Object-Oriented Programming and Design - Monsoon 2023

Project on Multi-threaded Programming and Profiling

This project carries a total of 200 marks.

1. **Basic implementation of commands:** The Linux shell provides a number of internal commands for final manipulation, including `cd`, `mv` and `rm`, `ls` and `cp`. These commands can be classified into two types – (1) which modify/write to the file system (`mv`, `rm` and `cp`), and (2) which do not make any modifications to the file system (`cd` and `ls`). Design your own shell that accepts all these five commands. Note that these commands also accept additional options, to have suitable modifications to their behavior. Implement at least four options of your own choice, including the recursive (wherever applicable) and help options. Your implementation must utilize classes and objects, and should support wildcard characters.
2. **Profiling:** Use a bash script to create two directories – (1) containing 100 files of 1GB each, (2) containing 10000 files of 10MB each, (3) containing only 10 MB files, but only 100 of them directly, and having another subdirectory with the same thing as above done recursively until a total of 10000 files (excluding directories) are created. Use the time command to find out the time taken by each of the above commands for each case.
3. **Multi-threaded Implementation:** For the external commands, implement a multi-threaded implementation for anything involving recursion. The number of threads should be created intelligently based on the number of cores available and based on the amount of workload. This should use the multi-threaded library.
4. **Profiling of Multi-threaded Version:** Re-profile this multi-threaded version and compare the performance with the previous one.

What and How To Submit

- The C++ program sources. Classes and inheritance must be used wherever appropriate.
- The bash script files to create the directory and file structures.
- `Makefile` to compile the sources and generate the running binary for the shell. The Makefile should generate two versions of the binary – one for debugging and another for optimized execution.
- Utilization of multiple files to run the entire program is compulsory.

- A readme text file, explaining the commands needed to build the file, and the format of the input files. If code is copied from anywhere else (not that copying from any other student is plagiarism, but using textbook or open-source code is allowed), that should be mentioned here.
- At least 4 significant commits on a **private** github repository, with proper descriptions of the commits. You may have as many commits as you wish.
- Make the assigned TA the admin of the github repository, **and** submit the same code in zipped form on Google Classroom by the due date.

Grading Rubric

1. 80 marks (16 marks each) for basic implementation of commands – 6 marks for the basic functionality, 5 marks for all options except recursive, 5 marks for the recursive version. For a command like cd where no recursion is possible, 11 marks for the basic functionality.
2. 50 marks for profiling – 10 marks for the bash script for condition (1), 10 marks for condition (2) and 15 marks for condition (3). Additional 5 marks each for proper recording of output of the time commands.
3. 45 marks for implementation of the multi-threaded version – 30 marks for proper utilization and creation of threads and 15 marks for intelligent creation of threads.
4. 15 marks for re-profiling and its proper output.
5. 10 marks for utilizing git and makefile properly

Late Submission Policy

- -0.25 per hour for the first 96 hours.
- Submissions beyond 4 days of delay would only be accepted with official leaves of absence.