IPC CLIENT-SERVER FILE TRANSFER MESSAGING SYSTEM

Linux Client-Server

Linux message queue based internal process client-server system for sending/receiving files

Carson Roscoe

Contents

Purpose

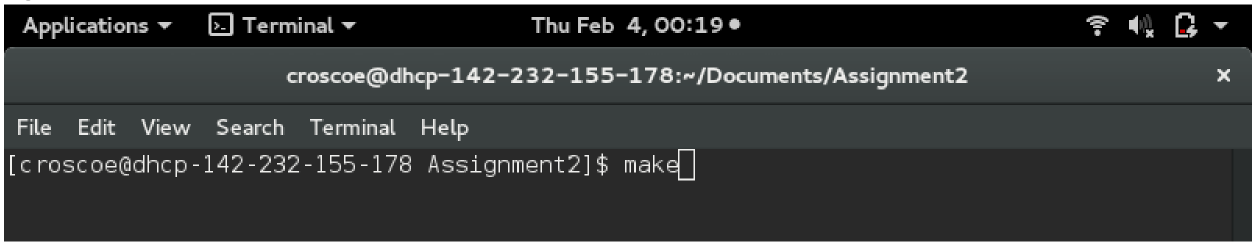
The purpose of these applications is to utilize message queues to send & receive file data. The clients request a file name & a priority between low/medium/high, and in turn the server fetches that data from the file and returns it to the client via messaging queues.

The server needs to be able to handle multiple clients, as well as have both the server and client handle cleanup when something goes wrong, like the other dies unexpectedly.

Usage

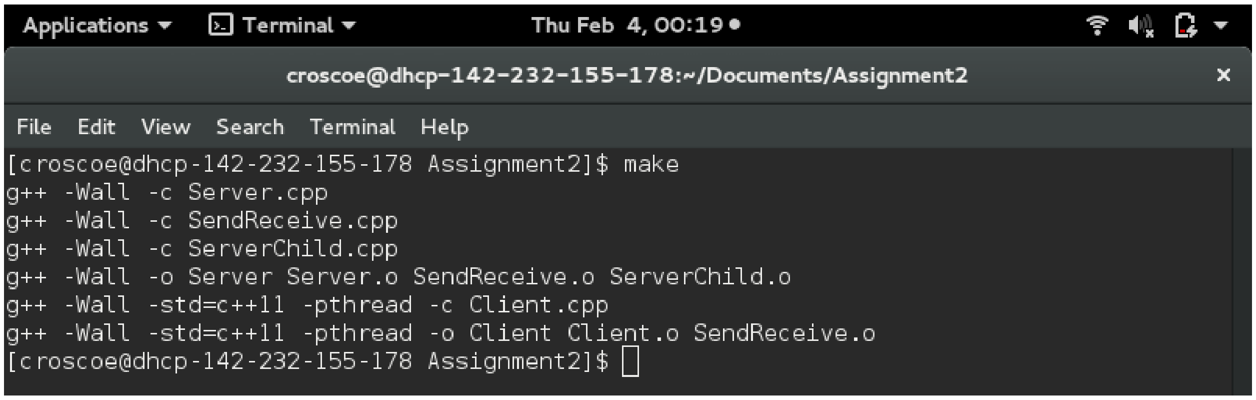
The first step is to compile our program. Redirect your terminal to the appropriate directory where the files are stored and simply type make as follows.

Figure A: Before Make File



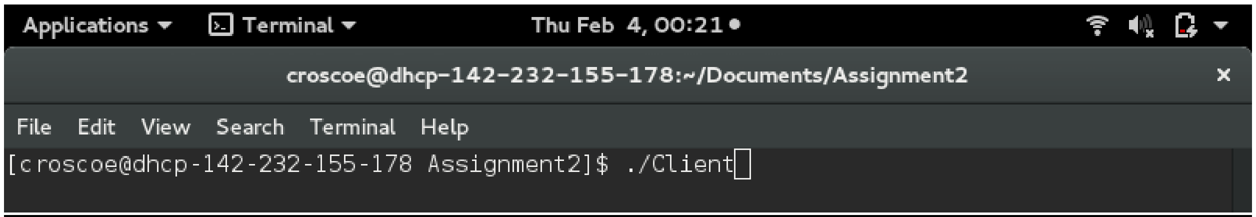
```
[croscoe@dhcp-142-232-155-178 Assignment2]$ make
```

Figure B: After Make File

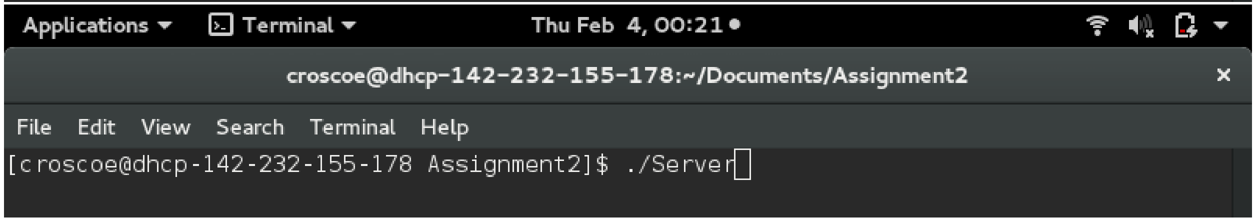


```
[croscoe@dhcp-142-232-155-178 Assignment2]$ make
g++ -Wall -c Server.cpp
g++ -Wall -c SendReceive.cpp
g++ -Wall -c ServerChild.cpp
g++ -Wall -o Server Server.o SendReceive.o ServerChild.o
g++ -Wall -std=c++11 -pthread -c Client.cpp
g++ -Wall -std=c++11 -pthread -o Client Client.o SendReceive.o
[croscoe@dhcp-142-232-155-178 Assignment2]$
```

After they have been compiled, you run the client by typing ./Client into the terminal and following along with the instructions. You run the server by typing ./Server into the terminal.



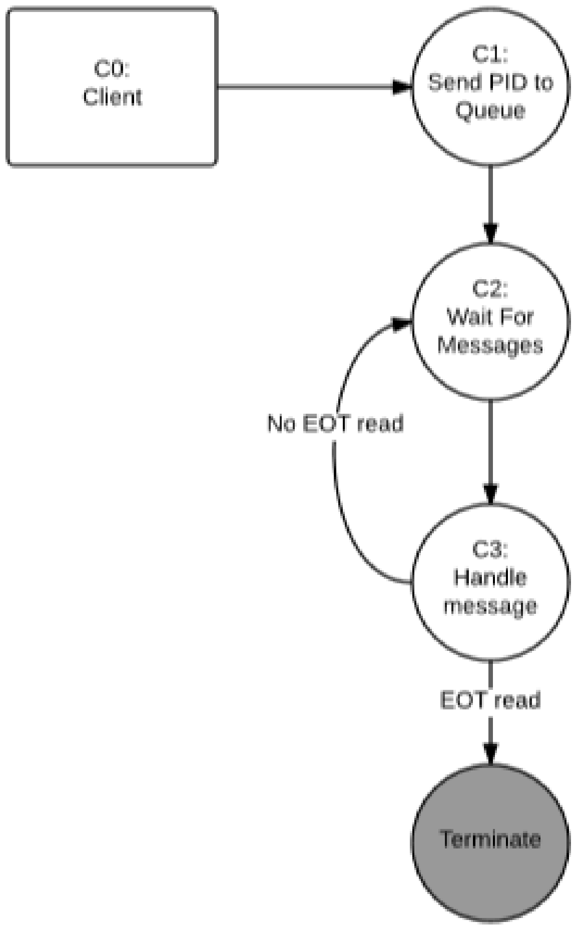
```
[croscoe@dhcp-142-232-155-178 Assignment2]$ ./Client
```



