**Project**

**LockedMe.com-Virtual Key for Repositories**

# **Contents of Document:**

* Sprint planning and Task completion
* Flow of the Application
* Core concepts used in project
* Describing the product capabilities, appearance and user interactions
* Unique Selling Points of the Application
* Conclusions

# **Developer details**

The project is developed by **Himanshi Singh** from **FSD Java Developers (All engineers batch)** and the code for this project is hosted at [himanshisinghh/LockedMEmain.com (github.com)](https://github.com/himanshisinghh/LockedMEmain.com)

# **Sprints planning and Task completion**

The project is planned to be completed in 3 sprints. The tasks are expected to be completed in the sprints are:

* **Sprint 1**

1. Creating the flow of the application

2. Initializing git repositories to track changes in task as development progresses.

* **Sprint 2**

1. Writing the Java program to achieve the project requirements.

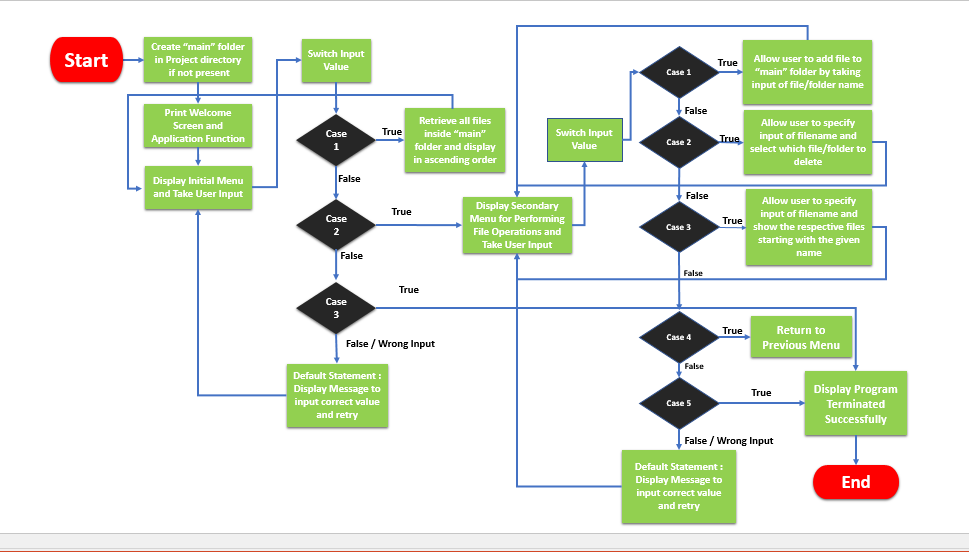
2. Testing the Java program with various User inputs

* **Sprint 3**

1. Pushing code to GitHub
2. Creating the Specification document highlighting the Application Capabilities, Appearance, and User interactions

# Flow of the Application

**This includes flowcharts & Algorithm:**





# Core concepts used in this project

* File Handling
* Collections framework
* Searching & Sorting
* Exceptions Handling
* Flow Control statements & Recursions
* Data Structures: Arrays, Strings

