Operating Systems A1:

# Data structures:

* Struct process
  + Int bound
    - CPU bound must want more time than normally allowed in CPU before fast I/O spending most time in CPU.  I/O bound should stay in CPU much less time than normal CPU time and spend most time waiting for I/O.  Even should spend about the same amount of time in CPU as doing I/O.
    - 0 = cpu bound
    - 1 = between 1 and 2 (75% cpu, 25% I/O)
    - 2 = even cpu and I/O
    - 3 = between 2 and 4 (25% cpu, 75% I/O)
    - 4 = I/O bound
    - A. Time in CPU needed before/between I/O (set once)
    - B. Time I/O takes (set once)
    - C. Total time in machine not counting time in wait queue (set at startup counts down). How much time process spends doing something before it exits. Sum of total time in CPU and total time in I/O.
    - D. Time in CPU currently. Set to 0 when moved into CPU, when reaches A or quantum is up moves out of CPU.
    - E. Time left waiting for current I/O
    - F. Time process has been waiting in ready queue
    - G. Total time in CPU
    - H. Total time in I/O
    - I. Total time in ready queue
    - J. Smallest time in ready queue
    - K. Longest time in ready queue
  + Int startingPriority
    - Values range between 0 and 20, with 20 being the highest priority
  + Int currentPriority
    - Values range between 0 and 20, with 20 being the highest priority
  + Char\* name
    - Name of the process
  + Int location
    - 0 = cpu
    - 1 = I/O
    - 2 = ready
* Struct process[] priority queue
  + Holds all 100 processes
* Struct process[] IOqueue
  + A list of processes waiting for I/O
* Struct process cpu
  + Holds the process currently in the cpu
* processParams
  + parameters common to all processes
    - Max time in CPU before being bumped to ready queue if no I/O, quantum.
    - Max wait time user processes in ready queue (try different times, how small can your scheduling algorithm handle?)
* struct process[] initQueue;

For the processes, the base CPU times are as follows:

1. a
2. a
3. a
4. a
5. a