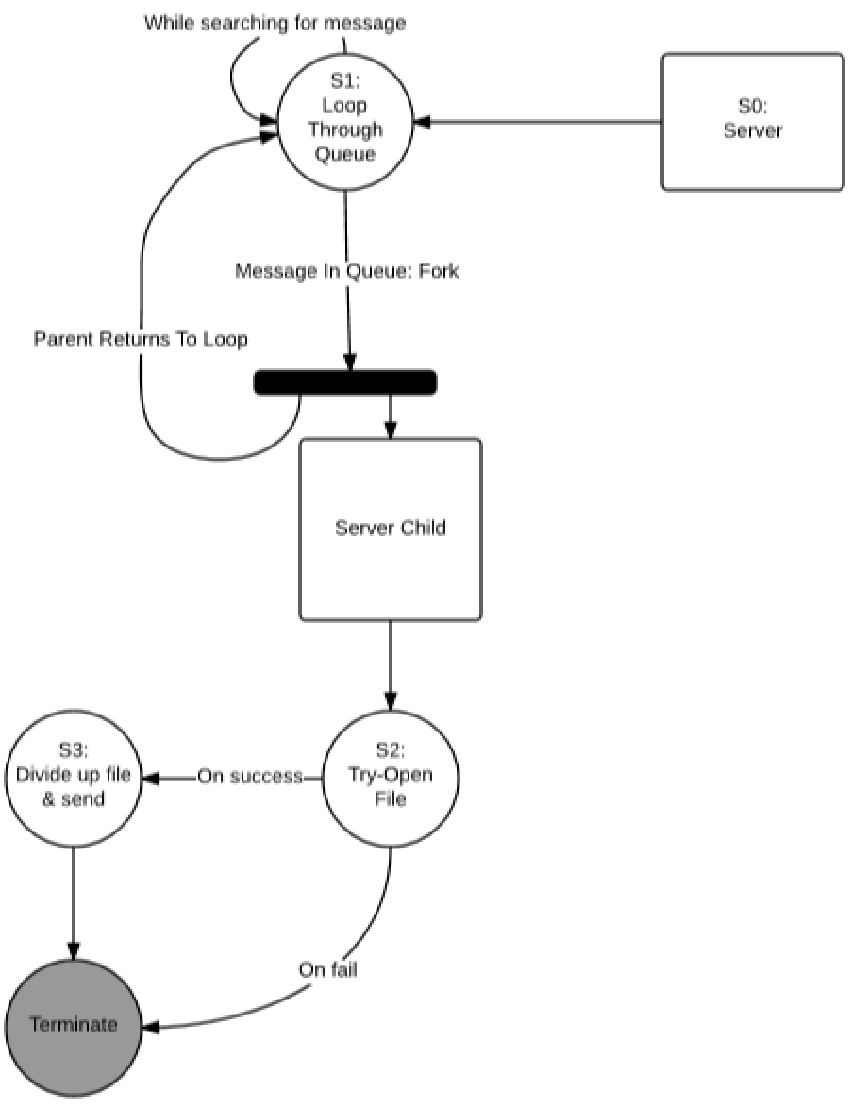
```
[croscoe@dhcp-142-232-155-178 Assignment2]$ ./Server
```

State Diagrams

CLIENT DIAGRAMS



SERVER DIAGRAMS


While searching for message
S1: Loop Through Queue
S0: Server
Message In Queue: Fork
Parent Returns To Loop
Server Child
S3: Divide up file & send
On success
S2: Try-Open File
On fail
Terminate

Pseudocode

Client Pseudocode

C0: Client

```
    if arguments passed != 2
        error stating to add a file name parameter
        exit
    messageQID = openQueue(key)
    if messageQID < 0
        error stating it failed
        exit
    C1()
```

C1: Send PID to Queue

```
    while(true)
        if (sendMessage(messageQID, data) == -1)
            error stating failed to send
            exit
        while(true)
            C2()
            C3()
```

C2: Wait For Messages

```
    if (readMessage(type, messageQID, data) == -1)
        error stating failed to read message
        exit
```

C3: Handle Message

```
    if (data == EOT)
        print(File done sending)
        exit
    else
        print(data)
```

Server Pseudocode

S0: Server

```
    messageQID = openQueue(key)
    if messageQID < 0
        error stating it failed
        exit
    S1()
```

S1: Loop Through Queue

```
    while(true)
        readMessage(type, messageQID, data)
        clientPID = data;
        if (fork() == 0)
            S2()
            exit
```

S2: Try-Open File

```
    if (openFile(file) == false)
        error failed to open file
        exit
    S3()
```

S3: Divide Up File & Send

```
    while(true)
        fileData = read set amount of bytes from file
        sendMessage(clientPID, fileData)
    sendMessage(clientPID, EOT)
    exit
```

Tests Summary

Screenshots and more information about specific tests can be found in the section below correlation to the section number column of any specific test.

Section #	**Description**	**Test**	**Expected Output**	**Success**
1	Client runs without crashing	Run the program	The program does not crash upon starting.	Passed
2	Server runs without crashing	Run the program	The program does not crash upon starting.	Passed
3	Client prompts for file name & priority	Run the client and enter a file name & number between 1 and 3	The program should display the prompts and accept the input assuming valid	Passed
4	Invalid file names are rejected by the client	Run the client and enter an invalid file name	An error message will appear on both the client & the server stating the file name was invalid	Passed
5	Invalid priorities prompt the user of proper usage & ask again	Run the client and enter an invalid priority	The client will see a usage explanation and will be asked to put in another filename/priority.	Passed
6	Valid file names & priorities will have the server send lots of data to the client regarding the file name sent	Run the client, enter “100mb.txt” and any priority from 1-3	Lots of data of numbers should appear, as the file contains numbers	Passed
7	Client and server acknowledge when a file transfer is completer	Run test #6 and wait for the transfer to finish.	Client program should exit and the server should have a message staying which PID has finished the transfer of the file	Passed
8	The server can handle more than one client request at separate times	Run test #7 multiple times with varying priorities.	The server program should display three separate priority numbers & transfer finished messages one after the other	Passed
9	The server can handle more than one clients requests simultaneously	Run test #7, however with multiple clients at once	The server should print more than one priority, and then show them one after another exiting. All files should be sent successfully	Passed
10	The server handles priority, where if you start two clients at the same time to read the same file, but one has a	Run test #7 on 2-3 clients at the same time. Start at the same time. Same	Priority 3 should exit before priority 2, and priority 2 should exit before priority 1.	Passed

	higher priority, that higher priority should finish first.	file name, just priority differs.		
11	If two clients are running and one force exits, the other clients should not be affected and the server should clean up all leftover messages.	Run test #9, however on one client hit control-C mid transfer.	The other client should still finish receiving data, and typing IPCS should show 0 messages in the queue.	Passed
12	If the server force exits, the clients should all exit as well being shown an error, and the message queue should be cleaned up.	Run test # 9 and his control-C on the server.	Server should exit, clients should exit with an error message & typing IPCS should show the message queue has been cleaned up.	Passed