# Describing the product capabilities, appearance and user interactions

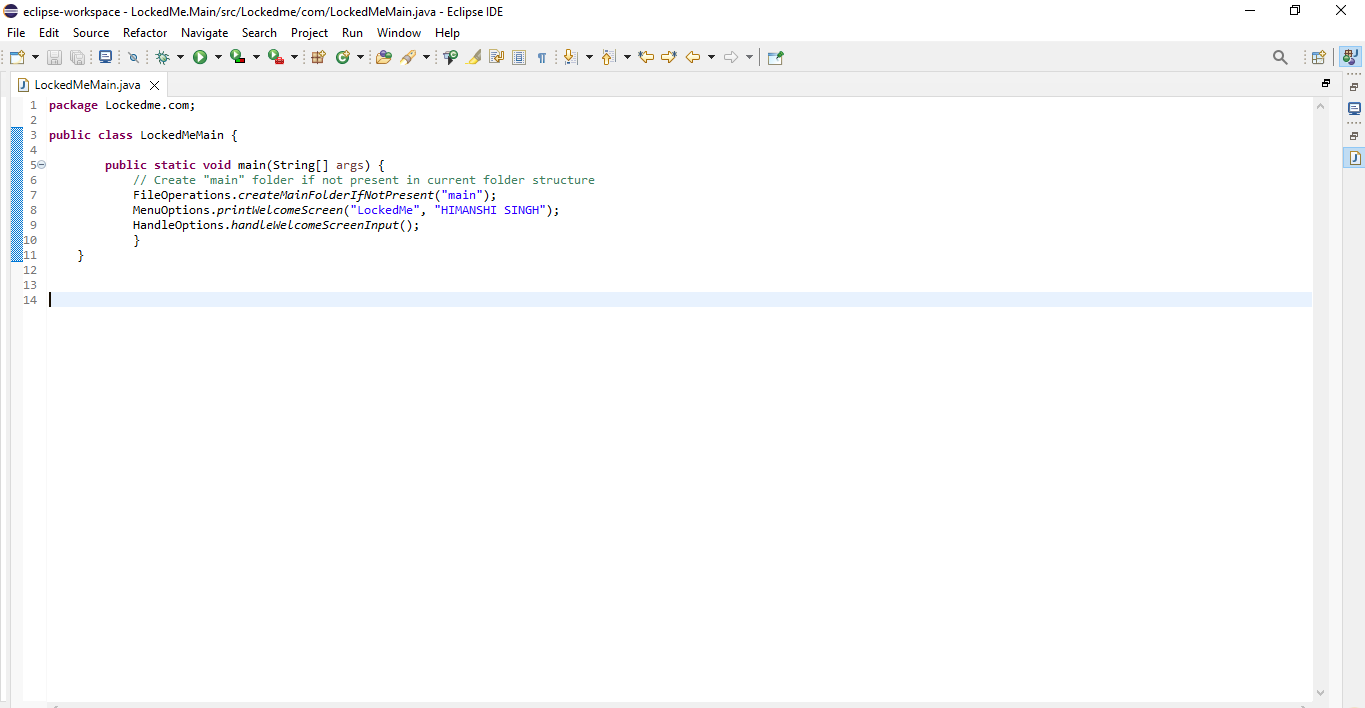
To demonstrate the products capabilities, appearance and user interactions, following are the sub-sections configured for the project:

1. **Creating the project code in Eclipse**
2. **Writing a program in Java for entry point the application (LockedMe.java)**
3. **Writing a program in Java to display list options available for the user (ListOptions.java)**
4. **Writing a program in Java to handle List options selected by user (HandleOptions.java)**
5. **Writing a program in Java to perform the File handling operations as specified by user (FileHandlingOps.java)**
6. **Pushing the code to GitHub repository**

## Step 1: Creating a new project in Eclipse

* **Open Eclipse**
* **Go to file--> New--> Project--> Java Project--> Next.**
* **Type in any project name and click on “Finish”.**
* **Select your project and go to file 🡪 New🡪 Class.**
* **Enter LockedMe as class name and do check the checkbox “public static void main (String [] args)”, and click on “Finish**”.

