```
import java.io.File;
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Collections;
import java.util.List;
import java.util.Scanner;
import java.util.function.IntPredicate;
import java.util.stream.Collectors;
import java.util.stream.IntStream;

public class FileOperations {

	public static void createFolderIfFolderNotAvailable(String folderName) {
		File file = new File(folderName);



		if (!file.exists()) {
			file.mkdirs();
		}
	}

	public static void fileDisplay(String path) {
		FileOperations.createFolderIfFolderNotAvailable("proj");

		System.out.println("Files are Displying in Asscending order with directory\n");
```

```
		List<String> fileNamesList = FileOperations.filesInDirectory(path, 0, new
ArrayList<String>());

		System.out.println("Displaying files in ascending order\n");
		Collections.sort(fileNamesList);
		for(String file : fileNamesList) {
			System.out.println(file);
		}
	}

	public static List<String> filesInDirectory(String path, int indent, List<String> fileNamesList) {
		File directory = new File(path);
		File[] filesArray = directory.listFiles();
		List<File> filesList = Arrays.asList(filesArray);

		Collections.sort(filesList);

		if (filesArray != null && filesArray.length > 0) {
			for (File file : filesList) {

				System.out.print(" ".repeat(indent * 2));

				if (file.isDirectory()) {
					System.out.println("    " + file.getName());

					// Recursively indent and display the files
					fileNamesList.add(file.getName());
					filesInDirectory(file.getAbsolutePath(), indent + 1,
fileNamesList);
				} else {
					System.out.println("    " + file.getName());
					fileNamesList.add(file.getName());
```

```
                    }
                }
            } else {
                System.out.print("  ".repeat(indent * 2));
                System.out.println("    Empty Directory");
            }
            System.out.println();
            return fileNamesList;
        }

        public static void createNewFile(String fileToAdd, Scanner sc) {
            FileOperations.createFolderIfFolderNotAvailable("proj");
            Path pathToFile = Paths.get("./proj/" + fileToAdd);
            try {
                Files.createDirectories(pathToFile.getParent());
                Files.createFile(pathToFile);
                System.out.println(fileToAdd + " File created successfully");

                System.out.println("Do you want to add some content to the file?
(yes/no)");
                String choice = sc.next().toLowerCase();

                sc.nextLine();
                if (choice.equals("yes")) {
                    System.out.println("\n Input content added succesfully and press
enter ");
                    String content = sc.nextLine();
                    Files.write(pathToFile, content.getBytes());
                    System.out.println(" Content added to file " + fileToAdd);
                    System.out.println("Content can be read using by using any editor");
                }
```

```
		} catch (IOException e) {
			System.out.println("Failed to create new file " + fileToAdd);
			System.out.println(e.getClass().getName());
		}
	}

	public static List<String> fileLocations(String fileName, String filePath) {
		List<String> fileListNames = new ArrayList<>();
		FileOperations.fileSearch(filePath, fileName, fileListNames);

		if (fileListNames.isEmpty()) {
			System.out.println(" Couldn't find any file with given file name " + fileName
+ "\n");
		} else {
			System.out.println("Found file at below location(s):");

			List<String> filesList = IntStream.range(0, fileListNames.size())
					.mapToObj(index -> (index + 1) + ": " +
fileListNames.get(index)).collect(Collectors.toList());

			for(String file : filesList) {
				System.out.println(file);
			}
			for(String file : filesList) {
				System.out.println(file);
			}

			for(String file : filesList) {
				System.out.println(file);
			}

		}
```

```
        return fileListNames;
}

public static void fileSearch(String filePath, String fName, List<String> fileNamesList) {
        File dir = new File(filePath);
        File[] files = dir.listFiles();
        List<File> filesList = Arrays.asList(files);

        if (files != null && files.length > 0) {
                for (File file : filesList) {

                        if (file.getName().startsWith(fName)) {
                                fileNamesList.add(file.getAbsolutePath());
                        }

                        if (file.isDirectory()) {
                                fileSearch(file.getAbsolutePath(), fName, fileNamesList);
                        }
                }
        }
}

public static void fileDelete(String filePath) {

        File currentFile = new File(filePath);
        File[] files = currentFile.listFiles();

        if (files != null && files.length > 0) {
                for (File file : files) {
```

```
                    String fileName = file.getName() + " at " + file.getParent();
                    if (file.isDirectory()) {
                        fileDelete(file.getAbsolutePath());
                    }

                    if (file.delete()) {
                        System.out.println(fileName + " deleted successfully");
                    } else {
                        System.out.println("Failed to delete the file " + fileName);
                    }
                }
            }

            String currentFileName = currentFile.getName() + " at " + currentFile.getParent();
            if (currentFile.delete()) {
                System.out.println(currentFileName + " deleted successfully");
            } else {
                System.out.println("Failed to delete the file " + currentFileName);
            }
        }
}
```

```
public class Menu {

    public static void printWelcomeScreen(String application, String developer) {
        String company = String.format("Welcome to"+application + "This application was
developed by "+ developer);
        String appFeatures = "You can use this application to :-\n"