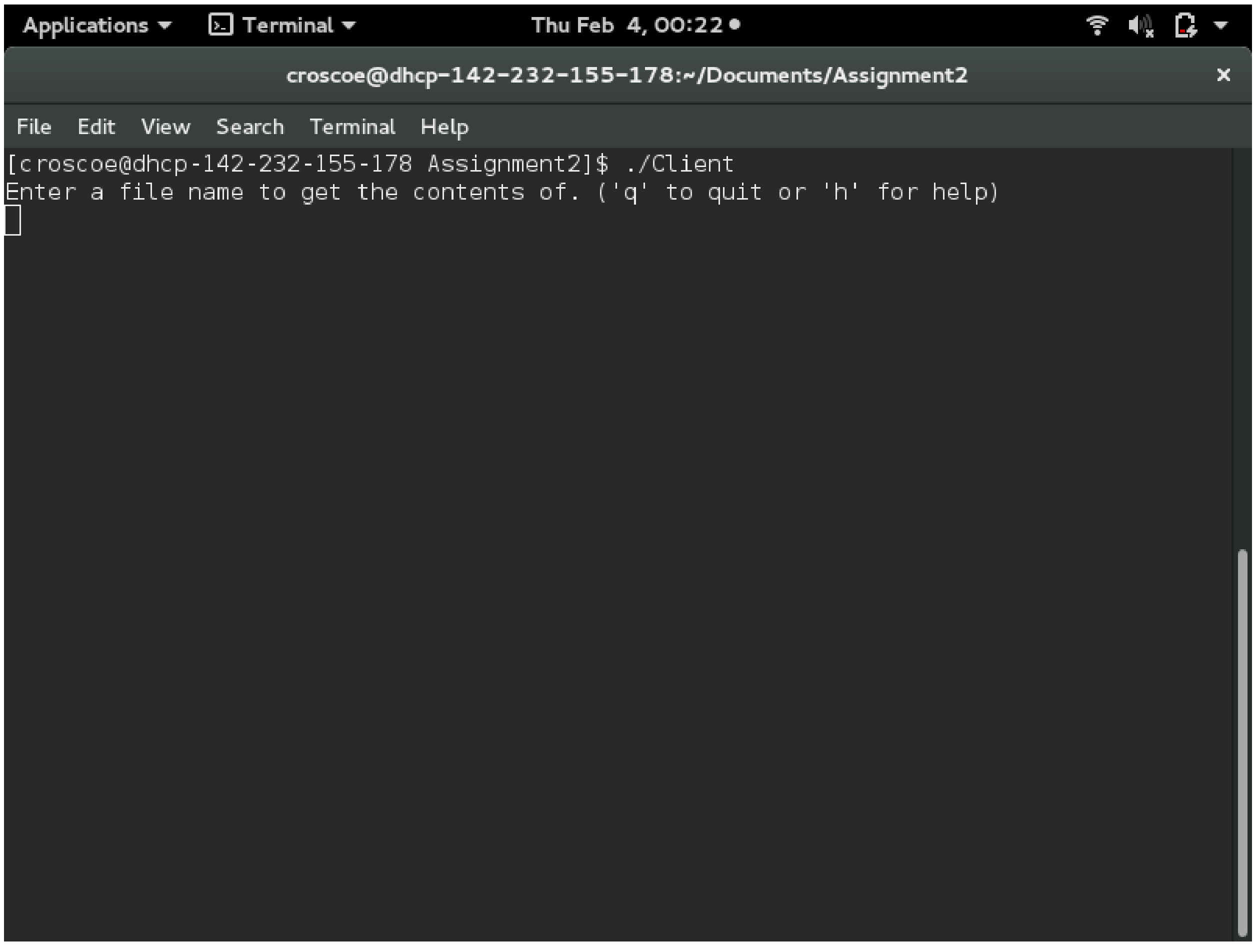
Test 1) Client runs without crashing

Test Explanation: The client program should be able to run without crashing

Expected Output: The program does not crash upon starting.

Result: Passed

FIGURE 1: Program Output


Applications
Terminal
Thu Feb 4, 00:22
croscoe@dhcp-142-232-155-178:~/Documents/Assignment2
File Edit View Search Terminal Help

```
[croscoe@dhcp-142-232-155-178 Assignment2]$ ./Client
Enter a file name to get the contents of. ('q' to quit or 'h' for help)
```

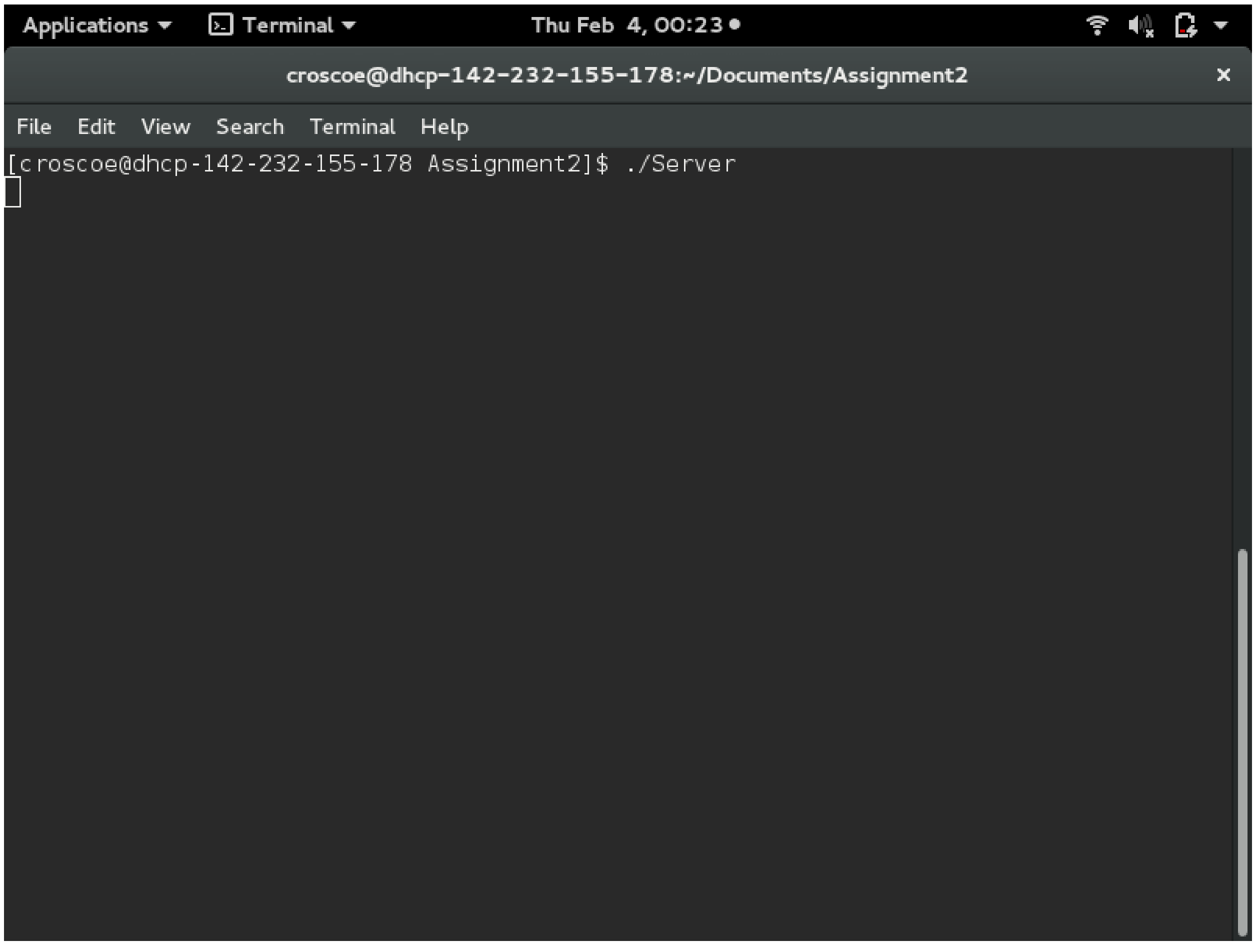
Test 2) Server runs without crashing

Test Explanation: The server program runs without crashing, it simply hangs for a message from the client.

Expected Output: The program does not crash upon starting.

Result: Passed

FIGURE 2: Program Output


Applications
Terminal
Thu Feb 4, 00:23
croscoe@dhcp-142-232-155-178:~/Documents/Assignment2
File Edit View Search Terminal Help

```
[croscoe@dhcp-142-232-155-178 Assignment2]$ ./Server
```

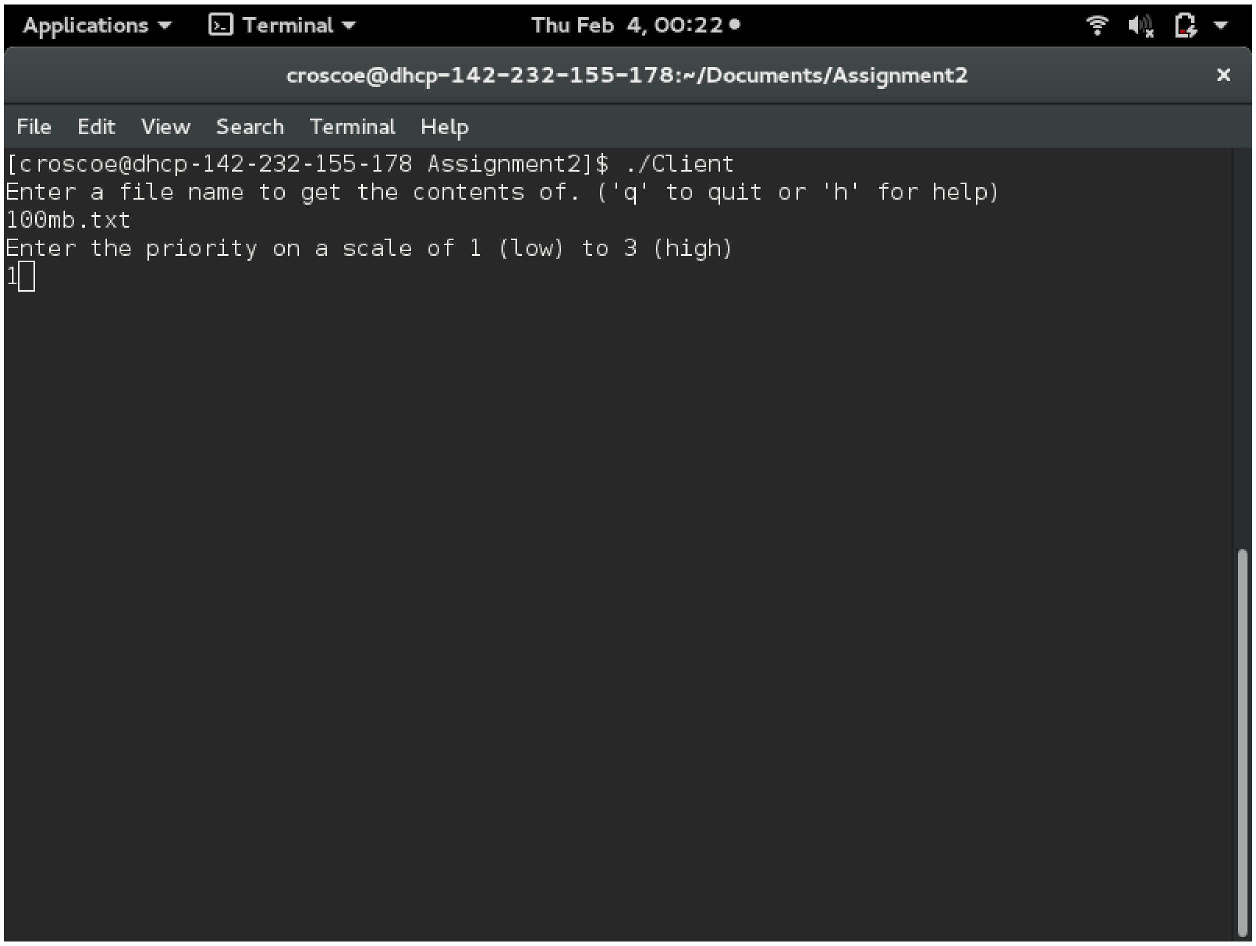
Test 3) Client prompts for file name & priority

