method.
* Write the logic for this class.

 Text

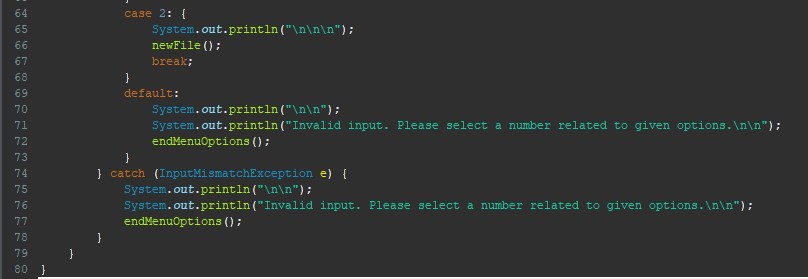
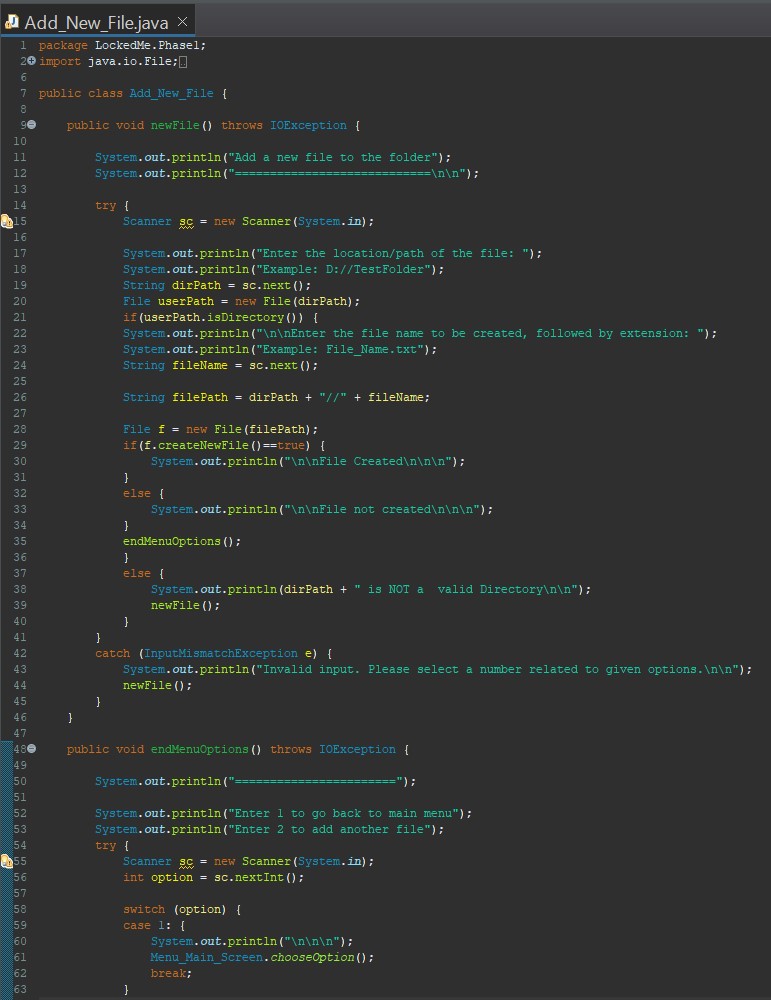
Description automatically generated

* Output of the Logic



**Step 6: Writing a program in Java to Add Files in a given directory (Add\_New\_File.java)**

* Create a new class **Add\_New\_File.java** without the main method.
* Write the logic for this class.



