**שיעורי בית**

1. איך יוצרים תיקייה בג'אווה?

כדי ליצור תיקייה בג'אווה, משתמשים במחלקה File ו משתמשים בשיטה mkdir(). הפעולה יוצרת תיקייה אם היא לא קיימת.

1. איך יוצרים קובץ?

קובץ יוצרים באמצעות המחלקה File והשיטה createNewFile(). אם הקובץ כבר קיים, הוא לא ייווצר מחדש. חשוב לטפל בשגיאות. (IOException)

1. איך בודקים אם קובץ או תיקייה קיימים?

בודקים עם השיטה exists() שנמצאת במחלקה File. היא מחזירה true אם הקובץ או התיקייה קיימים במערכת.

1. איך מציגים את התוכן של תיקייה?

משתמשים בשיטה listFiles() של המחלקה File, שמחזירה את כל הקבצים והתיקיות שבתוך תיקייה מסוימת.

1. מה ההבדל בין File ל־Path?

File היא מחלקה ישנה מ־java.io, ולעומתה Path שייכת למערכת החדשה יותר

.(java.nio.file). Path נחשבת מודרנית, גמישה ומומלצת לשימוש בפרויקטים חדשים

Readme file:

# 🐚 MiniShell

## 📖 Project Overview

\*\*MiniShell\*\* is a lightweight Java-based command-line tool that simulates essential shell commands for navigating and manipulating the file system. It provides a minimal yet functional interface for performing operations like creating files, making directories, changing directories, and listing contents, all within the constraints of a Java application.

---

## ✨ Features

The following commands are supported:

- `pwd`: \*\*Print Working Directory\*\* – Displays the current absolute path.

- `ls`: \*\*List\*\* – Lists all files and directories in the current directory.

- `cd [directory\_name]`: \*\*Change Directory\*\* – Navigates to the specified directory. Use `..` to go up one level.

- `mkdir [directory\_name]`: \*\*Make Directory\*\* – Creates a new subdirectory in the current path.

- `touch [file\_name]`: \*\*Create File\*\* – Creates a new empty file.

- `help`: \*\*Help\*\* – Lists all available commands with brief descriptions.

- `exit`: \*\*Exit\*\* – Terminates the shell session.

---

## 🧱 Project Structure

The codebase consists of two main classes:

### `MiniShell.java`

- Acts as the main entry point for the application.

- Uses a `Scanner` to receive user input.

- Contains a loop that:

- Displays a prompt with the current working directory.

- Accepts and parses user commands.

- Delegates command execution to the `ShellCommandHandler`.

### `ShellCommandHandler.java`

- Provides the implementation for each shell-like command.

- Maintains and updates the current working directory.

- Encapsulates file system logic like listing, creating, and navigating directories and files.

---

## 🔧 Implementation Details

### `ShellCommandHandler` Class

| Method | Description |

|--------|-------------|

| `printWorkingDirectory()` | Prints the absolute path of the current directory. |

| `listDirectory()` | Lists the names of files and folders in the current directory. |

| `changeDirectory(String name)` | Changes directory to a subfolder or parent (`..`). |

| `makeDirectory(String name)` | Creates a new directory inside the current one. |

| `createFile(String name)` | Creates a new file if it doesn't already exist. |

| `printHelp()` | Displays usage information for all supported commands. |

| `getCurrentDirectory()` | Returns the current working directory as a `File` object. |

Helper Method:

- `tryCreateFile(File file)`: Handles checked exceptions during file creation.

---

## ▶️ How to Run

### 📦 Compile

```bash

javac ShellCommandHandler.java

javac MiniShell.java

🚀 Execute

bash

Copy

Edit

java MiniShell

💻 Sample Usage

bash

Copy

Edit

welcome to MiniShell, type 'help' for commands

/home/user > pwd

/home/user

/home/user > mkdir projects

/home/user > cd projects

/home/user/projects > touch notes.txt

/home/user/projects > ls

notes.txt

/home/user/projects > cd ..

/home/user > ls

projects

/home/user > help

supported commands:

pwd , print current directory

ls , list files and directories

cd (directory) , change directory

mkdir (directory) , create a new directory

touch (filename) , Create a new file

help , show this help message

exit , exit the shell

/home/user > exit

Exiting MiniShell.

⚠️ Error Handling

The shell includes basic error handling for common situations:

Attempting to cd into a non-existent directory.

Trying to create a directory or file that already exists.

Entering unrecognized commands.

Handling IOException during file creation gracefully.

🚫 Limitations

Does not support absolute paths or complex path navigation (e.g., ../../folder).

No support for deleting, copying, or editing files.

Lacks advanced shell features like piping, redirection, or wildcard expansion.

File content manipulation is not supported (only file creation).

🔮 Future Improvements

Here are some suggested enhancements:

Add file deletion and renaming functionality.

Enable reading and writing to file content.

Improve path handling (support . and absolute paths).

Add command history and navigation.

Integrate tab completion for files and folders.

Format ls output to distinguish files and directories more clearly.

📝 License

This project is free and open-source. You may use, modify, and distribute it for educational or personal purposes.