Test Explanation: When the client starts it asks for a file name and after giving it the file name it asks for priority. These output messages should show their requests and it will let you input messages.

Expected Output: The program should display the two prompts & accept user input at the appropriate times

Result: **Passed**

FIGURE 3: Output below showing it



```
[croscoe@dhcp-142-232-155-178 Assignment2]$ ./Client
Enter a file name to get the contents of. ('q' to quit or 'h' for help)
100mb.txt
Enter the priority on a scale of 1 (low) to 3 (high)
1
```

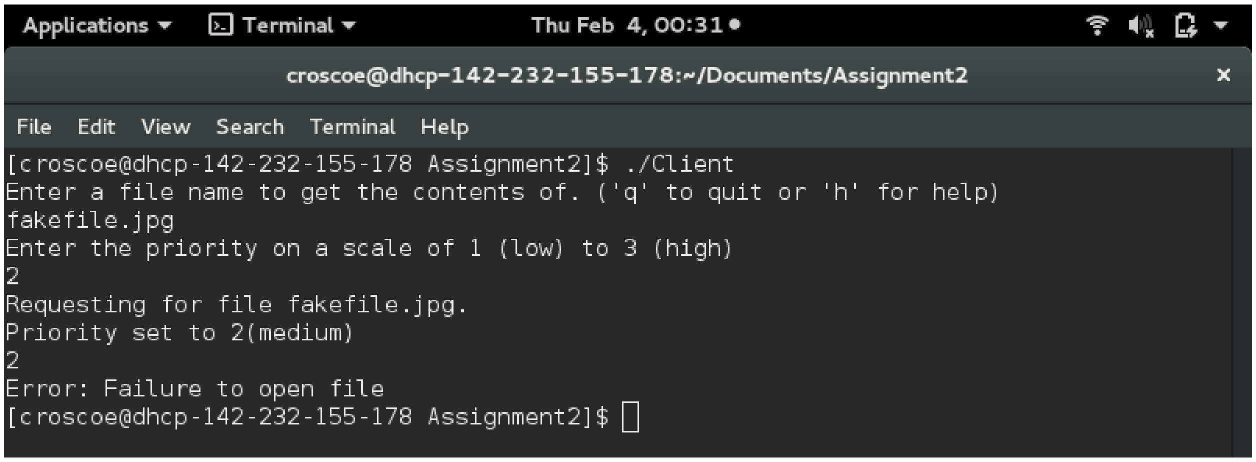
Test 4) Server rejects invalid file names

Test Explanation: If the client sends the server an invalid file name, the server will reply with an error message.

Expected Output: Client & server display an error message. The client will close, however the server will remain open.

Result: Passed

FIGURE 4: Client output, showing at the bottom that there was an error opening the file.


Applications
Terminal
Thu Feb 4, 00:31
croscoe@dhcp-142-232-155-178:~/Documents/Assignment2
File Edit View Search Terminal Help

```
[croscoe@dhcp-142-232-155-178 Assignment2]$ ./Client
Enter a file name to get the contents of. ('q' to quit or 'h' for help)
fakefile.jpg
Enter the priority on a scale of 1 (low) to 3 (high)
2
Requesting for file fakefile.jpg.
Priority set to 2(medium)
2
Error: Failure to open file
[croscoe@dhcp-142-232-155-178 Assignment2]$
```

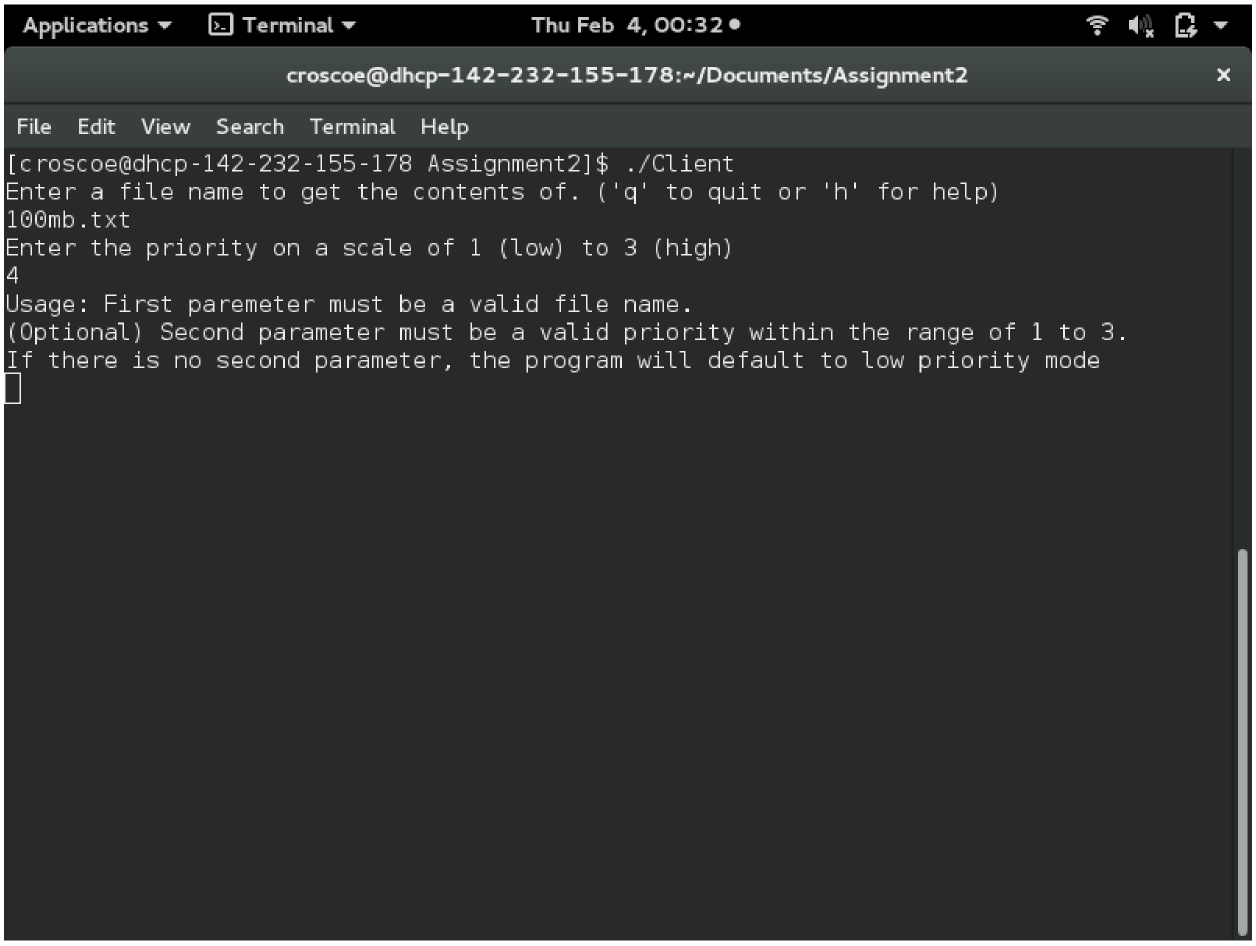
Test 5) Client setting invalid priority will prompt for proper usage

Test Explanation: The client handles invalid priority. Type a priority outside the given range

Expected Output: After typing priority 4, a usage explanation should appear.

