Purpose of the Operating System:
The purpose of this operating system code is to implement a simplified, in-memory file system that demonstrates core OS functionalities such as file management, concurrency, and user interaction. The program provides a user-space interface that mimics basic operating system behavior by managing files and handling concurrent access through thread synchronization. It allows users to execute common file operations via a command-line interface, enabling functional insights into how real OS-level file handling and process synchronization work.

Explanation of implemented Functions:
The operating system includes the following key functionalities:

- **create_file**: Creates a new file in the file system with a specified name and content.
- **read_file**: Displays the content of a file if it exists.
- **append_file**: Appends additional data to an existing file.
- **rename_file**: Renames an existing file by creating a new one with the same data and deleting the original.
- **delete_file**: Removes a file from the system.
- **list**: Lists all files currently stored in the system.
- **free_file_system**: Deallocates memory used by the file system.
- **Concurrency Support**: Uses pthread and pthread_mutex_t to handle simultaneous file operations safely.
- **User Interface**: Accepts command-line input for performing file operations like cr, r, a, d, rn, ls, and exit.

Total Number of lines of code: 225
This includes files file_system.c, file_system.h, kernel.c