

# CPSC\_351\_Assignment\_1

\*\*\_CPSC 351- Section 2(13643)\_\*\*

\*\*Spring 2019\*\*

\*\*Group Members:\*\*

James I Ku  
894841865  
thatoneddrguy@csu.fullerton.edu

Esteban Montelongo  
888847456  
EstebanMontelongo@csu.fullerton.edu

Bony Roy  
898161054  
broy91@csu.fullerton.edu

\*\*Purpose:\*\*

Use shared memory, and message queues in order to implement an application which synchronously transfers files between two processes.

\*\*Programming Language Used:\*\*

C++

\*\*Extra Credit:\*\*

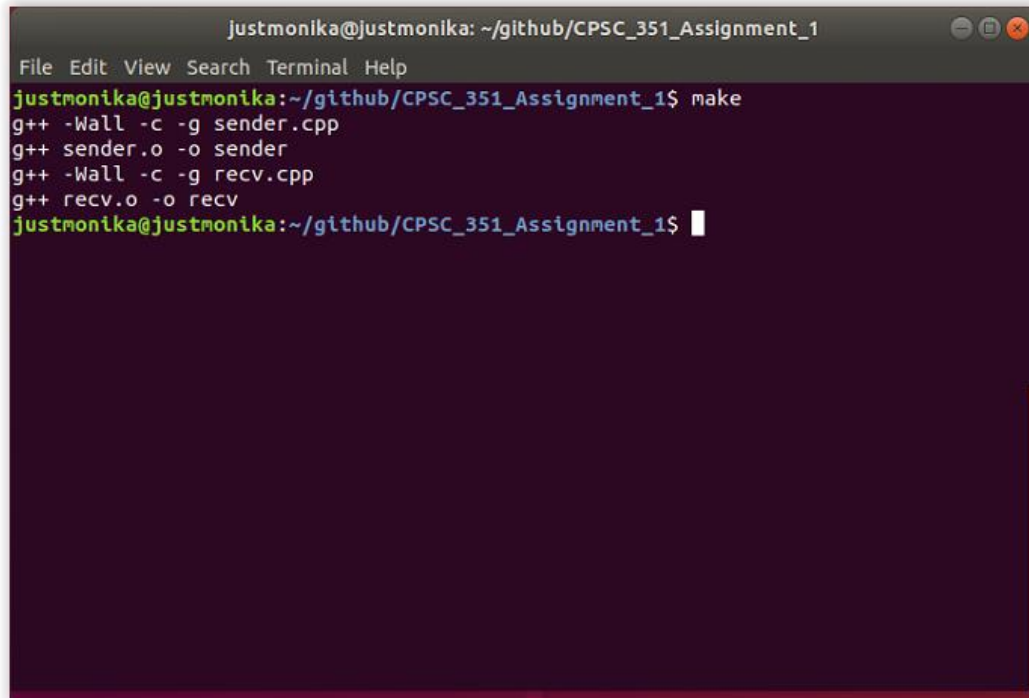
Not Implemented

\*\*File Names:\*\*

1. p1-roykumont.tar
2. sender.cpp: CPP File that sends the message from the text file to recv.cpp
3. recv.cpp : CPP File the receives the message from the sender.cpp
4. keyfile.txt: Text file that holds a string that is used to generate the same key for both the sender and the receiver.
5. msg.h: Header File that holds a struct of the message relayed through the message queues
6. Makefile: Makefile to build both the sender and receiver files

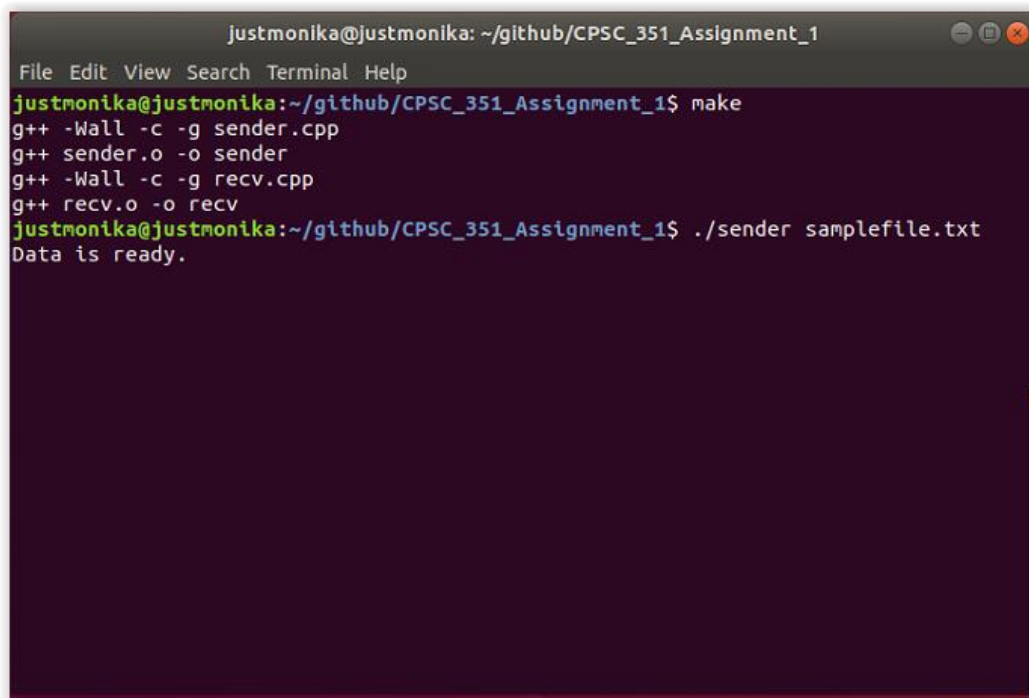
**\*\*To run our program:\*\***

1. Download tar archive.
2. Extract files from tar archive.
3. Open terminal in directory where files are located.
4. In terminal type make and press enter.



