McGill University
Computer Science and Engineering
Midterm Exam

COMP-310 and ECSE-427
Operating Systems

Examiner and Instructor: Joseph Vybihal

Exam Date: October 18, 2007
An in class exam

Student Name:_____
Student ID: _____

Instructions

1. Closed Book Exam
2. Calculators are not permitted
3. Attempt all questions since part marks will be given
4. Answer all questions on the exam sheet. Enough space has been provided. If more space is required you can write on the back of the exam pages.
5. Ask questions if you do not understand something.

Marking

	Value	Grade
Question 1	25%	
Question 2	25%	
Question 3	25%	
Question 4	25%	
	---	---
Total	100 %	

Question 1: Semaphores

Assume there exists an emergency service that uses a computer network. The network allows two kinds of data transmissions: regular and emergency. A data transmission is defined to be a message that is sent from an arbitrary computer A to an arbitrary computer B. The bandwidth of the network is 1000 GHz. A regular transmission uses up 100 GHz, while an emergency transmission uses up 800 GHz. This means that if no emergency messages are on the network, ten regular messages can be using the network at the same time. If an emergency message is on the network then only it plus two regular messages can be on the network at the same time.

Using semaphores, figure out how the send function (from computer A) and the receive function (from computer B) must operate. (Solve this in the same style as presented on the slides in class and during the tutorials)

Question 2: Process Queuing

Devise a process queuing strategy that handles the following situation:

An operating system is needed to manage the machinery, network and application software of a gas station. This is a specialized environment because the operating system will need to be able to handle the following types of processes: background, priority, real-time and regular. A regular process would be the cash register program used by the teller. A real-time process would be the program that runs the gas pump machine. A background process would be the printer spooler. A priority process would be a warning signal from some location in the network.

Note: in your solution it is important to keep in mind how much CPU time a class of process will receive. You can measure time in Q where that represents a quantum of time. Assume that OS overhead is 0.5Q. Your queuing strategy must optimize over time. Your system must also be fair to all process classes. The strategy should minimize starvation and indefinite postponement. MAKE SURE to discuss these issues in your solution.

Note: I expect you to add additional fields to the PCB structure to facilitate your implementation as long as they are reasonable.

Question 3: Threads

Assume you have the following C commands:

```
int fork( );              // where int = = 0 if child, == child PID if parent.
int exec("filename");     // where int is the value returned by main( ) in filename
wait (PID);               // permits one thread to wait for the termination of another
```

As discussed in class, the fork( ) command creates a new child process with a LWP connected to the parents PCB. In this question, exec( ) replaces a child process with a new program having its own PCB and quantum, it is not a LWP.

Write a C-Pseudo-code program using the two commands above to do the following:

You are writing a program that when launched will display to the user a menu, will keep track of a buffered input stream S, and an un-buffered input stream P. Stream S is an input character stream that is already buffered for you. You can use the function fgetc(S) to read a character from the stream. Your job is to write that character to a file given to you from one of the menu options. These characters are written into the file until a '\0' is received. Then the file is closed and the stream S no longer needs to be watched. Stream P is not buffered and therefore the characters that are sent to you can be missed if you are busy doing something else. You must process the characters from P in a similar way as you processed them for S but instead of a file name and printer name is supplied. Keep in mind that you can fopen a file or printer name and use them similarly. To get characters from un un-buffered stream use the command freadc(P). The program's menu will have the following options: (1) set file and printer name, (2) is stream S finished, (3) is stream P finished, (4) quit. Option 2 and 3 display a message on the screen indicating YES or NO. After selecting 1, 2 or 3, the user is returned to the menu. Only in option 4 does the program terminate, but only after the stream processing is complete. The user should be able to use the menu while your program is managing the two streams.

Note: your threads can share (be able to read and write) the same memory if the reserved word SHARE is used. In other words, int x, when forked exists in both parent and child but they are local to each instance of the thread. SHARE int x, on the other hand, is a single variable that can be used as you normally use a variable in C but in this case they are the same variable for every instance of the thread.

Note: You are writing C-pseudo-code so you do not need to write the entire program in C but it should look similar to C.

Question 4: OS Architecture

At the beginning of each class a block diagram of a standard OS is presented. Please do the following:

- Redraw that block diagram
- Identify each section, and
- Define each section