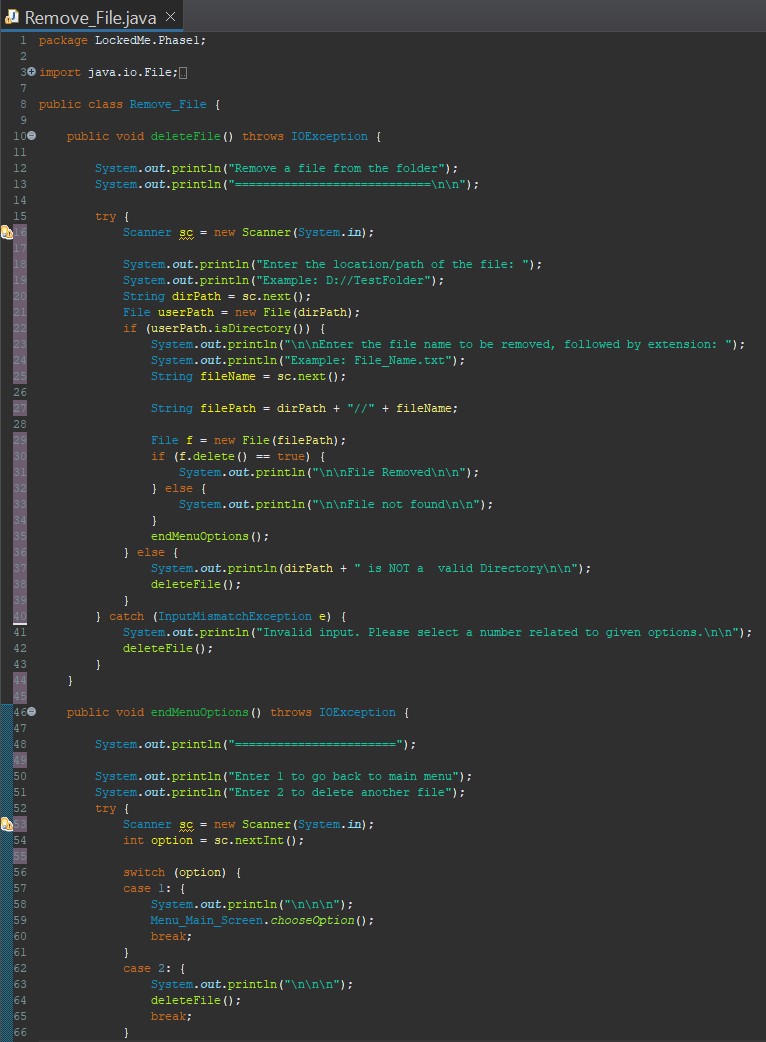
* Output of the logic.

Text

Description automatically generated

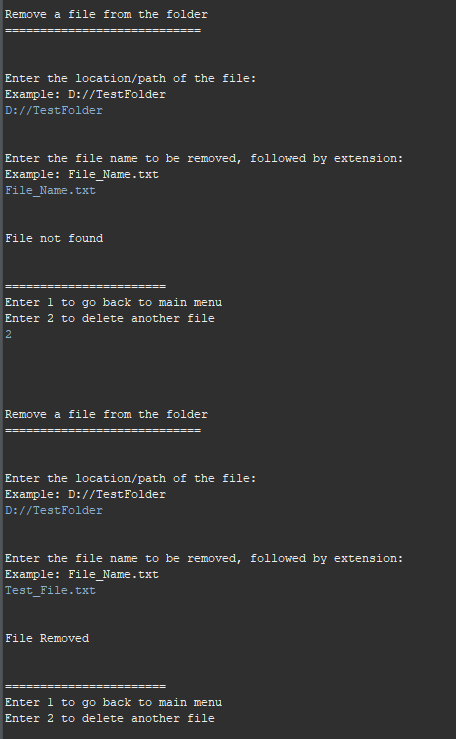
**Step 7: Writing a program in Java to Remove Files in a given directory (Remove\_File.java)**

* Create a new class **Remove\_File.java** without the main method.
* Write the logic for this class.

