CAB403 REPORT

A companion report to the game of Hangman

Duval Longa	n9452257
Sam Bradley	n9659307

1.) **Statement of completeness:**
Our submission addresses tasks 1 & 2. The server will refuse to accept any more than ten clients. There are no deficiencies that we are aware of.

2.) **Team members:**
Duval Longa, n9452257
Sam Bradley, n9659307

3.) **Team member contributions:**
Both team members contributed equally.

4.) **Data structure for Leader Board:**
The leaderboard is a linked list composed of structures containing the user details for each of the ten accounts. The leaderboard linked list is similar to the linked list used to hold authentication details for client accounts, although it is dynamic. Upon a client request, and if the critical section requirements are met, the list is re-ordered based on total wins, win ratio, and user names.

5.) **Critical-section solution:**
The readers/writers problem is solved through the use of semaphores. The leaderboard database is only written to by one thread at a time. Other threads attempting to simultaneously write to the leaderboard are signalled to wait (sem_wait(&sbAccess)) until the writing thread is complete. Once the current write operation is completed, they will be signalled using (sem_post(&dbAccess)).

There are no issues with multiple threads reading the database simultaneously. But, no writer threads can access the database while other threads are reading, and thus the same signalling and posting procedure is used to force the writer threads to wait until all current reader threads have completed reading.

6.) **Thread pool solution:**
Not attempted.

7.) **Instructions for compilation and execution:**
Compilation:
Server: `c99 Hangman_Server.c -g -pthread -o Hangman_Server`

Client: `c99 Hangman_Client.c -g -o Hangman_Client`

Execution:
Server: `./Hangman_Server [port number]`

Client: `./Hangman_Client [localhost] [port number]`