# MyFind - Protocol

This protocol describes the key parallelization concepts and the synchronization mechanisms used in the `myfind` project, which is designed to search for multiple files in a directory and its subdirectories in parallel.

### Parallelization Concepts

## Forking Processes:

- For each filename provided as an argument, the main program forks a new child process using `fork()`. This allows each file search to be performed in parallel.

- Each child process executes the search independently, ensuring that multiple files are searched simultaneously, reducing the overall search time.

#### Argument Handling:

- The program uses `getopt()` for parsing command-line arguments, handling options `-R` for recursive search and `-i` for case-insensitive search.

- The search path and filenames are extracted from the parsed arguments, and each filename triggers a new child process.

### Synchronization of Output

#### 1. Output Synchronization:

- To ensure the output from multiple child processes is synchronized and readable, a mutex (mutual exclusion) mechanism is employed.

- The `OutputSynchronizer` class uses a `std::mutex` to lock the output stream when a child process prints its results. This prevents interleaved and jumbled output from concurrent processes.

**2. Process Termination Handling:**

- The parent process uses `wait()` to handle the termination of child processes. This prevents the creation of zombie processes and ensures all child processes are properly terminated before the program exits.

**Code Overview:**

ArgumentParser:

Handles the parsing of command-line arguments and stores the parsed search path, filenames, and options (`-R` and `-i`).

FileFinder:

Implements the search functionality, checking each file in the directory (and subdirectories if recursive) against the specified filename, considering case sensitivity if needed.

ProcessManager:

Manages the creation of child processes, initiates the search for each filename, and ensures synchronization of the output.

OutputSynchronizer:

Provides a thread-safe mechanism for printing the output of each child process using a mutex.

# Program Execution:

The makefile is auto generated using Cmake to ensure compatibility across Os's.

We tested it on both windows and linux.

cd d_sys-myfind/build && cmake .. && make

followed by executing the ./myfind program.