

## Main Program:



The screenshot shows an IDE with several tabs: BugFix.java, LockedMeMain.java, FileOperations.java, HandleOptions.java, and MenuOptions.java. The LockedMeMain.java tab is active, displaying the following code:

```
1
2
3 public class LockedMeMain {
4     public static void main(String[] args) {
5
6         // Create "main" folder if not present in current folder structure
7         FileOperations.createMainFolderIfNotPresent("main");
8
9         MenuOptions.printWelcomeScreen("LockedMe", "Padmaja Yadav");
10
11        HandleOptions.handleWelcomeScreenInput();
12    }
13
14
15 }
```

The right sidebar shows the Outline view with a tree structure: LockedMeMain > main.

## File Operation:

```
1
2 import java.io.File;
3 import java.io.IOException;
4 import java.nio.file.Files;
5 import java.nio.file.Path;
6 import java.nio.file.Paths;
7 import java.util.ArrayList;
8 import java.util.Arrays;
9 import java.util.Collections;
10 import java.util.List;
11 import java.util.Scanner;
12 import java.util.stream.Collectors;
13 import java.util.stream.IntStream;
14
15 public class FileOperations {
16
17     public static void createMainFolderIfNotPresent(String folderName) {
18         File file = new File(folderName);
19
20         // If file doesn't exist, create the main folder
21         if (!file.exists()) {
22             file.mkdirs();
23         }
24     }
25
26     public static void displayAllFiles(String path) {
27         FileOperations.createMainFolderIfNotPresent("main");
28         // All required files and folders inside "main" folder relative to current
29         // folder
30         System.out.println("Displaying all files with directory structure in ascending order\n");
31
32         // listFilesInDirectory displays files along with folder structure
33         List<String> fileListNames = FileOperations.listFilesInDirectory(path, 0, new ArrayList<String>());
34
35         System.out.println("Displaying all files in ascending order\n");
36         Collections.sort(fileListNames);
37
38         fileListNames.stream().forEach(System.out::println);
39     }
40
41     public static List<String> listFilesInDirectory(String path, int indentationCount, List<String> fileListNames) {
42         File dir = new File(path);
43         File[] files = dir.listFiles();
44         List<File> fileList = Arrays.asList(files);
45
46         Collections.sort(fileList);
47
48         if (files != null && files.length > 0) {
49             for (File file : fileList) {
50
51                 System.out.print(" ".repeat(indentationCount * 2));
```

```
50      System.out.print(" ".repeat(indentationCount * 2));
51
52      if (file.isDirectory()) {
53          System.out.println("|-- " + file.getName());
54
55          // Recursively indent and display the files
56          fileListNames.add(file.getName());
57          listFilesInDirectory(file.getAbsolutePath(), indentationCount + 1, fileListNames);
58      } else {
59          System.out.println("|-- " + file.getName());
60          fileListNames.add(file.getName());
61      }
62  }
63  }
64  } else {
65      System.out.print(" ".repeat(indentationCount * 2));
66      System.out.println("|-- Empty Directory");
67  }
68  System.out.println();
69  return fileListNames;
70  }
71
72  public static void createFile(String fileToAdd, Scanner sc) {
73      FileOperations.createMainFolderIfNotPresent("main");
74      Path pathToFile = Paths.get("./main/" + fileToAdd);
75      try {
76          Files.createDirectories(pathToFile.getParent());
77          Files.createFile(pathToFile);
78          System.out.println(fileToAdd + " created successfully");
79
80          System.out.println("Would you like to add some content to the file? (Y/N)");
81          String choice = sc.next().toLowerCase();
82
83          sc.nextLine();
84          if (choice.equals("y")) {
85              System.out.println("\n\nInput content and press enter\n");
86              String content = sc.nextLine();
87              Files.write(pathToFile, content.getBytes());
88              System.out.println("\nContent written to file " + fileToAdd);
89              System.out.println("Content can be read using Notepad or Notepad++");
90          }
91      } catch (IOException e) {
92          System.out.println("Failed to create file " + fileToAdd);
93          System.out.println(e.getClass().getName());
94      }
95  }
96  }
97
98  public static List<String> displayFileLocations(String fileName, String path) {
99      // Implementation of displayFileLocations method
100  }
```

```

BugFix.java  LockedMeMain.java  FileOperations.java ×  HandleOptions.java  MenuOptions.java
115
116 public static void searchFileRecursively(String path, String fileName, List<String> fileListNames) {
117     File dir = new File(path);
118     File[] files = dir.listFiles();
119     List<File> fileList = Arrays.asList(files);
120
121     if (files != null && files.length > 0) {
122         for (File file : fileList) {
123
124             if (file.getName().startsWith(fileName)) {
125                 fileListNames.add(file.getAbsolutePath());
126             }
127
128             // Need to search in directories separately to ensure all files of required
129             // fileName are searched
130             if (file.isDirectory()) {
131                 searchFileRecursively(file.getAbsolutePath(), fileName, fileListNames);
132             }
133         }
134     }
135 }
136
137 public static void deleteFileRecursively(String path) {
138
139     File currFile = new File(path);
140     File[] files = currFile.listFiles();
141
142     if (files != null && files.length > 0) {
143         for (File file : files) {
144
145             String fileName = file.getName() + " at " + file.getParent();
146             if (file.isDirectory()) {
147                 deleteFileRecursively(file.getAbsolutePath());
148             }
149
150             if (file.delete()) {
151                 System.out.println(fileName + " deleted successfully");
152             } else {
153                 System.out.println("Failed to delete " + fileName);
154             }
155         }
156     }
157
158     String currFileName = currFile.getName() + " at " + currFile.getParent();
159     if (currFile.delete()) {
160         System.out.println(currFileName + " deleted successfully");
161     } else {
162         System.out.println("Failed to delete " + currFileName);
163     }
164 }
165 }
166

