CS360 – Operating Systems
Assignment 3 – Multi-threaded web server

Due date: Friday March 11, 2022 at 11:59 PM
Solve this project on your own relying on materials and examples demonstrated in class.

This project was borrowed from Prof. Remzi Arpaci-Dusseau Operating Systems class at the *University of Wisconsin – Madison*.

Operating Systems: Three Easy Pieces
Remzi H. Arpaci-Dusseau and Andrea C. Arpaci-Dusseau
Arpaci-Dusseau Books
August, 2018 (Version 1.00)

Objective

- Practice writing concurrent multi-threaded programs.
- Practice working on an existing code base. This is an activity that most developers dislike at the beginning – I did too. In general developers don't like to put the effort to understand someone else's code. In my experience, it is something you get used to as you keep doing it. And you will eventually get to a point when you no longer mind it.

Instructions

- Go to this page. This is the project you will do: change an existing single-threaded webserver to make it multi-threaded.
- Read all the sections and familiarize yourself with how web-servers work in general.
- Login to your VM machine.
- Open a terminal window and clone Remzi's ostep-projects by executing the git clone command:

  ```
  git clone https://github.com/remzi-arpacidusseau/ostep-projects
  ```

  The above creates a folder ("ostep-projects") in the directory where you executed the clone command.
- Create a folder ("Assignment3") somewhere on your VM.
- Copy subfolder "src" (found inside "ostep-projects/ concurrency-webserver") and paste it into the "Assignment3" folder. You should now have "src" inside "Assignment3":

Assignment3
src

- You can now delete the “ostep-projects” folder because you don’t need all the other projects.

- cd to Assignemnt3/src.

- To build the webserver just run: `make`

  This creates executable `wserver` (this is the web-server) and other tools like `wclient`.

- In the “Command-line Parameters” section are instructions on how to run the web server. Read the meaning of each command-line argument.

```
prompt> ./wserver [-d basedir] [-p port] [-t threads] [-b buffers] [-s schedalg]
```

- The `basedir` is the root directory of where your html files are. For example, if you put a few html test files in a folder called `htmlfolder` in your home directory, then `basedir` will be: ~/htmlfolder

  The parameter p is the port number the server is listening on. If you don’t specify a value for -p, the server will be listening on port 10000.

- Before writing any code, test how the server works. You can start it like this (assuming you put a few HTML test files in folder `htmlfolder` located in your home directory):

  prompt> `./wserver -d ~/htmlfolder`

  Then you can test it by typing the following in your VM browser (assuming you have file index.html in htmlfolder):

  `http://localhost:10000/index.html`

  The content of file index.html should load in your browser.

- When you implement multithreading, the number of threads you create is configurable and is passed to the server via the `-t` flag. Same for the size of the buffer which is passed via the `-b` flag. In other words, you should not hardcode the number of threads to create and the size of the buffer. Instead, you create those based on the values passed to main from the command line.

- For the `-schedalg` argument, you can implement FIFO only. No need to implement SFF as instructed.

- Enclosed are 2 HTML test files.

- At the beginning try to manually test your webserver (using a browser or tool `wclient`). Then use the automated tests provided to you in subfolder "HowToRunTests". Read file "MultiThreadedServerTest.pdf" to learn how to run the tests.

  Your grade will be based on how your implementation does against the automated tests.

- I should be able to build your code by simply typing: `make`

  Note that if you add any new source code files, then you need to update `make` so that these files are included in the build. But you don't have to (your code can be in existing source code files).

Grading

Your implementation will be graded using automated tests (see subfolder HowToRunTests – read file "MultiThreadedServerTest.pdf" to learn how to run the tests).

What to Submit

- Create folder **john_smith_hw2** (replace john smith with your name)
- Put the webserver source code (that includes the modifications you made) in folder **src**.
- Put folder **src** in **john_smith_hw3**.

```
john_smith_hw3
    src
        wserver.c
        request.c
        …
        …
        make
        …
        …
```

- Zip john_smith_hw3 to generate **john_smith_hw3.zip**.
- Upload **john_smith_hw3.zip** to Canvas before the due date (Friday March 11, 11:59 PM).