Project Name: Virtual Key for Your Repositories

Developer Details:

NAME: Pallavi Thokala

EMAIL ID: pallavi.sivakumar03@gmail.com

GITHUB: pallavi010416

Sprints planned and the tasks achieved:

The project is planned to be completed in 1 sprint. Tasks assumed to be completed in the sprint are:

- Creating the flow of the application
- Initializing git repository to track changes as development progresses.
- Writing the Java program to fulfill the requirements of the project.
- Testing the Java program with different kinds of User input
- Pushing code to GitHub.
- Creating this specification document highlighting application capabilities, appearance, and user interactions.

Algorithm:

Step 1: Create a new Project in Eclipse

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

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

Step 3.1: Writing method to display Welcome Screen

Step 3.2: Writing method to display Initial Menu

Step 3.3: Writing method to display Secondary Menu for File Operations

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

Step 4.1: Writing method to handle user input in initial Menu

Step 4.2: Writing method to handle user input in Secondary Menu for File Operations

Step 5: Writing a program in Java to perform the File operations as specified by user (**FileOperations.java**)

Step 5.1: Writing method to create “main” folder in project if it’s not present

Step 5.2: Writing method to display all files in “main” folder in ascending order and also with directory structure. (“`--" represents a directory. “|--” represents a file.)

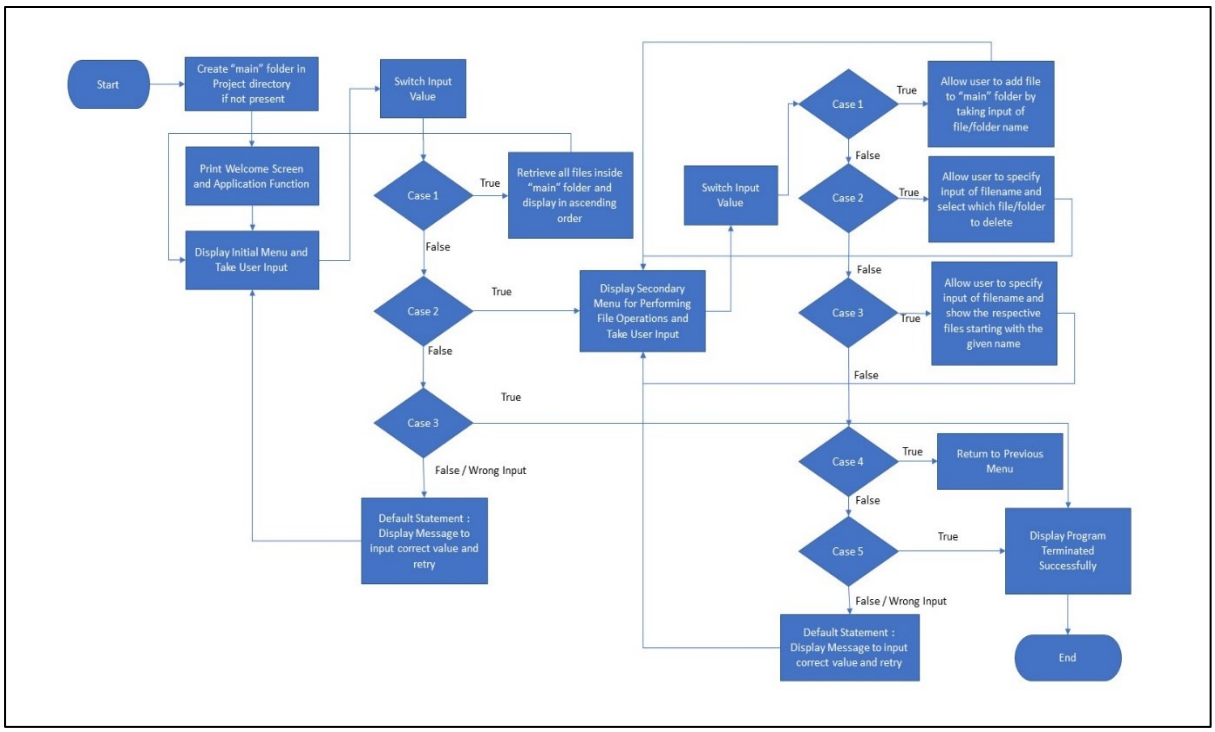
Step 5.3: Writing method to create a file/folder as specified by user input.

Step 5.4: Writing method to search for all files as specified by user input in “main” folder and its subfolders.

Step 5.5: 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 it’s 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.

Step 6: Pushing the Code to Git Repository.

Flowchat:



Core Concepts Used in the Project:

Collections framework, File Handling, Sorting, Flow Control, Recursion, Exception Handling, Streams API

Git Hub Link:

https://github.com/pallavi010416/MainProject-Phase1-Simplilearn-

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, appropriate option needs to be selected.

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

3. User is also provided 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.

Conclusion:

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.