```

## Handle Options:

BugFix.java LockedMeMain.java FileOperations.java HandleOptions.java × MenuOptions.java

```
1
2 import java.util.List;
3 import java.util.Scanner;
4
5 public class HandleOptions {
6     public static void handleWelcomeScreenInput() {
7         boolean running = true;
8         Scanner sc = new Scanner(System.in);
9         do {
10             try {
11                 MenuOptions.displayMenu();
12                 int input = sc.nextInt();
13
14                 switch (input) {
15                     case 1:
16                         FileOperations.displayAllFiles("main");
17                         break;
18                     case 2:
19                         HandleOptions.handleFileMenuOptions();
20                         break;
21                     case 3:
22                         System.out.println("Program exited successfully.");
23                         running = false;
24                         sc.close();
25                         System.exit(0);
26                         break;
27                     default:
28                         System.out.println("Please select a valid option from above.");
29                 }
30             } catch (Exception e) {
31                 System.out.println(e.getClass().getName());
32                 handleWelcomeScreenInput();
33             }
34         } while (running == true);
35     }
36
37     public static void handleFileMenuOptions() {
38         boolean running = true;
39         Scanner sc = new Scanner(System.in);
40         do {
41             try {
42                 MenuOptions.displayFileMenuOptions();
43                 FileOperations.createMainFolderIfNotPresent("main");
44
45                 int input = sc.nextInt();
46                 switch (input) {
47                     case 1:
48                         // File Add
49                         System.out.println("Enter the name of the file to be added to the \"main\" folder");
50                         String fileToAdd = sc.next();
51                         FileOperations.createFile(fileToAdd);
52                 }
53             }
54         }
55     }
56 }
```

```

BugFix.java  LockedMeMain.java  FileOperations.java  HandleOptions.java  MenuOptions.java
50     String fileToDelete = sc.next();
51
52     FileOperations.createMainFolderIfNotPresent("main");
53     List<String> filesToDelete = FileOperations.displayFileLocations(fileToDelete, "main");
54
55     String deletionPrompt = "\nSelect index of which file to delete?"
56         + "\n(Enter 0 if you want to delete all elements)";
57     System.out.println(deletionPrompt);
58
59     int idx = sc.nextInt();
60
61     if (idx != 0) {
62         FileOperations.deleteFileRecursively(filesToDelete.get(idx - 1));
63     } else {
64         // If idx == 0, delete all files displayed for the name
65         for (String path : filesToDelete) {
66             FileOperations.deleteFileRecursively(path);
67         }
68     }
69
70     break;
71
72     case 3:
73         // File/Folder Search
74         System.out.println("Enter the name of the file to be searched from \"main\" folder");
75         String fileName = sc.next();
76
77         FileOperations.createMainFolderIfNotPresent("main");
78         FileOperations.displayFileLocations(fileName, "main");
79
80     break;
81
82     case 4:
83         // Go to Previous menu
84         return;
85
86     case 5:
87         // Exit
88         System.out.println("Program exited successfully.");
89         running = false;
90         sc.close();
91         System.exit(0);
92
93     default:
94         System.out.println("Please select a valid option from above.");
95     }
96 } catch (Exception e) {
97     System.out.println(e.getClass().getName());
98     handleFileMenuOptions();
99 }
100 } while (running == true);
101 }

```

## Menu Options:

```
1 |
2 public class MenuOptions {
3     public static void printWelcomeScreen(String appName, String developerName) {
4         String companyDetails = String.format("*****\n"
5             + "Welcome to %s.com. \n" + "This application was developed by %s.\n"
6             + "*****\n", appName, developerName);
7         String appFunction = "You can use this application to :-\n"
8             + "1) Retrieve all file names in the \"main\" folder\n"
9             + "2) Search, add, or delete files in \"main\" folder\n"
10            + "3) Please be careful to ensure the correct filename is provided for searching or deleting files.\n";
11         System.out.println(companyDetails);
12         System.out.println(appFunction);
13     }
14 }
15
16 public static void displayMenu() {
17     String menu = "\n\n*** Select any option number from below and press Enter ***\n\n"
18         + "1) Retrieve all files inside \"main\" folder\n" + "2) Display menu for File operations\n"
19         + "3) Exit program\n";
20     System.out.println(menu);
21 }
22
23
24 public static void displayFileMenuOptions() {
25     String fileMenu = "\n\n*** Select any option number from below and press Enter ***\n\n"
26         + "1) Add a file to \"main\" folder\n" + "2) Delete a file from \"main\" folder\n"
27         + "3) Search for a file from \"main\" folder\n" + "4) Show Previous Menu\n" + "5) Exit program\n";
28     System.out.println(fileMenu);
29 }
30 }
31
32 }
```

## **Output:**

\*\*\* Select any option number from below and press Enter \*\*\*

- 1) Add a file to "main" folder
- 2) Delete a file from "main" folder
- 3) Search for a file from "main" folder
- 4) Show Previous Menu
- 5) Exit program

1

Enter the name of the file to be added to the "main" folder

padmajayadav

padmajayadav created successfully

Would you like to add some content to the file? (Y/N)

y

Input content and press enter

hai

Content written to file padmajayadav

Content can be read using Notepad or Notepad++

\*\*\* Select any option number from below and press Enter \*\*\*

- 1) Add a file to "main" folder
- 2) Delete a file from "main" folder
- 3) Search for a file from "main" folder
- 4) Show Previous Menu
- 5) Exit program

5

Program exited successfully.

<