CMPT 300: Tutorial Notes #6

Assignment 3 Structure

What kind of data structures will you need for your simulation?
Possibilities:
- a process control block (PCB) structure
  - what does this structure have inside it?
    - a process ID (PID)
    - a priority (0, 1, or 2)
    - some way of representing the state of the process (e.g. running, ready, blocked)
    - may need a place to store any messages that another process "send"s or "reply"s to this process
- 3 ready queues, one for each type of priority
  - each queue is a list of what? Answer: PCB's
- a queue of processes waiting on a send operation (i.e. waiting for a reply)
- a queue of processes waiting on a receive operation
- semaphore data structure (you only need 5 semaphores)
  - what does this structure have inside it?
    - an integer representing the value of the semaphore
    - a list of processes waiting on that semaphore

There is also a special init process that exists as long as the simulation is running.
- init process only runs when no other processes are ready to execute (i.e. all ready queues are empty)
- init process can do anything other processes can do, but it never blocks (must deal with it as a special case for send/receive and semaphores)
- init process cannot be killed or exited unless it is the last process in the system (i.e. no processes on any ready queue or any blocked queue)
- once init process is killed/exited the simulation terminates

Notes about some of the commands:
- Kill (K) allows you to kill a process even when it is not currently executing (e.g. if it is on a blocked queue)
- Exit (E) only allows you to kill the currently executing process
- Quantum (Q) is the ONLY way to signal that the time quantum for round robin scheduling has expired
  - when this occurs we must choose the next process to execute from the appropriate ready queue (or just the init process if no processes are ready)
  - when a new process becomes ready of a higher priority than the currently executing process, you do NOT need to pre-empt the currently executing process; just wait until quantum expires
- Send (S) sends the message to the named process and places the sender on a blocked queue
  - you must put the message somewhere so that the named process will be able to receive it when it is next executed
- Receive (R) checks if there is a message waiting for the currently executing process, if there is it receives it, otherwise it gets blocked
- Reply (Y) delivers reply to sender (works similar to Send) and unblocks the sender
- Semaphore P or V operations: execute P or V operation on the named semaphore (just like in the course notes): may block or unblock a process