**Comp 4735 Winter 2015**

## Lab Instructor: \_Mirela Gutica\_ SET :\_4D\_

**Name: \_Ignacio Marquez\_ Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

# Lab 3

Solve the following exercises. Work in pairs. Discuss each exercise with your lab instructor.

1. Discuss Figure 4.1. What is the difference between the four models?
   1. Single process, single thread/process
      1. in real world terms, this really just means one function is executed at a time
   2. multiple process, single thread/process
      1. multiple programs may run, but only one function at a time
   3. single process, multiple thread/process
      1. one program may run multiple functions
   4. multiple process, multiple thread/process
      1. multiple programs may run multiple functions
2. Discuss Figure 4.2. What is the difference between processes and threads?
   1. In a single process, single thread model, each process will have it's own process control block, user address space, user stack, and kernel stack
   2. In a multithreaded process, each process will have it's own process control block and user address space. Each thread will have it's own thread control block, user stack, and kernel stack
3. Discuss Figure 4.6.
   1. Trace the execution of Process B.
      1. 4.6a-4.6b Process B goes from Running state to Blocked state (possibly through I/O call)
      2. 4.6b-4.6c Kernel determines that Process B timeslice is through, switches it to Ready state
      3. 4.6d Process B starts Running state.
   2. Why are Thread 2 in the running state and Process B Blocked??
      1. Thread 2 is in the Running state only in name… it is actually not executing. Process B is in fact blocked.
4. What is the difference between ULT and KLT?

ULT – thread management is done by the application (kernel does not know about thread)

KLT – thread management is done by the kernel

1. What are the advantages and disadvantages of each model?
   1. ULT advantages
      1. Switching between ULT threads does not involve kernel mode. In fact kernel does not even know ULT exists. This saves lots of time.
      2. Scheduling algorithms can be customized to the particular application
      3. OS independent
   2. ULT disadvantages
      1. if the ULT makes a system call, the entire process will be blocked.
      2. ULT cannot take multiprocess
   3. KLT advantages
      1. KLT can multiprocess on multiple threads
      2. KLT can run another thread doing the same work as a blocked thread
      3. kernel can be multithreaded
   4. KLT disadvantages
      1. when KLT wants to switch threads within a process, it must first switch to kernel mode. This is expensive.
2. Solve problems: 4.4, 4.7, 4.8

4.4:

In this model, multithreaded programs run faster than their counterparts because the KLT can simultaneously issue blocking system calls and let the other threads continue. In a single thread single processor model, that one thread would need to do the blocking system call, wait for the timeslice to end, and then continue running.

4.7