## Step 2: Writing a program in Java for the entry point of application (start)

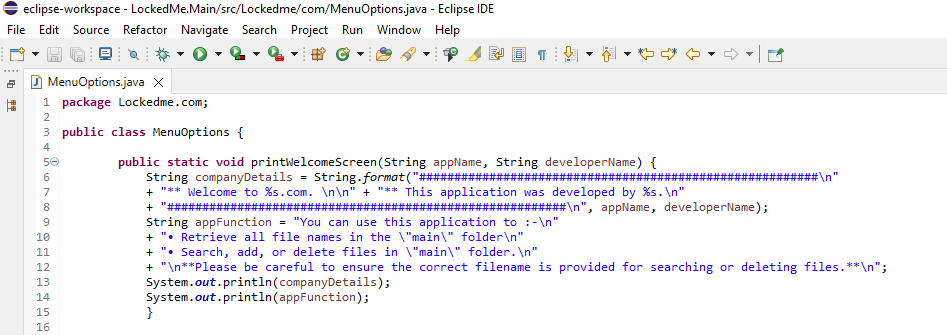


## Step 3: Writing a program in Java to display List options available for the user (MenuOptions.java)

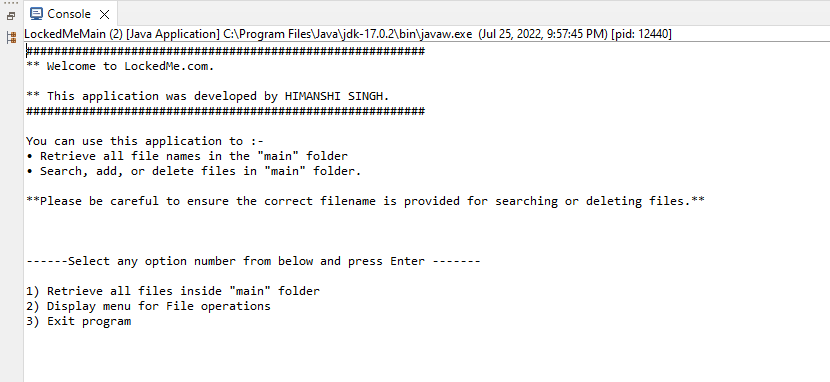
* **Select your project and go to File🡪 New 🡪 Class**
* **Enter MenuOptions as class name and click “Finish.”**
* **MenuOptions include methods for:**

1. **Displaying Welcome Screen (Welcome to LockedMe.com)**
2. **Displaying Initial Menu**
3. **Displaying Secondary Menu for File Operations available**

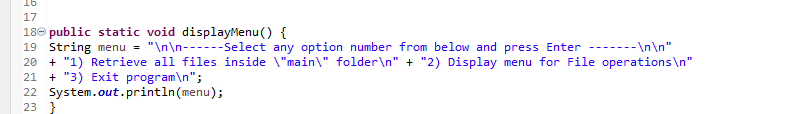
**Step 3(i): Writing method to display Welcome Screen.**



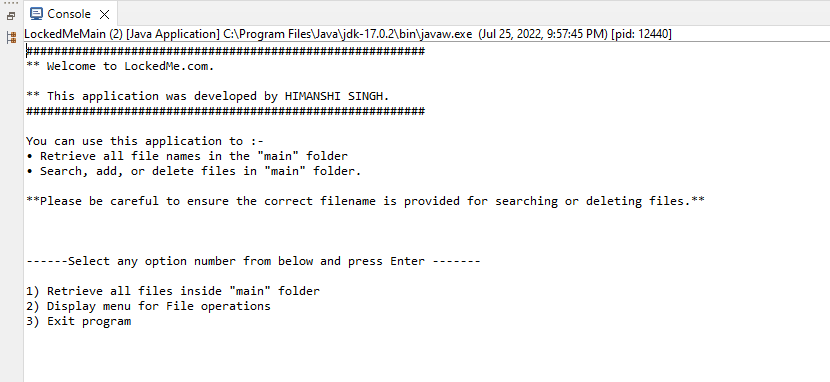
**Output:**

****

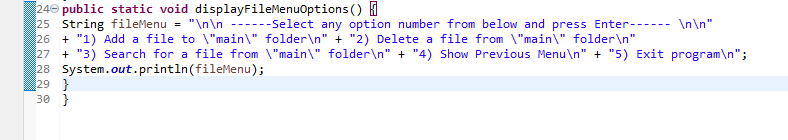
**Step 3(ii): Writing method to display Initial Menu.**



