In-Memory File System Documentation

Introduction
This C++ program implements a basic in-memory file system that allows users to perform file and directory operations through a command-line interface. The file system supports commands such as creating directories, navigating the directory structure, listing contents, creating and modifying files, moving, copying, and deleting files and directories.

`FileSystem` Class
Private Members
- `string currentDir`: Represents the current working directory in the file system.
- `map<string, string> files`: A map to store file contents, where the key is the file path, and the value is the file content.

Public Member Functions
1. **`FileSystem()`**
   - Constructor: Initializes the file system with the root directory (`/`).

2. **`mkdir(const string& dirName)`**
   - Creates a new directory.

3. **`cd(const string& path)`**
   - Changes the current directory.
   - Supports relative paths like `..`, absolute paths like `/`, and navigating to a specified path.

4. **`ls()`**
   - Lists the contents of the current directory.
   - Differentiates between directories and files in the output.

5. **`touch(const string& fileName)`**
   - Creates a new empty file.

6. **`echo(const string& content, const string& fileName)`**
   - Writes text to a file.

7. **`cat(const string& fileName)`**
   - Displays the contents of a file.

8. **`mv(const string& source, const string& destination)`**
   - Moves a file or directory to another location.

9. **`cp(const string& source, const string& destination)`**
   - Copies a file or directory to another location.

10. **`rm(const string& fileName)`**
    - Removes a file or directory.

`main` Function
- The `main` function serves as the entry point of the program.
- Creates an instance of the `FileSystem` class.
- Accepts user input in an infinite loop, processing and executing commands until the user enters "exit."

Usage
- **Commands:**
  - `mkdir`: Create a new directory.
  - `cd`: Change the current directory.
  - `ls`: List directory contents.
  - `touch`: Create a new empty file.
  - `echo`: Write text to a file.
  - `cat`: Display the contents of a file.
  - `mv`: Move a file or directory.
  - `cp`: Copy a file or directory.
  - `rm`: Remove a file or directory.
  - `exit`: Terminate the program.

- **Syntax:**
  - Commands are entered with appropriate parameters (e.g., `mkdir dirname`, `cd path`, `ls`, etc.).
  - Ensure proper quoting for filenames and text containing spaces or special characters.

Error Handling
- The program provides feedback for invalid commands or files not found during execution.
- Users are informed when attempting to move or copy nonexistent files or directories.

Notes
- The program supports an interactive mode where users can input commands directly.
- It is essential to handle user input carefully, especially when dealing with filenames and spaces.
- The program supports basic file operations and provides a foundation for more advanced functionalities if needed.