Implementation of code: In-memory file system

1. The **InMemoryFileSystem** class is designed to represent a simple in-memory file system. It has two main attributes:
2. **current_directory**: Keeps track of the current working directory.
3. **file_system:** A dictionary that represents the file system structure. Directories are keys, and their values are dictionaries representing files and subdirectories.
4. The **mkdir** method creates a new directory. It forms the full path of the new directory using os.path.join and then adds an entry to the file_system dictionary with an empty dictionary as its value.
5. The **touch** method creates a new empty file. It normalizes the file path and adds an entry to the file_system dictionary with an empty content field.
6. The **cd** method changes the current directory based on the provided path. It handles absolute paths, '..' for moving to the parent directory, and other relative paths.
7. The **ls** method lists the contents of a directory. It normalizes the path and checks if the directory exists in the file system. If it does, it prints the contents.
8. **File Operations (cat, echo, mv, cp, rm)**
   These methods handle various file operations such as
   - displaying file content.
   - writing to a file.
   - moving.
   - copying.
   - removing files.

   They use normalized paths and manipulate the file_system dictionary accordingly.
9. **Data Types Used:**

   String: Strings are used to represent directory and file paths throughout the code.

   Example: self.current_directory, directory_name, file_name, file_path.

   Dictionary: Dictionaries are used to represent the file system structure.

   The keys are directory paths, and the values are dictionaries representing files and subdirectories within each directory.

   Example: self.file_system.

   List: Lists are used to store the contents of directories when listing them.

Example: contents in the ls method.

JSON (serialized dictionary): The json module is used to serialize and deserialize the file system state when saving and loading.

Example: json.dump(self.file_system, file) in save_state, self.file_system = json.load(file) in load_state.

Packages/Modules Used:

OS:

The os module is used for interacting with the operating system, constructing file paths, and performing path normalization.

Example: os.path.join, os.path.normpath, os.path.dirname.

json:

The json module is used for JSON serialization and deserialization.

Example: json.dump, json.load.

re (regular expression):

The re module is used for regular expression matching in the grep method (bonus functionality).

Example: re.search.

10. Code Overview

Initialization:

The __init__ method initializes the in-memory file system with the root directory (/) and an empty file system dictionary.

Directory and File Operations: mkdir, touch, cd, ls, cat, echo, mv, cp, rm methods handle directory and file operations.

Regular Expression (Bonus): The grep method uses regular expressions to search for a specified pattern in a file.

Path Handling: The os module is used for constructing and normalizing file paths.

File System State Persistence: The save_state and load_state methods use the json module to serialize and deserialize the file system state for persistence.