**Output:**

****

Step 3(iii) writing method to display Initial Menu



**Output:**

## Step 4: Writing a program in Java to handle List options selected by user (HandleOptions.java)

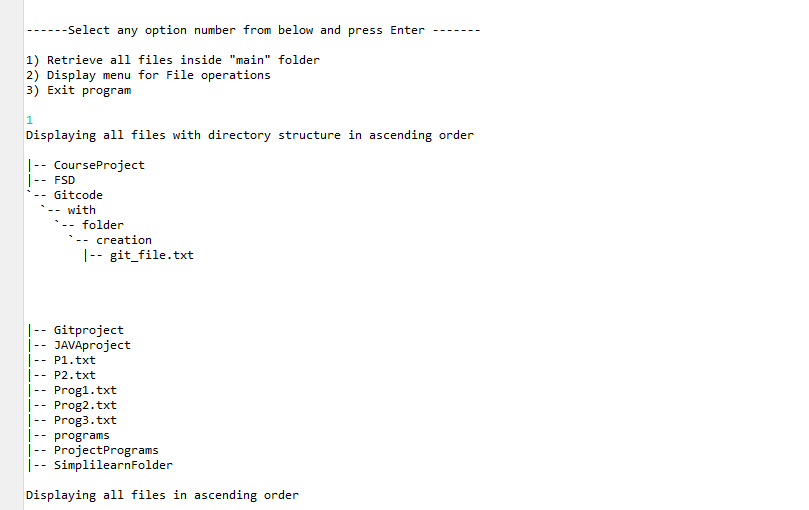
* **Select your project and go to File -> New -> Class.**
* **Enter HandleOptions in class name and click on “Finish.”**
* **HandleOptions consists methods for -:**

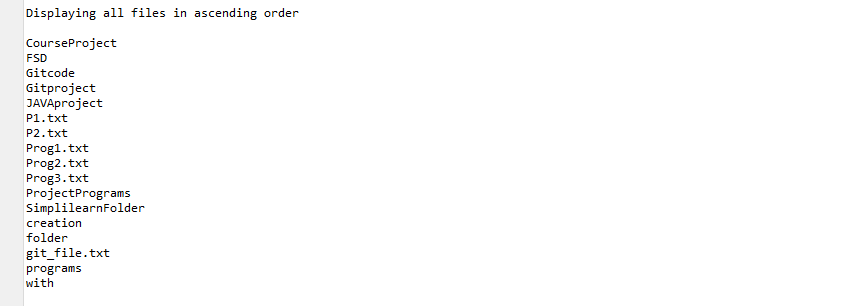
1. Handling input selected by user in initial Menu
2. Handling input selected by user in secondary Menu for File Operations