Result: Passed

FIGURE 5: Program output displaying proper usage after invalid usage



```
[croscoe@dhcp-142-232-155-178 Assignment2]$ ./Client
Enter a file name to get the contents of. ('q' to quit or 'h' for help)
100mb.txt
Enter the priority on a scale of 1 (low) to 3 (high)
4
Usage: First paremeter must be a valid file name.
(Optional) Second parameter must be a valid priority within the range of 1 to 3.
If there is no second parameter, the program will default to low priority mode
```

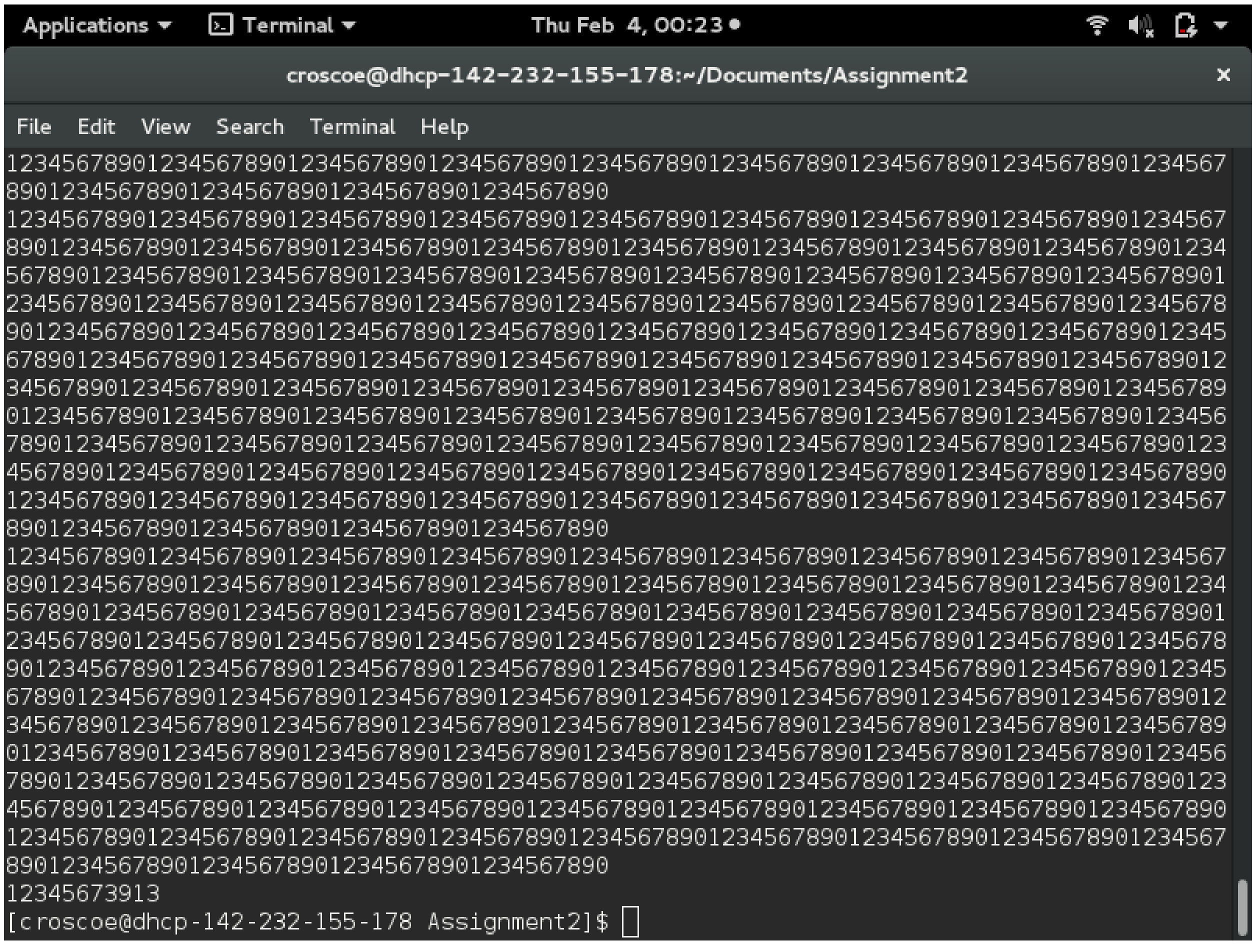
Test 6) Client receives data from server

Test Explanation: Client asking for a valid file name & priority ends with the client receiving lots of information regarding the file requested.

Expected Output: The entire screen should be flooded with information/numbers.

Result: Passed

FIGURE 6: Program output of receiving packets



```
12345678901234567890123456789012345678901234567890123456789012345678901234567
8901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567
89012345678901234567890123456789012345678901234567890123456789012345678901234
56789012345678901234567890123456789012345678901234567890123456789012345678901
23456789012345678901234567890123456789012345678901234567890123456789012345678
90123456789012345678901234567890123456789012345678901234567890123456789012345
67890123456789012345678901234567890123456789012345678901234567890123456789012
34567890123456789012345678901234567890123456789012345678901234567890123456789
01234567890123456789012345678901234567890123456789012345678901234567890123456
78901234567890123456789012345678901234567890123456789012345678901234567890123
45678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567
8901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567
89012345678901234567890123456789012345678901234567890123456789012345678901234
56789012345678901234567890123456789012345678901234567890123456789012345678901
23456789012345678901234567890123456789012345678901234567890123456789012345678
90123456789012345678901234567890123456789012345678901234567890123456789012345
67890123456789012345678901234567890123456789012345678901234567890123456789012
34567890123456789012345678901234567890123456789012345678901234567890123456789
01234567890123456789012345678901234567890123456789012345678901234567890123456
78901234567890123456789012345678901234567890123456789012345678901234567890123
45678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567
8901234567890123456789012345678901234567890
12345673913
[croscoe@dhcp-142-232-155-178 Assignment2]$
```

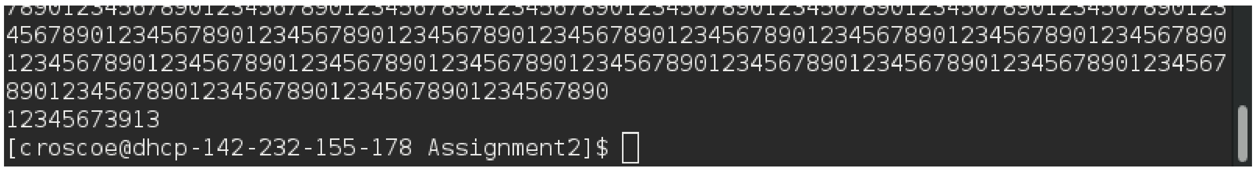
Test 7) Client and server acknowledge when a transfer is complete

Test Explanation: After transferring a file, both client and server acknowledge it is finished. Run test #6 again but wait for it to finish.

Expected Output: Client should exit upon completion, server should state the clients PID has finished and hang for more input.

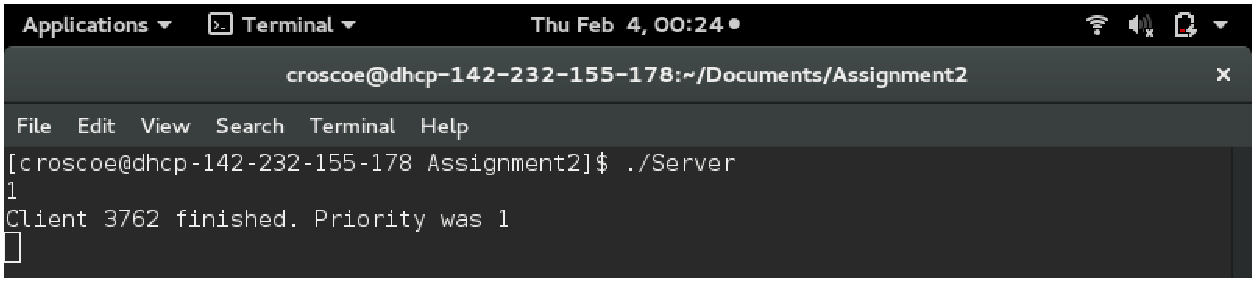
Result: Passed

FIGURE 7a: Client program output



```
7890123456789012345678901234567890123456789012345678901234567890123456789012345678901234
4567890123456789012345678901234567890123456789012345678901234567890123456789012345678901
1234567890123456789012345678901234567890123456789012345678901234567890123456789012345678
8901234567890123456789012345678901234567890
12345673913
[croscoe@dhcp-142-232-155-178 Assignment2]$ 
```

FIGURE 7b: Server program output


Applications
Terminal
Thu Feb 4, 00:24
croscoe@dhcp-142-232-155-178:~/Documents/Assignment2
File Edit View Search Terminal Help

```
[croscoe@dhcp-142-232-155-178 Assignment2]$ ./Server
1
Client 3762 finished. Priority was 1
```

