1. Interrupts (20pts)

   Suppose that you are developing a new computer and OS with significant resource constraints. The
   hardware is only allowed to support either clock interrupts or I/O interrupts, but not both.
   (a) Discuss how your system would support multiprogramming without clock interrupts, ensuring that
       all processes can perform I/O, that prevents any process from monopolizing the CPU or going into an
       infinite loop, and that supports time sharing by switching quickly amongst processes.
   (b) Do the same but for a system without I/O interrupts.
   (c) Argue which you would choose with the goal of providing the best user performance for a low
       price. What makes your solution more economical than the alternative? Under what solutions would
       your OS perform well? Under what conditions would it respond poorly? Note that your design is for
       a uniprocessor system and that user processes and the OS must take turns with the one CPU.

   There are many possible answers. Be sure that your solution is consistent with multiprogrammed
   uniprocessor systems (what we’ve talked about so far in class) and interrupts. Do not provide an-
   swers where the OS acts as a monitor watching over actions and intervening (e.g., an OS noticing a
   PC instruction run 1000 times and switching processes is a wrong answer). You are welcome to use
   diagrams for clarification if necessary. A good answer will be a page or less single-spaced.

2. Textbook Questions (20 pts)

   Answer questions 2.8, 2.19, 3.6, 3.7, 4.2 in Silberschatz 8/e. You only need a few sentences to answer
   each question.

   2.8: What is the main advantage of the layered approach to system design? What are the disadvan-
   tages of using the layered approach?

   2.19: Why is the separation of mechanism and policy desirable?

   3.6: Describe the differences among short-term, medium-term, and long-term scheduling.

   3.7: Describe the actions taken by a kernel to context-switch between processes.

   4.2: What are two differences between user-level threads and kernel-level threads? Under what cir-
   cumstances is one type better than the other?

3. Process Analysis (20 pts)

   Design a methodology using the Linux proc filesystem to extract as much information as you can
   about a running process, without actually having the source code of its program. The experiment may
   begin something like this:

   % sleep 600 &
   [1] 1363
   . . .
You should `cd` to `/proc/<processID>` (in this case, `/proc/1363`, since that was the number returned when the process went into the background - you can also find it by typing `ps`) and `ls` through the directory to look at names of files. You can `cat` those files to see what information they have that are relevant. Explain which piece of information each step of the methodology is designed to supply, and how that related to the program, process, and operating system. Look at `proc(5)` for more information about the individual files.

**Note:** Like all assignments in this class you are prohibited from copying any content from the Internet or sharing ideas, code, configuration, text or anything else or getting help from anyone in or outside of the class, except where noted. Consulting online sources is acceptable, but under no circumstances should anything be copied. Failure to abide by this requirement will result in sanctions ranging from zero on the assignment to dismissal from the class.