**Step 4 (i): Writing method to handle user input in initial Menu**



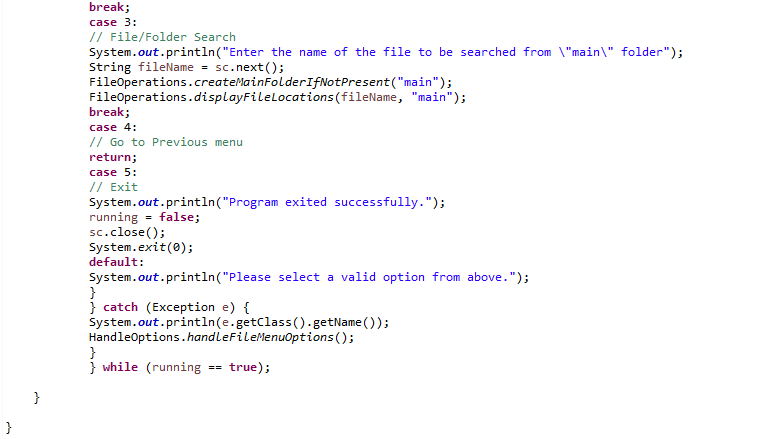
**Output:**

****

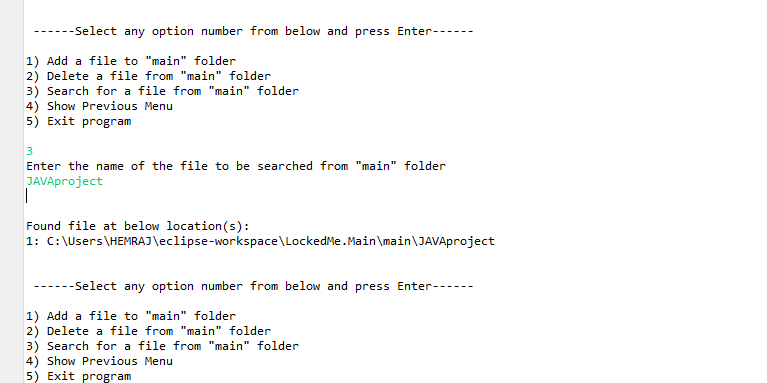
****

**Step 4(ii): Writing method to handle user input in Secondary Menu for File Operations**





**Output:**

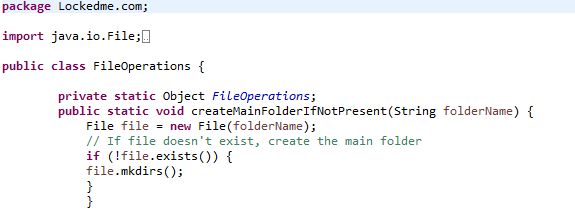
****

## Step 5: Writing a program in Java to perform the File Handling operations as specified by user ()

* Select your project and go to File -> New -> Class.
* Enter **FileOperations** in class name and click on “Finish.”
* **FileOperations** consists methods for -:

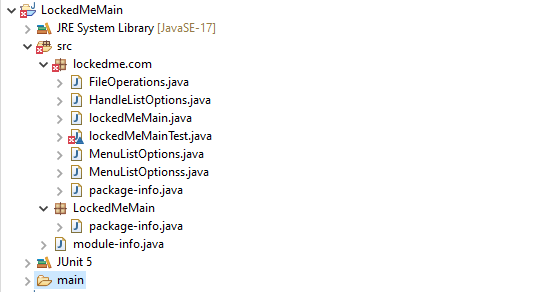
1. Creating “main” folder in project if it’s not already present
2. Displaying all files in “main” folder in ascending order and also with directory structure.
3. Creating a file/folder as specified by user input.
4. Search files as specified by user input in “main” folder and it’s subfolders.
5. Deleting a file/folder from “main” folder