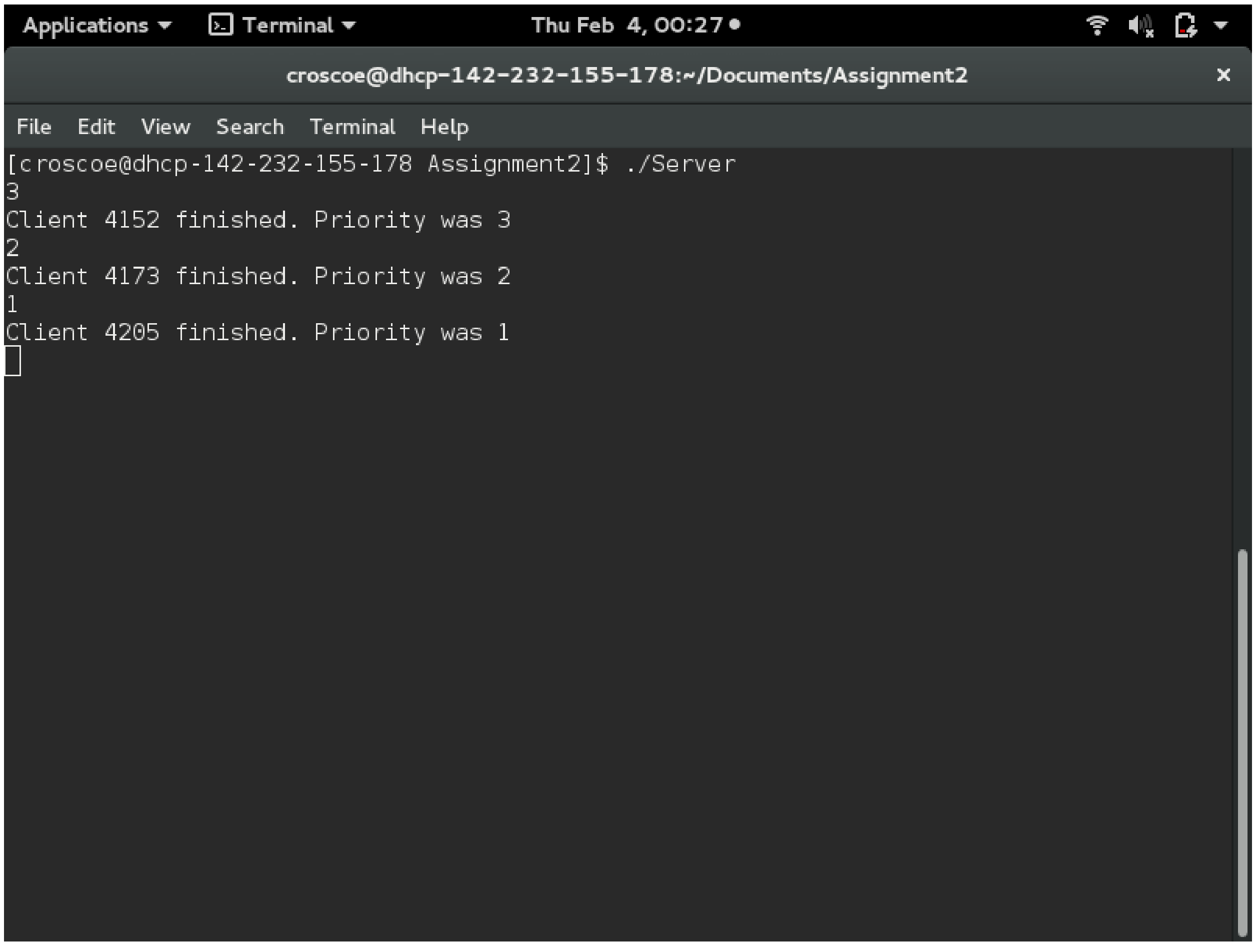
Test 8) Server can handle multiple client requests at separate times

Test Explanation: Redo test #7 multiple times. The server should handle them all equally and not lose functionality after finishing one.

Expected Output: The server should say all tests started & finished

Result: Passed

FIGURE 8: Server output


Applications
Terminal
Thu Feb 4, 00:27
croscoe@dhcp-142-232-155-178:~/Documents/Assignment2
File Edit View Search Terminal Help

```
[croscoe@dhcp-142-232-155-178 Assignment2]$ ./Server
3
Client 4152 finished. Priority was 3
2
Client 4173 finished. Priority was 2
1
Client 4205 finished. Priority was 1
```

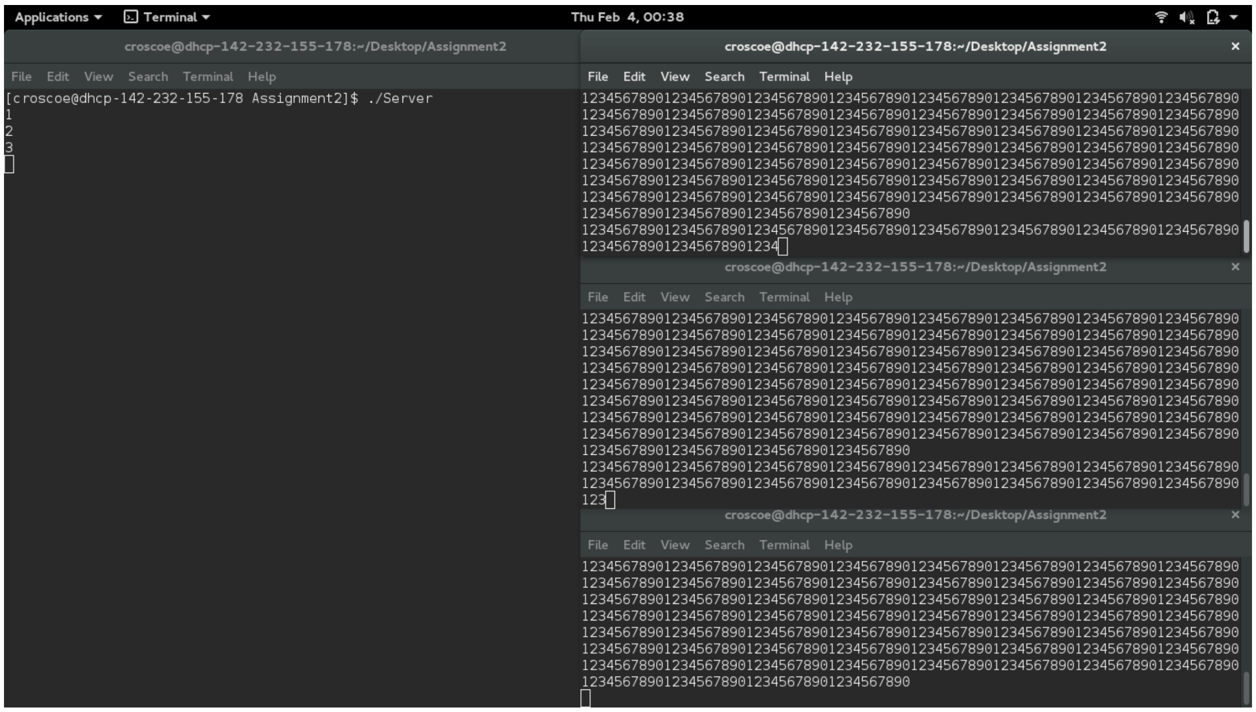
Test 9) Server can handle multiple client request simultaneously

Test Explanation: Run test #7 but on multiple clients at the same time.

Expected Output: The clients should all receive data & the server should handle all requests.

