CMSC 412 Midterm #1 (Spring 2010) - Solution

1.) (20 points) Define and explain the following terms:

a) kernel

The highest privileged part of the Operating System. It is code that manages processes and physical devices.

b) critical section

A region of code accessing shared information that needs to be protected by mutual exclusion (i.e. only one process at a time is allowed to accesses it)

c) memory protection

OS/Hardware feature that prevents one process from reading or writing the memory of the kernel or another process

d) process control block (PCB)

The data structure storing the state of a process including registers, stack, memory protections, time used, etc.

2.) (20 Points) (20 points) (20 points) Synchronization: Given an implementation of binary semaphores (i.e. a V operation on a binary semaphore with a value of 1 does not change the value of the semaphore), implement counting semaphores. Show any variables or semaphores you use and their initial values

Semaphore and variable declarations:

```
Sempaphore mutex = 1;
Int sem = limit <initial limit provided)>
Int wait = 0;
Int turn = 0;
```

P():

```
P(mutex);
If (sem > 0) {
      Sem--;
      V(mutex);
} else {
      Wait++;
      V(mutex);
      P(turn);
}
```

V():

```
P(mutex);
If (sem == limit) {
      V(mutex);
} else if (!waiting) {
      Sem++;
      V(mutex);
} else {
      Waiting--;
      V(mutex);
      V(turn);
}
```

3.) (20 Points) Scheduling

a) Why would a scheduling algorithm that gave priority to processes based on how much time they have consumed (i.e. processes whose total CPU time is higher have higher priority) be a bad idea?

Using CPU time gives your more time. This promotes a counter strategy of starting a process and running for a long time so that it will have a high priority when it is ready to do real work. Also, run aways process have priority (but do not starve!) other processes.

b) Explain how short-term and medium-term scheduling are different? What issues are each concerned with?

Short term – dispatcher. Schedules the CPU with runnable processes. Time scale on the order of mili-seconds.

Medium term – schedules memory. Can be more complex since it is not as time critical. Time scale on the order of seconds.

4.) (12 points) Given a test-and-set instruction, write an implementation of swap(a,b) atomic instruction.

```
Swap(a,b)
      While (test-and-set(x)) {
            Temp = a;
            B = a;
            B = temp;
            X= 0;
      }
```

5.) (8 points) What are the four conditions required for deadlock?

- circular wait
- mutual exclusion
- no preemption
- hold and wait

6.) (20 points) Project

a. Why does project #2 have the system call Sys_RegDeliver? What does it do?

Defines the function to return to at the end of a signal. This registered function then returns to the kernel.

b. The project used Spawn_Program(<program name>) to implement process creation. What would you have to change in the system to support UNIX style process creation with two system calls: Fork() followed by Exec(<program name>)?

```
Split current spawn into two functions:
Fork:
      Needs to copy everything (including memory) of all process)
      Creates a new pid and sets time used to zero
Exec:
      Loads new executable into memory
      Uses same pid
      Doesn't reset elapsed trime
```

c. Given that all memory in the system is mapped into the kernel's address space, why is there a function in the kernel called Copy_From_User?

Provides offset for address since pointers passed in are relative to the process's notion of zero, not the kernel Provides memory protections in the form of bound checking so user processes can't read parts of kernel memory (or other users processes)