**Comp 4735 Winter 2015**

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

**Name: \_\_\_\_\_ Ignacio Marquez \_\_\_\_\_\_\_\_\_\_ Signature: \_\_\_\_\_\_**

# Lab 2

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

1. Discuss Figures 3.2; 3.3 and 3.4.
   1. What is the role of dispatcher?
      1. To switch processes, and handling appropriate scheduling and time slicing of multiple processes
   2. What switch is perform between Line 6 and 7?
      1. A timeout (timer interrupt occurs for the process because the maximum time for the time slice allocated for the process is over, and switches over to the dispatcher for further switching
2. Discuss Figure 3.6. Describe the five states in details.
   1. New – When a process to created, it is put into this state, to further be admitted into the ready to execute state.
   2. Ready – When a process is in this state, it is on a ready to execute queue, be to further dispatched into the running state for a limited amount of time
   3. Running – When a process is in this state, the program code is in execution for a limited amount of time until a timer interrupt occurs and the dispatcher returns it back to the ready to execute state
   4. Blocked – A process enters this state when it starts waiting for a specific event to occur. When the event does occur, it will return to the ready to execute state.
   5. Exit – When a process is in the running state and is finished executing, it will be released into the Exit state. Memory will be deallocated.
3. Discuss Figure 3.7.
   1. Why does process C not start immediately after process B?
      1. The dispatcher has to come in to dispatch the next process in the ready to execute queue.
   2. Why is the time slice for process B shorter?
      1. Process B decided to wait on an event during its executed, thus placing it into the blocked state until an event occurs. This causes the dispatcher to execute afterwards to execute the next ready to execute process.
4. Discuss Figure 3.8.
   1. What is the advantage of having separate queues for events?
      1. Different kinds of events are usually common in most applications, which is better than having one queue, as one queue can handle blocking with one event.
      2. When the specific event for the specific queue occurs, it’s easier to pop the process from the front of the queue into the ready to execute state.

Discuss Figure 3.9b.

* 1. Why is a process suspended?
     1. swapping – OS does it when memory is needed for ready processes
     2. other os reason – OS will do it if process causing a problem
     3. interactive user request – user can suspend for debugging and such
     4. timing – process executed periodically, waiting for next time interval
     5. parent process request – parent suspends to examine/modify child
  2. Why does the process have two suspended states?
     1. The ready/suspend state, means the process is ready to execute, but is not active due a specific reason
     2. Blocked/suspend state, means the process is waiting on an event, but is not active
  3. Give examples of situations of transitions from states in Figure 3.9b.
     1. OS needs more memory, suspends idling process from the Ready state into the Ready/suspend state (swapping)
     2. Blocked process receives event, returns into the ready to execute state, the dispatcher dispatches it to the running state

1. Discuss the process control structures and their role in switching from a process to another at timeout.
   1. Process Control Block
      1. **Process Identification** – numeric id may be stored with PCB, pid of parent, user id
      2. **Processor State Information** - contents of process registers (user visible registers, control/status registers, stack pointers), program status word (PSW, status information(i.e.EFLAGS on Pentium)) – 4 the dispatcher
      3. **Process Control Information** – info needed to control multiple active processes, scheduling/state information, state: running/ready/waiting/halt, priority, scheduling,events, IPC(flags,signals,msgs), memory priveleges
   2. The PCB is needed to save/restore the state of a process (registers,flags) to perform efficient time slicing with the dispatcher.
2. Discuss in details switching from a process to another in the three cases from subchapter 3.5.
   1. What is the consequence of a separate kernel (figure 3.15a) if a process issues a system call?
      1. Process has little control over syscall execution due to there being one kernel stack for the entire OS
      2. Process time slice will have to end early and become blocked until the syscall is finished
   2. What are the advantages/disadvantages in each case?
      1. Separate kernel:
         1. advantages: each process does not need a kernel stack (less memory),
         2. disadvantage: process needs to be blocked every syscall and privilege management is difficult
      2. OS in user processes:
         1. advantages: control of syscall due to each process having a kernel stack, privilege management easier;
         2. disadvantage: a little more complex, process time is split into kernel time and user time
      3. OS as separate processes:
         1. advantages: better concurrency, all process time is user time;
         2. disadvantages: dispatcher has to dispatch more OS processes
3. Discuss process creation and termination. What should the OS do in these situations?

* Process Creation
  + assign pid
  + allocate space
  + init PCB
  + set linkages
  + creates/expands other data structures
* Process Termination
  + Deallocate memory
  + deallocate PCB