**Step 5(i): Writing method to create “main” folder in project if it’s not present**

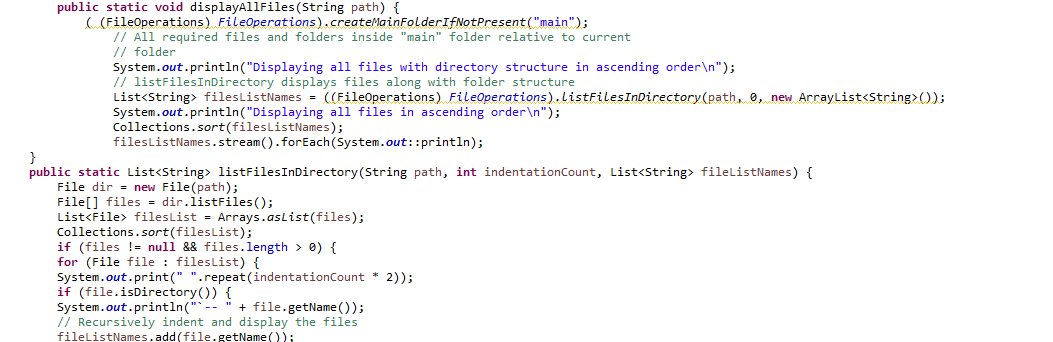


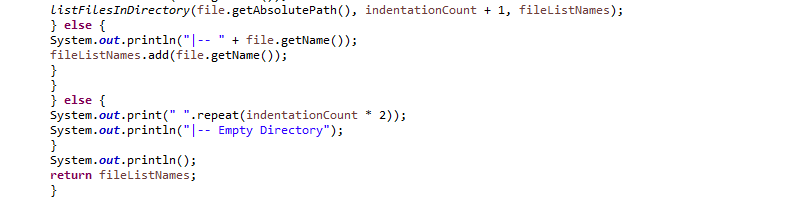
**Output:**

**-**

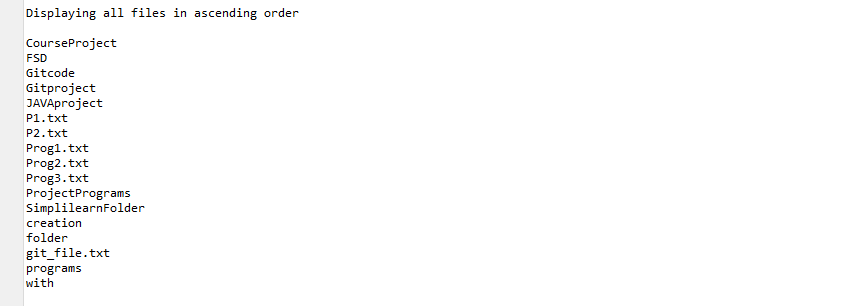
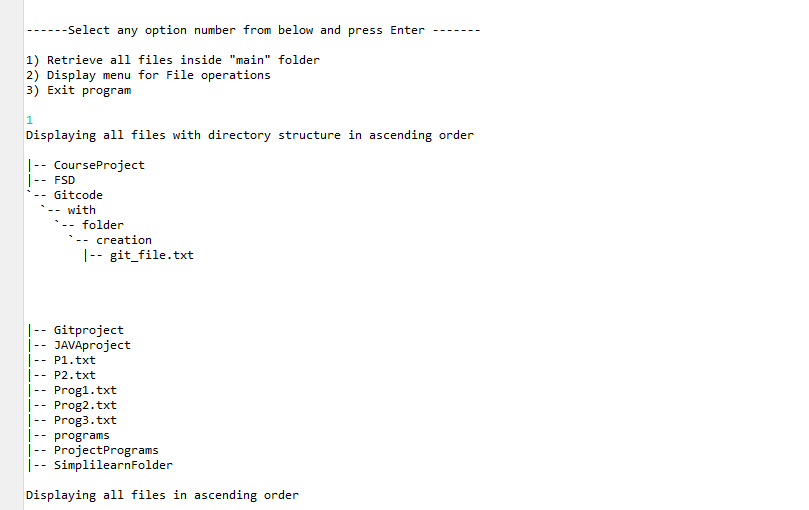
****

**Step 5(ii): Writing method to display all files in “main” folder in ascending order and also with directory structure. (“`--" represents a directory. “|--” represents a file.)**

