CS 376	**Operating Systems**	Test 1
Name:	Fall 1999	Max Pts: 41

Important: This is an open book test. You can use any books, notes, or paper. *Do not log into the computer during the test.* Any calculations and rough work can be done on the back side of the test pages. You will loose five points for not writing your name.

1. [5 pt] Between a batch processing system and a time-sharing system, which system is likely to have a shorter response time? Explain with a reason. Could there be some conditions that could be considered an exception to the rule? If yes, name one.

2. [5 pt] A multiprogramming OS must have process and resource manager, a memory manager, a file manager, and a device manager. Which of these modules are essential for a system with no multiprogramming support? Comment on each one of the modules in not more than two sentences.

3. [5 pt] When a new process moves from the ready state to the running state, each register in the CPU must be set either to initial values or to the values the process had when it was last interrupted. Explain why the PC register is the last register loaded for the new process.

4. [10 pt] Does the following code meet the stated requirements for the solution to critical section problem. If yes, argue its correctness. If not, provide a counterexample and discuss the cause for failure. The code is meant for two processes; there is no assumption about whether it runs on single CPU or two CPUs.

```
extern bool flag[2] = { false, false };            // Shared flags

void process ( const int i )    // i is the process number (0 or 1)
{
    while ( 1 )
    {
        compute();              // Section to perform computations

        // Entry section

        flag[i] = true;
        while ( flag[(i+1) % 2] );

        critical_section();

        // Exit section

        flag[i] = false;
    }
}
```

5. [10 pt] Two processes, p_1 and p_2, have been designed so that p_2 prints a byte stream produced by p_1. Write a skeleton code (like the ones in notes) executed by p_1 and p_2 to illustrate how they synchronize with one another using

   (a) semaphores

   (b) monitor

6. [6 pt] Give an example of a concurrent application in which a sender process could use an asynchronous rather than a synchronous `send` operation. Provide another scenario in which the sender should use a synchronous `send` operation in order for the application to operate properly.