Text

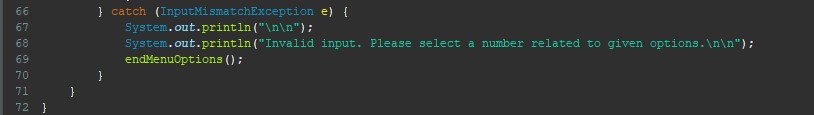
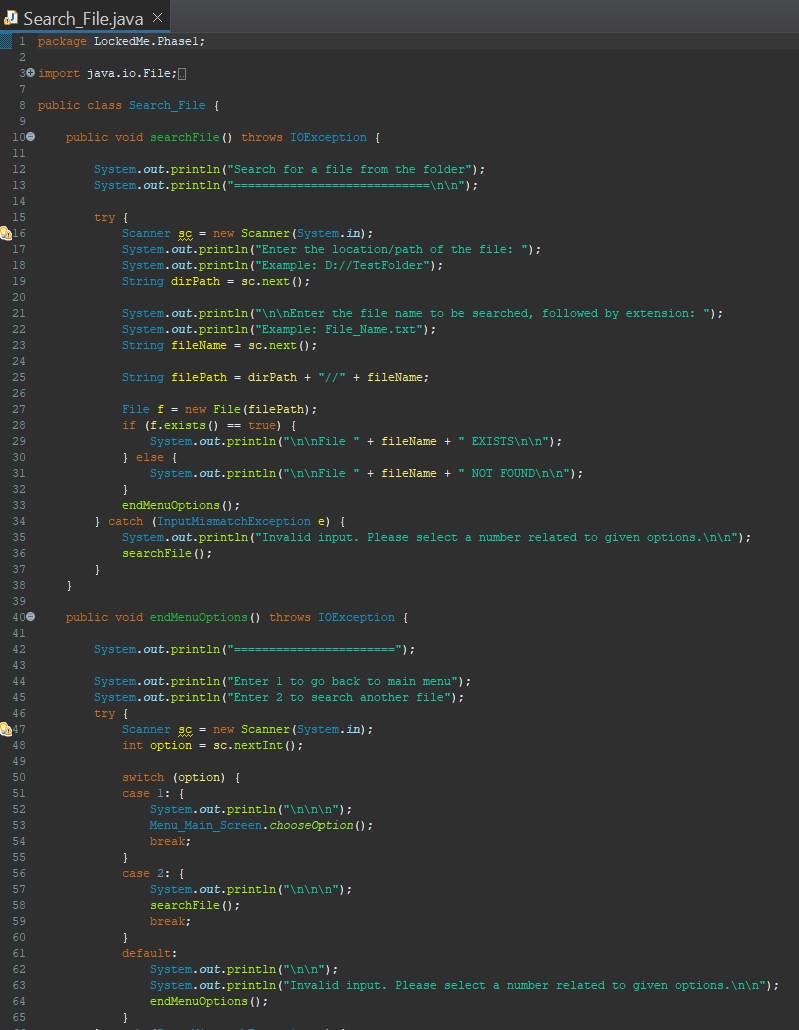
Description automatically generated

* Output of the logic.

****

**Step 7: Writing a program in Java to Search Files in a given directory (Search\_File.java)**

* Create a new class **Search\_File.java** without the main method.
* Write the logic for this class.



* Output of the logic

Text

Description automatically generated

## 

**Step 8: Linking the local Git repository to GitHub**

* Create a new Repository on GitHub named LockedMe.
* Do not initiate the repository.
* On GitBash, change the current directory to the root directory of the project.
* Push the repository to GitHub.

**USPs**

1. The program is able to handle all the exceptions, even if the user enters wrong or invalid input.
2. The program can be terminated upon proper selection of options.
3. The application can take any given valid / authorized directory in the system to work upon.
4. The application can be used to perform the CRUD (Create, Read, Update, Delete) file operations.
5. The coding is loosely coupled to support addition of further modules

**Further Enhancements**

* User Authorization: Conditions to check if user is allowed to delete the file or add the file at the specific locations.
* Confirmation Step: Asking user to verify if they really want to delete the selected directory if it’s not empty.
* Modification or addition of data to the files being created or already created.