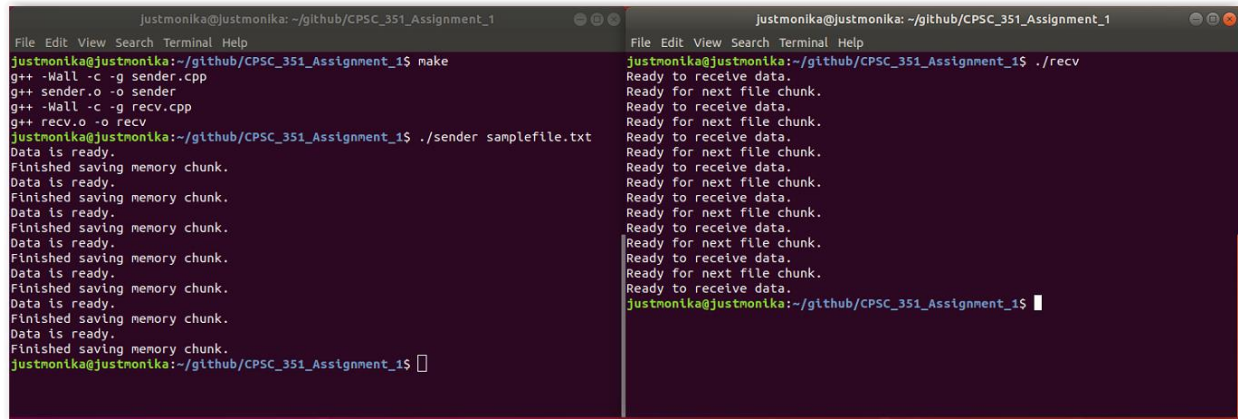
```
justmonika@justmonika: ~/github/CPSC_351_Assignment_1
File Edit View Search Terminal Help
justmonika@justmonika:~/github/CPSC_351_Assignment_1$ make
g++ -Wall -c -g sender.cpp
g++ sender.o -o sender
g++ -Wall -c -g recv.cpp
g++ recv.o -o recv
justmonika@justmonika:~/github/CPSC_351_Assignment_1$
```

5. Sender: open terminal in directory that contains the sender file. run : type ./sender <name of file>,press enter.



```
justmonika@justmonika: ~/github/CPSC_351_Assignment_1
File Edit View Search Terminal Help
justmonika@justmonika:~/github/CPSC_351_Assignment_1$ make
g++ -Wall -c -g sender.cpp
g++ sender.o -o sender
g++ -Wall -c -g recv.cpp
g++ recv.o -o recv
justmonika@justmonika:~/github/CPSC_351_Assignment_1$ ./sender samplefile.txt
Data is ready.
```

6. Receiver: open new terminal in directory containing file. run: type ./recv, press enter. Received file is saved as recvfile.



The image shows two terminal windows side-by-side. The left window is titled 'justmonika@justmonika: ~/github/CPSC\_351\_Assignment\_1' and shows the compilation of 'sender.cpp' and 'recv.cpp' using 'g++'. It then runs './sender samplefile.txt', which outputs a series of 'Data is ready.' and 'Finished saving memory chunk.' messages. The right window is also titled 'justmonika@justmonika: ~/github/CPSC\_351\_Assignment\_1' and shows the execution of './recv', which outputs a series of 'Ready to receive data.' and 'Ready for next file chunk.' messages.

```
justmonika@justmonika: ~/github/CPSC_351_Assignment_1
File Edit View Search Terminal Help
justmonika@justmonika:~/github/CPSC_351_Assignment_1$ make
g++ -Wall -c -g sender.cpp
g++ sender.o -o sender
g++ -Wall -c -g recv.cpp
g++ recv.o -o recv
justmonika@justmonika:~/github/CPSC_351_Assignment_1$ ./sender samplefile.txt
Data is ready.
Finished saving memory chunk.
Data is ready.
Finished saving memory chunk.
Data is ready.
Finished saving memory chunk.
Data is ready.
Finished saving memory chunk.
Data is ready.
Finished saving memory chunk.
Data is ready.
Finished saving memory chunk.
Data is ready.
Finished saving memory chunk.
justmonika@justmonika:~/github/CPSC_351_Assignment_1$

justmonika@justmonika: ~/github/CPSC_351_Assignment_1
File Edit View Search Terminal Help
justmonika@justmonika:~/github/CPSC_351_Assignment_1$ ./recv
Ready to receive data.
Ready for next file chunk.
Ready to receive data.
Ready for next file chunk.
Ready to receive data.
Ready for next file chunk.
Ready to receive data.
Ready for next file chunk.
Ready to receive data.
Ready for next file chunk.
Ready to receive data.
Ready for next file chunk.
Ready to receive data.
Ready for next file chunk.
Ready to receive data.
Ready for next file chunk.
justmonika@justmonika:~/github/CPSC_351_Assignment_1$
```

**\*\*Team Collaboration:\*\***

Our team met up three times at the CSUF library to research the necessary system function calls and collaborate on the code via GitHub.

**\*\*Sources:\*\***

Professor Yun Tian, Skeleton Code posted on Titanium and relevant links posted in document.