Result: **Passed**

FIGURE 9: Program output


Applications
Terminal
Thu Feb 4, 00:38
croscoe@dhcp-142-232-155-178:~/Desktop/Assignment2
File Edit View Search Terminal Help

```
[croscoe@dhcp-142-232-155-178 Assignment2]$ ./Server
1
2
3
```

```
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
1234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
123456789012345678901234
```

```
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
1234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
123
```

```
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
1234567890123456789012345678901234567890
```

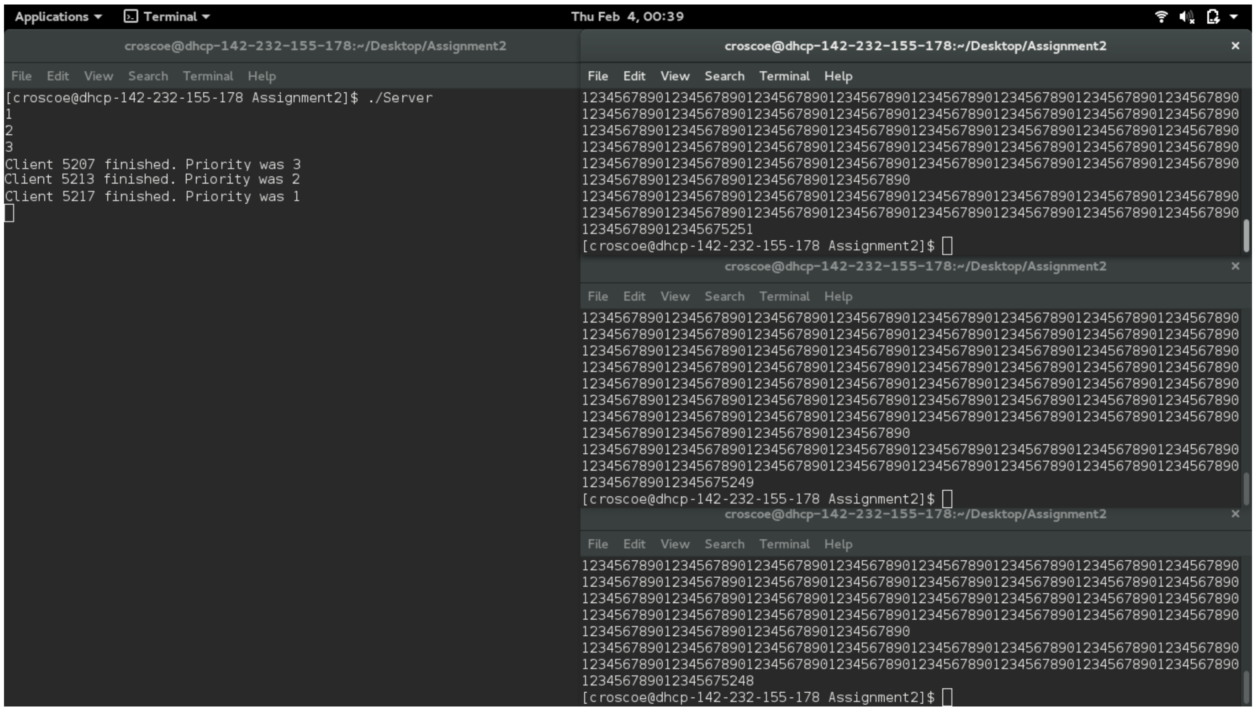
Test 10) Server handles priority

Test Explanation: The server handles priority, where if you start two clients at the same time to read the same file, but one has a higher priority, that higher priority should finish first. Run test #7 on 2-3 clients at the same time. Start at the same time. Same file name, just priority differs.

Expected Output: Higher priority clients should exit first

Result: Passed

FIGURE 10: Program output


Applications
Terminal
Thu Feb 4, 00:39
croscoe@dhcp-142-232-155-178:~/Desktop/Assignment2
File Edit View Search Terminal Help

```
[croscoe@dhcp-142-232-155-178 Assignment2]$ ./Server
1
2
3
Client 5207 finished. Priority was 3
Client 5213 finished. Priority was 2
Client 5217 finished. Priority was 1
```

```
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
1234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567 5251
[croscoe@dhcp-142-232-155-178 Assignment2]$
```

```
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
1234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
123456789012345675249
[croscoe@dhcp-142-232-155-178 Assignment2]$
```

```
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
1234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
123456789012345675248
[croscoe@dhcp-142-232-155-178 Assignment2]$
```

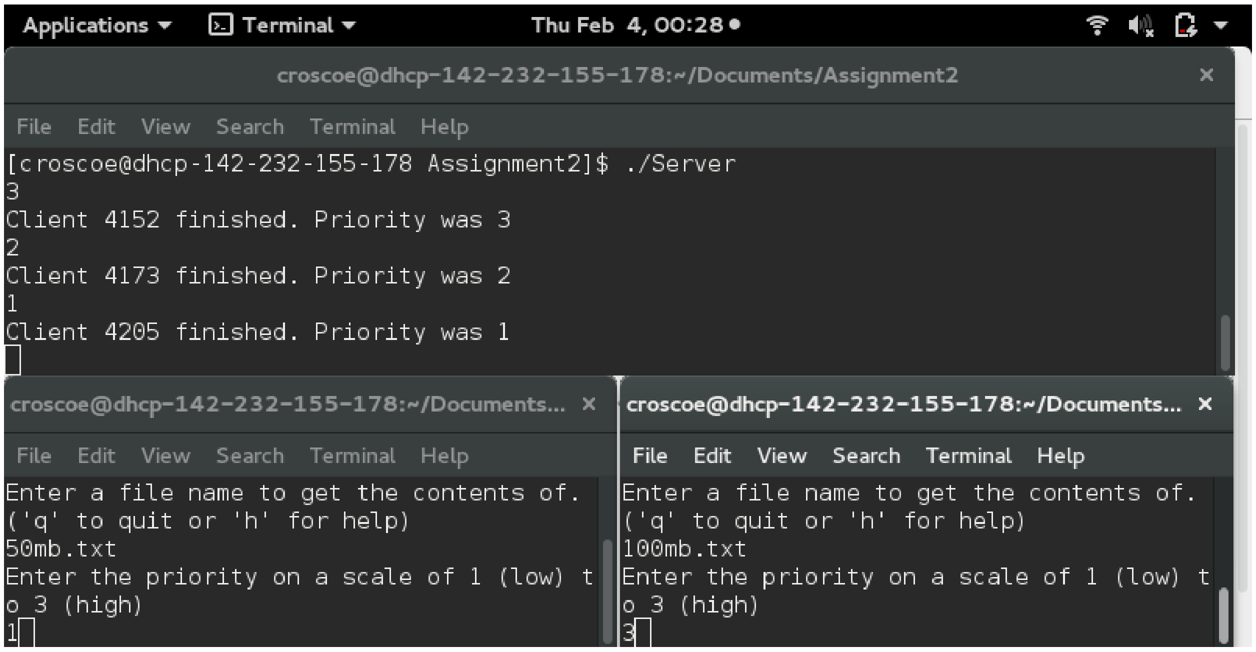
Test 11) Service Name & Protocol to Port Resolution

Test Explanation: If two clients are running and one force exits, the other clients should not be affected and the server should clean up all leftover messages. Run test #9 and control-C one of the clients.

Expected Output: One client exits, the rest are handled. Server acknowledges the exit and cleans up.

Result: Passed

FIGURE 11: Program output


Applications
Terminal
Thu Feb 4, 00:28
croscoe@dhcp-142-232-155-178:~/Documents/Assignment2
File Edit View Search Terminal Help
croscoe@dhcp-142-232-155-178:~/Documents...
File Edit View Search Terminal Help
croscoe@dhcp-142-232-155-178:~/Documents...
File Edit View Search Terminal Help

```
[croscoe@dhcp-142-232-155-178 Assignment2]$ ./Server
3
Client 4152 finished. Priority was 3
2
Client 4173 finished. Priority was 2
1
Client 4205 finished. Priority was 1
```

```
Enter a file name to get the contents of.
('q' to quit or 'h' for help)
50mb.txt
Enter the priority on a scale of 1 (low) t
o 3 (high)
1
```

```
Enter a file name to get the contents of.
('q' to quit or 'h' for help)
100mb.txt
Enter the priority on a scale of 1 (low) t
o 3 (high)
3
```