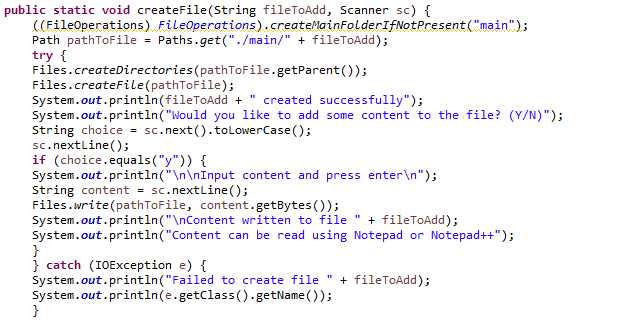


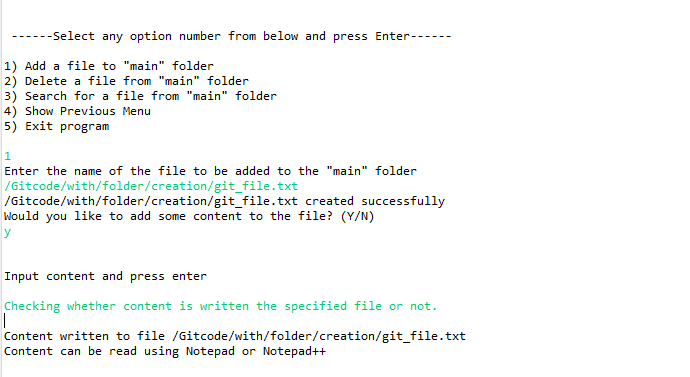


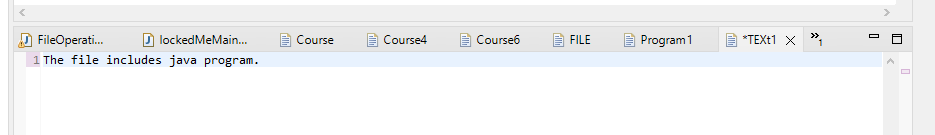
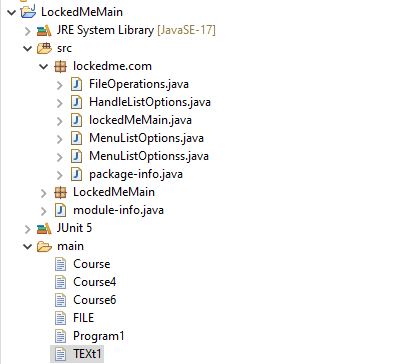
**Output:**

****

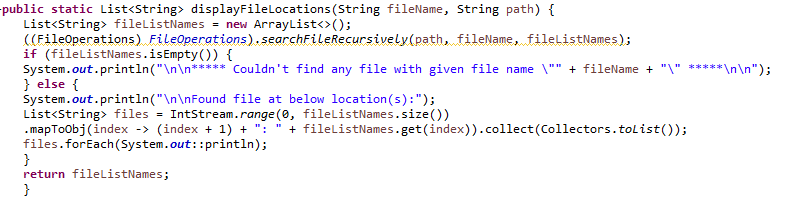
**Step 5(iii): Writing method to create a file/folder as specified by user input.**

