Program 1

# Problem Statement

The assigned task was to design, implement, and test a client/server program in C/C++ which uses message queues to transfer and request results. It must use the fork to split the client process from the server process. The client was to take user input, make requests and print results. The server was to process the requests and return the results to the client. The user should be able to request the domain name, host name, and uname of the system. The program must work properly on Ubuntu 20.04 LTS.

# Approach

The decision was made to develop the program in C due to programmer familiarity and its relatively low resource consumption compared to C++. The program was developed in Visual Studio Code on Ubuntu 20.04 LTS. The program was made so that the main process splits using fork, with the parent becoming the server and the child becoming the client. The server and client functions are implemented as separate functions. The decision was made to use two message queues: one for the client to send information to the server and another for the server to send information to the client. This was chosen to avoid the potential error risks that come with a single read/write message queue. To test the program, we can compile and run it on Ubuntu’s terminal.

# Solution

The assigned task was to design, implement, and test a client/server program in C/C++ which uses message queues to transfer and request results. The process forks in the main function, splitting into a parent and child. The parent becomes the server and the child becomes the client. The client opens both the message queues from its end. At the same time, the server opens both the message queues from its end. From there, the following happens until the user requests to quit:

1. The client asks the user for input.
2. The client gets the input from the user. If the user requests to quit, the loop breaks and both processes are closed.
3. Otherwise, the client sends the user’s request to the server
4. The server receives the request.
5. The server finds the requested information, and sends the results back to the client. If the request is invalid, the results are set to “Invalid request from client”.
6. The client receives the results.
7. The client outputs the results

We can build, run, and test the program in Linux’s terminal.

1. First, install gcc. If gcc is already installed, this step can be skipped.
   1. Use ‘sudo apt install build-essential’ or ‘apt install build-essential’ or ‘apt install gcc’ to install via terminal.
2. Navigate to the directory that the file is stored in.
3. Compile the program using the command ‘gcc bsa190001\_program1.c -lrt -o bsa190001\_program1’
4. Run the program in the terminal
   1. Preferred method is ‘./bsa190001\_program1’
5. Follow prompts on screen to test

Figure 1 demonstrates how the program was compiled and tested. The output of the test shows that all required functionality works properly.



*Figure 1. This image shows an example of the program being compiled, ran, and tested. All functionality works as outlined by the program. The program is built using gcc, and each option (Domain Name, Hostname, UNAME, and Exit) are tested by inputting the respective numbers.*