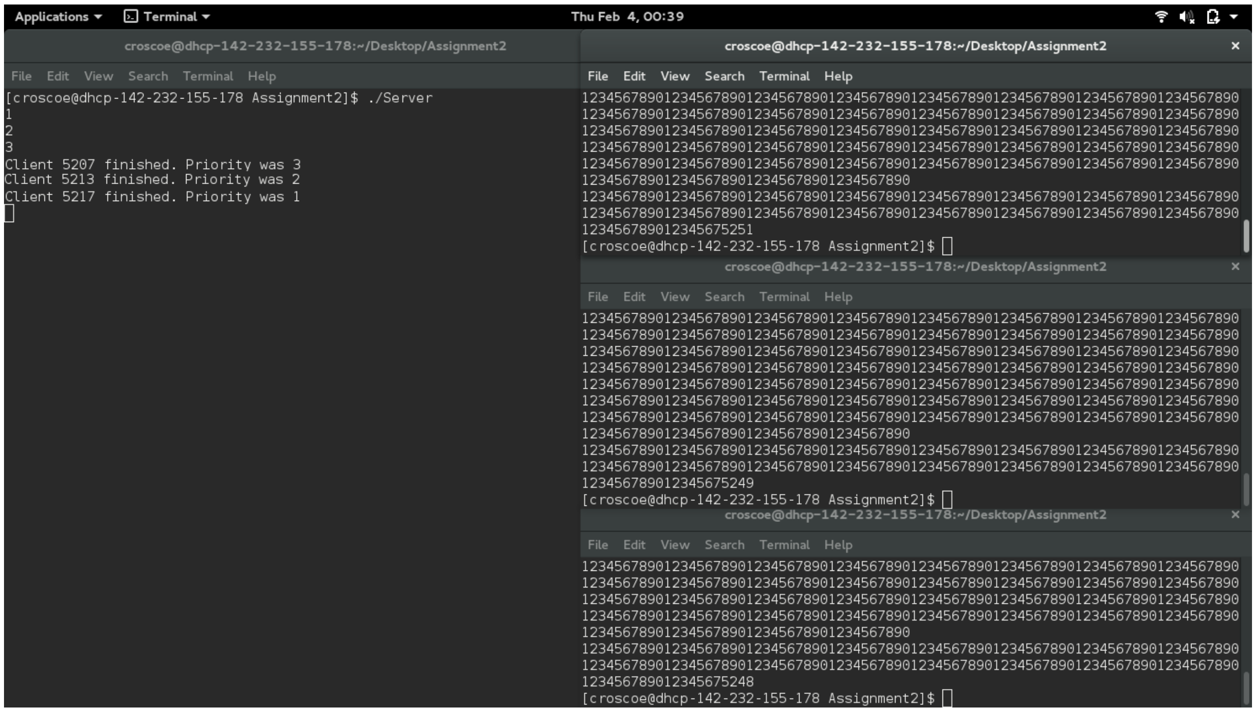
Test 12) Exiting the server will exit clients & cleanup

Test Explanation: If the server force exits, the clients should all exit as well being shown an error, and the message queue should be cleaned up. Run test #9 and control-C the server side. The clients should close.

Expected Output: Clients should close with an error, server closes. IPCS shows the message queue was closed

Result: Passed

FIGURE 10a: Program output



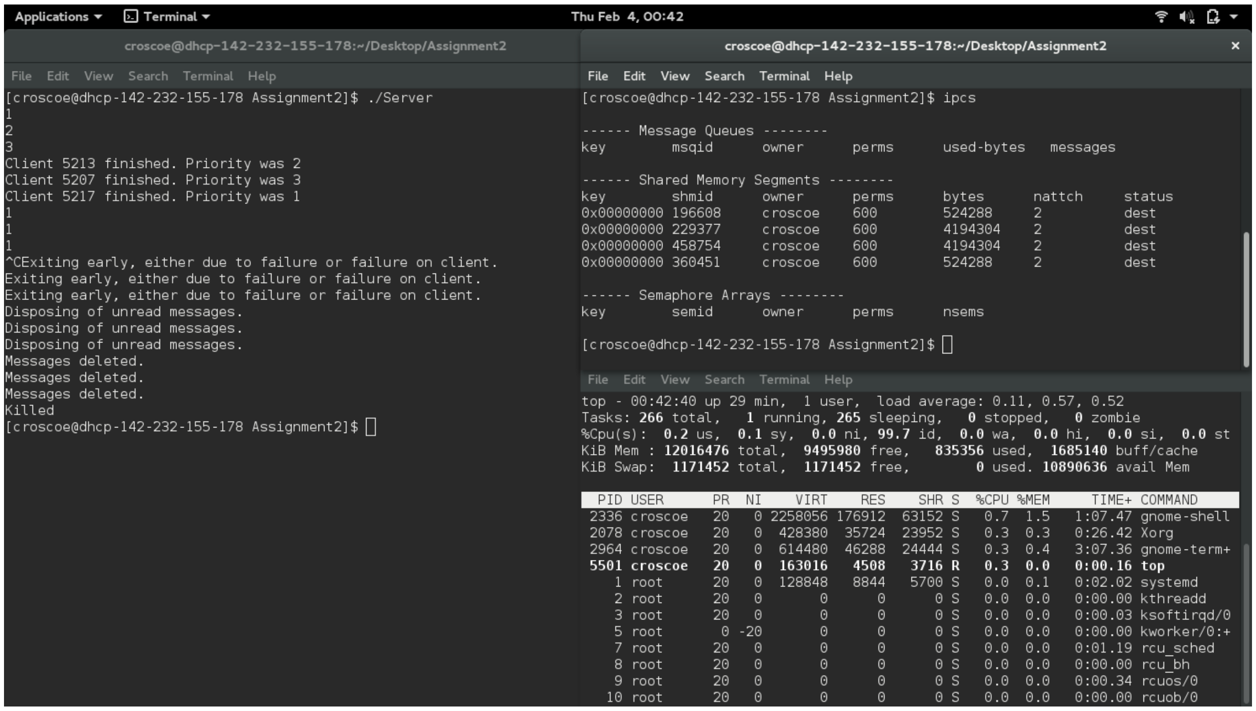
Applications
Terminal
Thu Feb 4, 00:39
croscoe@dhcp-142-232-155-178:~/Desktop/Assignment2
File Edit View Search Terminal Help

```
[croscoe@dhcp-142-232-155-178 Assignment2]$ ./Server
1
2
3
Client 5207 finished. Priority was 3
Client 5213 finished. Priority was 2
Client 5217 finished. Priority was 1
```

```
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
1234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
123456789012345675251
[croscoe@dhcp-142-232-155-178 Assignment2]$
```

```
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
1234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
123456789012345675249
[croscoe@dhcp-142-232-155-178 Assignment2]$
```

```
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
1234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
12345678901234567890123456789012345678901234567890123456789012345678901234567890
123456789012345675248
[croscoe@dhcp-142-232-155-178 Assignment2]$
```


Applications
Terminal
Thu Feb 4, 00:42
croscoe@dhcp-142-232-155-178:~/Desktop/Assignment2
File Edit View Search Terminal Help
croscoe@dhcp-142-232-155-178:~/Desktop/Assignment2
File Edit View Search Terminal Help
File Edit View Search Terminal Help

```
[croscoe@dhcp-142-232-155-178 Assignment2]$ ./Server
1
2
3
Client 5213 finished. Priority was 2
Client 5207 finished. Priority was 3
Client 5217 finished. Priority was 1
1
1
1
^CExiting early, either due to failure or failure on client.
Exiting early, either due to failure or failure on client.
Exiting early, either due to failure or failure on client.
Disposing of unread messages.
Disposing of unread messages.
Disposing of unread messages.
Messages deleted.
Messages deleted.
Messages deleted.
Killed
[croscoe@dhcp-142-232-155-178 Assignment2]$
```

```
[croscoe@dhcp-142-232-155-178 Assignment2]$ ipcs

------ Message Queues --------
key        msqid      owner      perms      used-bytes   messages

------ Shared Memory Segments --------
key        shmid      owner      perms      bytes      nattch     status
0x00000000 196608     croscoe    600        524288     2          dest
0x00000000 229377     croscoe    600        4194304    2          dest
0x00000000 458754     croscoe    600        4194304    2          dest
0x00000000 360451     croscoe    600        524288     2          dest

------ Semaphore Arrays --------
key        semid      owner      perms      nsems

[croscoe@dhcp-142-232-155-178 Assignment2]$
```

```
top - 00:42:40 up 29 min,  1 user,  load average: 0.11, 0.57, 0.52
Tasks: 266 total,   1 running, 265 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.2 us,  0.1 sy,  0.0 ni, 99.7 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem : 12016476 total,  9495980 free,   835356 used,  1685140 buff/cache
KiB Swap:  1171452 total,  1171452 free,        0 used. 10890636 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
 2336 croscoe   20   0 2258056 176912  63152 S   0.7  1.5   1:07.47 gnome-shell
 2078 croscoe   20   0  428380  35724  23952 S   0.3  0.3   0:26.42 Xorg
 2964 croscoe   20   0  614480  46288  24444 S   0.3  0.4   3:07.36 gnome-term+
 5501 croscoe   20   0  163016   4508   3716 R   0.3  0.0   0:00.16 top
    1 root      20   0  128848   8844   5700 S   0.0  0.1   0:02.02 systemd
    2 root      20   0       0      0      0 S   0.0  0.0   0:00.00 kthreadd
    3 root      20   0       0      0      0 S   0.0  0.0   0:00.03 ksoftirqd/0
    5 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 kworker/0:+
    7 root      20   0       0      0      0 S   0.0  0.0   0:01.19 rcu_sched
    8 root      20   0       0      0      0 S   0.0  0.0   0:00.00 rcu_bh
    9 root      20   0       0      0      0 S   0.0  0.0   0:00.34 rcuos/0
   10 root      20   0       0      0      0 S   0.0  0.0   0:00.00 rcuob/0
```