Discrete Structures	Homework 1
CS2800	Due Monday, July 17 (before lecture)

Note: As is often the case when working with systems, there is ambiguity in the specification of some of these questions. You should make (reasonable) clarifying assumptions and design choices, and write them down.

1. OS Interfaces

   (a) You have just installed a new webcam for your laptop. In order to use the webcam, you must install both a device driver and a chat application. Describe the purposes of and differences between these two pieces of software.

   (b) While installing the driver, the operating system asks you for administrative privileges, but it does not ask when installing the application. Why might this be the case?

   (c) What are the risks to the system if the chat application is faulty? What are the risks if the driver is faulty? Why are these risks different?

   (d) After installing everything and running the chat application, you click a “record” button, a red light turns on, and you see yourself on the screen.

   Describe the sequence of actions taken by the application, operating system, device drivers, and device controllers, starting with the click. Be sure to describe all of the traps and context switches that occur, and make a note of the privilege mode at each point of execution.

2. Processes

   Suppose we wished to write an operating system that doesn’t support time-sharing: it runs each process without interruption until it terminates.

   (a) What, if anything, should be recorded in each process’s PCB?

   (b) Under what conditions, if any, would there be a context switch?

   (c) What would the state-transition diagram for a process look like?

   (d) How do your answers to (a)-(c) change if the system uses n processors to run n processes?

   (e) Suppose we had a program with the following behavior:

   - for each of 10 small files,
     - read the file (from disk)
     - execute 10000 CPU instructions

   How much time would non-preemptive time-sharing save if we were to run 3 instances of this application? Make reasonable assumptions about disk latency and processor speed (and write them down).

3. Threads

   Consider a process containing multiple threads. For each of the following components, describe whether the components are shared between the threads or whether each thread has its own. For each shared component, explain what would go wrong if the component was not shared; for each unshared component explain what would go wrong if they were shared.

   (a) code

   (b) data

   (c) files

   (d) registers

   (e) program counter

   (f) stack

   (g) PCB

4. Threads vs. Processes

   The file `parcount.py` contains a partial implementation of a program that runs N jobs, each of which does some I/O and some processing. It uses three strategies to run the jobs: `sequential` runs them all sequentially. The `threaded` divides the jobs into n threads, each of which runs N/n of the jobs, and the `multiproc` implementation spreads the jobs across n processes.

   (a) Develop reasonable hypotheses about how the total run time varies with n for each of the three implementations, both for I/O bound tasks and for CPU bound tasks. Sketch plots of your hypotheses and explain the interesting features. This should be done *before* completing the implementation.

   (b) Complete the implementation. Incomplete portions are marked with "`TODO`"

   (c) Plot the time taken to run each of the three implementations with varying numbers of threads/processes. Explain the results you see, especially if they are different from your hypotheses.

   (d) The `do_step` function is I/O bound: most of the time is spent in the call to `time.sleep`. Modify `do_step` so that it does a large amount of computation and no I/O; measure the running time and explain your results.