1. (a) Give a definition of the term process. Explain the difference between procedure, program, process, task, and job.
(b) On all current computers, at least part of the interrupt handlers are written in assembly language. Why?
(c) For UNIX, give some examples of process manipulation functions.

2. The Running, Blocked and Ready states of a process imply that the process is always in main memory. However, high-level scheduling will include the ability to swap the process’s main memory to disk, which is defined as the Suspend state for a process, and reload the process into main memory when it is activated.
(a) Explain from which state or states a process can become suspended and why?
(b) Explain the problem in deciding which state a suspended process should go to when that process is activated? What is the UNIX System V solution?

3. (a) A blocked process still consumes system resources. Is it possible for a program to either accidentally or otherwise continually create processes that immediately block on some event that won’t occur and so bring the system down?
(b) If a parent process dies, what should happen to the child processes? What happens to a parent process when a child dies?

4. Five batch jobs A through E, arrive at a computer center at almost the same time. They have estimated running times of 10, 6, 2, 4, and 8 minutes. Their (externally determined) priorities are 3, 5, 2, 1 and 4, respectively, with 5 being the highest priority. For each of the following scheduling algorithms, determine the mean process turnaround time. Ignore process switching overhead
(a) Round Robin (RR).
(b) Priority Scheduling.
(c) First come, first served (FCFS).
(d) Shortest job first (SJF).

For (a), assume that the system is multi-programmed, and that each job gets its fair share of the CPU. For (b) through (d) assume that only one job at a time runs, until it finishes. All jobs are completely CPU bound.
5. (a) Consider the implementation of all the short-term scheduling algorithms discussed in the lecture.
   (i) Which algorithms are effectively impossible to implement? Carefully explain why?
   (ii) Which algorithms require a timer interrupt for the CPU?
   (iii) Hence explain why RR is the only real scheduling algorithm that can be used in practice.
(b) Can a single processor system have no processes in the ready queue? Explain what can happen in such a situation.

6. (a) Explain why two-level scheduling is commonly used.
(b) Define the difference between preemptive and nonpreemptive scheduling. State why strict nonpreemptive scheduling is unlikely to be used in a computer center.
(c) A CPU scheduling algorithm determines an order for the execution of its scheduled processes. Given $n$ processes to be scheduled on one processor, how many possible different schedules are there?

7. Consider the following measured CPU service times for a process:
   1 1 2 4 5 5 3 2 2 6 7 7 7
   Use the aging algorithm with $a = 0.5$ to predict the next CPU service time and comment on the accuracy of this method. Assume the initial predicted service time is 1.