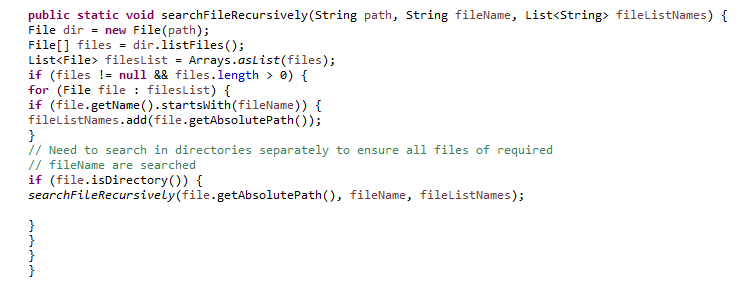


**Output:**

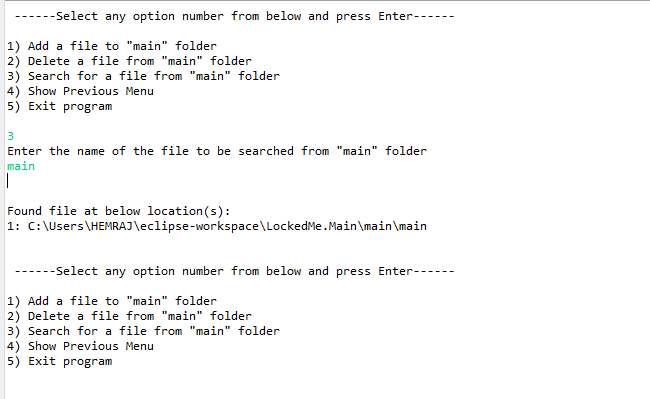
****

**Step 5(iv): Writing method to search for all files as specified by user input in “main” folder and its subfolders.**

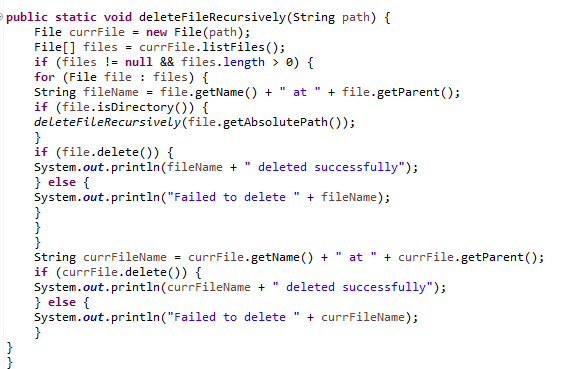


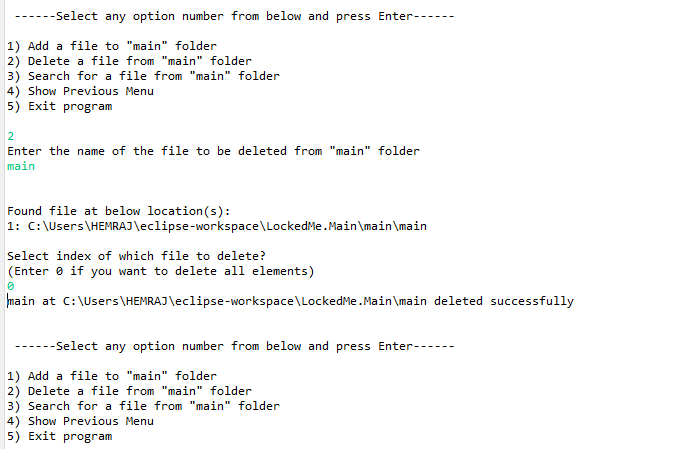


**Output:**

****

**Step 5(v): Writing method to delete file/folder specified by user input in “main” folder and its subfolders. It uses the searchFilesRecursively method and prompts user to specify which index to delete. If folder selected, all its child files and folder will be deleted recursively. If user wants to delete all the files specified after the search, they can input value 0**.



**Output:**

## Step 6: Pushing the code to GitHub repository

* Open the command prompt and navigate to the folder where you have created your files.

Cd< folder path>

* Initialize repository using following command:

**git init**

* Add all the files to your git repository using the following command:

**git add .**

* Commit the changes using the following command:

**git commit . –m <commit message>**

* Push the files to the folder you initially created using thefollowing command:

**git push –u origin master**

# Unique Selling Points of the Application

1. The application is designed to keep on running and taking user inputs even after exceptions occur. To terminate the application, must appropriate options needs to be selected.

2. The application can take any file/folder name as an input and even if the user wants create nested folder structure, user can just specify the relative path and the application takes care of creating the required folder structure.

3. User is also provide the option to write content if they want into the newly created file.

4. The application doesn’t restrict user to specify the exact filename to search/delete file/folder. They can specify the starting input, and the program searches all files/folder starting with the value and displays it. The user is then provided the option to select all files or to select a specific index to delete.

5. The application also allows user to delete folders which are not empty.

6. The user is able to seamlessly switch between options or return to previous menu even after any required operation like adding, searching, deleting or retrieving of files is performed.

7. When the option to retrieve files in ascending order is selected, user is displayed with two options of viewing the files.

7.1. Ascending order of folders first which have files sorted in them,

7.2. Ascending order of all files and folders inside the “main” folder.

8. The application is designed with modularity in mind. Even if one wants to update the path, they can change it through the source code. Application has been developed keeping in mind that there should be very less “hardcoding” of data.

## Conclusions

Further enhancements to the application can be made which may include:

* Conditions to check if user is allowed to delete the file or add the file at the specific locations.
* Asking user to verify if they really want to delete the selected directory if it’s not empty.
* Retrieving files/folders by different criteria like Last Modified, Type, etc.
* Allowing user to append data to the file.