Assignment 1

Write a client server socket program in C. Your program should have the following features. [2+(1+3)+1+ (3+1)+(2+2)+(1+2)+1]

1. Server sets up a TCP socket and listens for client connections.
2. The client creates a socket and initiates the TCP connection. Server should have the capability to handle multiple concurrent clients (multithreaded, concurrent server). The server accepts the client connection, hence, a new socket is created with 4 tuples (serverip, server listening port, client ip, client port). Server creates a new thread that continues to process the client connection (Hint: Use pthread library for multithreading). The original server socket continues to listen on the same listening port for newer incoming client connections.
3. After the client connection is established, the client sends a request to the server to get the information about server's top "N" CPU consuming processes.
4. Server finds out the top "N" CPU consuming processes (user+kernel CPU time), gathers information such as process name, pid (process id), and CPU usage of the process in user & kernel mode (you can report this time in clock ticks). Server writes this information to a local file.
   Hint: The server should make use of the open() system call to read the "proc" filesystem to get this information. You need to read ***/proc/[pid]/stat*** for all processes to parse the process name, pid, and CPU time (user+kernel). To understand the format of ***/proc/[pid]/stat*** file, refer https://man7.org/linux/man-pages/man5/proc.5.html
5. Server sends the file created in step 4 to the client. Client stores it in its local drive.
6. Client identifies the process that consumes the highest amount of CPU (similar to step 4, but reports only the top CPU consuming process) and sends the process information (process name, pid and CPU usage) to the server.
7. The client closes the connection.
8. **Bonus question (For the ones who want to learn more. Does NOT carry marks.):** Observe the client's response time as the number of concurrent requests increase, and plot a graph (X-axis: #concurrent requests, Y-axis: response time)

Sample format of *[/proc/[pid]/stat](https://man7.org/linux/man-pages/man5/proc.5.html)*
(Details at** https://man7.org/linux/man-pages/man5/proc.5.html**)
User space and kernel space CPU time (measured time in clock ticks) is marked in **RED**.

```
$ cat /proc/76893/stat
76893 (top) S 76734 76893 76734 34825 76893 4194304 400 0 0 0 573 806 0 0 20 0 1 0
1453568921 43216896 1066 18446744073709551615 4194304 4292532 140726554046848 0 0 0 0
0 2147155711 1 0 0 17 15 0 0 0 0 0 6393312 6398352 14311424 140726554052535
140726554052539 140726554052539 140726554054635
```

How can you test concurrent/parallel client requests at the server?

- Start the multithreaded server process
- Manually start multiple client processes (one after another). Before the client closes the connection, put the client process to sleep, for say, 30 seconds. This will ensure that multiple client connections exist simultaneously for some time at the server. Alternatively, you may write a script to initiate concurrent client connections.