To the TA that is grading this,
Please give an hour or so lead way on this submission as we experienced the severe weather (Tornado warning) where we had an hour and a half period that we had to take shelter.

Blocked Message Send

My test case implementation is that the receiving process was created before the sending ones and contains an infinite while loop that calls receive and then sleeps for a set time to allow the FIFO sender queue to build up. I then created 4 sender processes to send messages to the 1 receiver. To verify correctness of my implementation, I would print any messages received in my main from the senders. They printed out in the same order they were created. If a receiver process terminates while there are processes in its blocked send queue, those processes should terminate the message and resume. A better solution would prevent a process to terminate when there are processes in its block send queue.

For concern of messing with Xinu's queue functions, I created my own to avoid messing them up. I based all code of my queue on Xinu's current implementation.

Asynchronous IPC With Callback Function

In the main, I called cbreg as given in the handout, which calls the user space function mrecv_cb. This function sets the flag having a callback function and then sets the user space address to the process's fptr variable. I used the code given in the handout as the base for my callback function naming. On context switch, I check to see if the process has a callback function, and if so, I call it, which in turn, calls receive to receive the messages sent by my test case send processes.