COM3610
Programming Assignment 2
Threading and Synchronization

Overview: New Functionality

Multi-threaded Client

Your finished web client must be invoked exactly as follows:

```
client [host] [portnum] [threads] [schedalg] [filename1] [filename2]
```

The command line arguments to your web server are to be interpreted as follows.

- `host:` the name of the host that the web server is running on; the basic web client already handles this argument.
- `portnum:` the port number that the web server is listening on and that the client should send to; the basic web client already handles this argument.
- `threads:` the number of threads that should be created within the web client. Must be a positive integer.
- `schedalg:` the scheduling algorithm to be performed. Must be one of CONCUR or FIFO.
- `filename1:` the name of the file that the client is requesting from the server.
- `filename2:` the name of a second file that the client is requesting from the server. This argument is optional. If it does not exist, then the client should repeatedly ask for only the first file. If it does exist, then each thread of the client should alternate which file it is requesting.

The `schedalg` command line argument will either be:

- `FIFO:` The scheme implemented in the last assignment
- `CONCUR:` The client creates N threads and uses those threads to concurrently (i.e., simultaneously) perform N requests for the same file; this behavior repeats forever (until the client is killed). You should ensure that the N threads can overlap sending and waiting for their requests with each other. After all of the N threads receive their responses, the threads should repeat the requests. You may find the routine `pthread_barrier_wait` useful for implementing this. **In no case should busy-waiting be used**.

  If `filename2` is specified, all threads request `filename1` in the first "round". Then, all threads request `filename2` in the second "round", and so forth.

Grading

Hand in a zip file of your client source code, make file, and a README file to Canvas. Do not include any .o files, html files, or graphics files. Make sure that your name is listed in the README file.

In your `README` file you should have the following five sections:

- Design overview: A few simple paragraphs describing the overall structure of your code and any important structures.
- Complete specification: Describe how you handled any ambiguities in the specification.
- Known bugs or problems: A list of any features that you did not implement or that you know are not working correctly
- **Testing:** This requirement an aspect that I am very interested in. **I will assign points for answering these questions.** Describe how you tested the functionality of your web client.

NOTE: Your submission will be tested and graded using the COM3610 Virtual Machine, so be sure your code compiles and runs correctly there before handing in your submission.