                                + " Access all files in the \"proj\" folder\n"
```

```
                                                    + " Search, add, or delete files in \"proj\"
folder.\n"
                                                    + " Please enter correct filenames for
searching or deleting files";
                //System.out.println(company);


                System.out.println(appFeatures);
        }


        public static void menu() {
                String menu = " Select any option number from below and press Enter\n"
                                + "1) Access all files inside \"proj\" folder\n"
                                + "2) Display menu for File operations\n"
                                + "3) Exit program\n";
                System.out.println(menu);


        }


        public static void menuOptions() {
                String fileMenu = "\n Select any option number from below and press Enter \n"
                                + "1) Add new file to \"proj\" folder\n"
                                + "2) Delete a file from \"proj\" folder\n"
                                + "3) Search for a file from \"proj\" folder\n"
                                + "4) Show Previous Menu options \n"
                                + "5) Exit program\n";


                System.out.println(fileMenu);
        }


}
```

```
import java.util.List;
import java.util.Scanner;



public class Options {
	public static void welcomeInput() {
		boolean running = true;
		Scanner sc = new Scanner(System.in);
		do {
			try {
				Menu.menu();
				int option = sc.nextInt();

				switch (option) {
				case 1:
					FileOperations.fileDisplay("proj");
					break;
				case 2:
					Options.handleFileMenuOptions();
					break;
				case 3:
					System.out.println("Program Exited Successfully.");
					running = false;
					sc.close();
					System.exit(0);
					break;
				default:
					System.out.println("Please select a valid option from above
displayed options.");
```

```
                }
            } catch (Exception e) {
                System.out.println(e.getClass().getName());
                welcomeInput();
            }
        } while (running == true);
    }

    public static void handleFileMenuOptions() {
        boolean running = true;
        Scanner sc = new Scanner(System.in);
        do {
            try {
                Menu.menuOptions();
                FileOperations.createFolderIfFolderNotAvailable("proj");

                int option = sc.nextInt();
                switch (option) {
                case 1:
                    // File Add
                    System.out.println("Enter the name of the file to be added
to the \"proj\" folder");
                    String fileToAdd = sc.next();

                    FileOperations.createNewFile(fileToAdd, sc);

                    break;
                case 2:
                    // File/Folder delete
                    System.out.println("Enter the name of the file to be deleted
from \"proj\" folder");
                    String fileToDelete = sc.next();
```

```
                                        FileOperations.createFolderIfFolderNotAvailable("proj");
                                        List<String> filesToDelete =
FileOperations.fileLocations(fileToDelete, "proj");

                                        String deletionPrompt = "Select index of which file to
delete?"
                                                + "\n(Enter 0 if you want to delete all
elements)";
                                        System.out.println(deletionPrompt);

                                        int index = sc.nextInt();

                                        if (index != 0) {
                                                FileOperations.fileDelete(filesToDelete.get(index -
1));
                                        }
                                        else {
                                                for (String path : filesToDelete) {
                                                        FileOperations.fileDelete(path);
                                                }
                                        }


                                        break;
                                case 3:
                                        // File/Folder Search
                                        System.out.println("Enter the name of the file to search
from \"proj\" folder");
                                        String fileName = sc.next();

                                        FileOperations.createFolderIfFolderNotAvailable("proj");
                                        FileOperations.fileLocations(fileName, "proj");
```

```
                                        break;
                                case 4:
                                        // Go to Previous menu
                                        return;
                                case 5:
                                        // Exit
                                        System.out.println("Program exited successfully.");
                                        running = false;
                                        sc.close();
                                        System.exit(0);
                                default:
                                        System.out.println("Please select a valid option from the
above displayed options..");
                                }
                        } catch (Exception e) {
                                System.out.println(e.getClass().getName());
                                handleFileMenuOptions();
                        }
                } while (running == true);
        }
}

public class VirtualKeyInstaller {

        public static void main(String[] args) {

                FileOperations.createFolderIfFolderNotAvailable("proj");

                Menu.printWelcomeScreen(" Virtual key ", "**********");
```

```
                Options.welcomeInput();
        }


}
```