**Lab 3**

Tyler Holland (tyhollan)

1. User 2 will not be able to read the file, because it no longer exists. User 2 will get an whatever error is sent when that file does not exist.
2. Multiprogramming will increase CPU utilization when we want to run more than one process in "parallel". By switching back and forth between processes, each process will seem to be running at the same time, and using as much of the CPU as possible at a time.
3. Half the CPU capacity = 500,000,000 instructions. 1000 instructions = a system call. **50000 System calls.**
4. A race condition is when 2 programs are attempting to use the same variable, and the value of that variable depends on which program reaches that variable first. This is a problem because programs are not guaranteed to run at the same points in time each run, so your output could vary based on CPU load or other timing variables.
5. Yes it does work, because turn will either be 1 or 0, so one of the processes will always be able to run. This will only work though if it is just 2 processes, no more than that.
6. The up and down methods of the semaphore could be implemented as system calls, and the OS could briefly disable all interrupts while it tests the semaphore, updates it, and puts the process back to sleep. The rest of the semaphore runs as normal.
7. If a process occurred twice on the list, it would get another chance to run. One reason for doing this is if you have a process that you would like to run more often than the others, but still have round robin scheduling. You could put that one process multiple times in the list in order to give it a larger running time while leaving the other processes with normal running times.
8. 1. A = 30, B = 27, C = 13, D = 18 , E = 25
   2. A = 23, B = 7, C = 26, D = 30 , E = 13
   3. A = 10, B = 17, C = 20, D = 24 , E = 30
   4. A = 30, B = 20, C = 3, D = 7 , E = 13
9. 1. A = 30, B = 27, C = 13, D = 18 , E = 25
   2. A = 23, B = 7, C = 26, D = 30 , E = 13
   3. A = 10, B = 17, C = 20, D = 24 , E = 30
   4. A = 30, B = 20, C = 3, D = 7 , E = 13
10. It must be swapped in 5 times. It gets knocked down from priority 1 (the top) to priority 5, where it will finish.