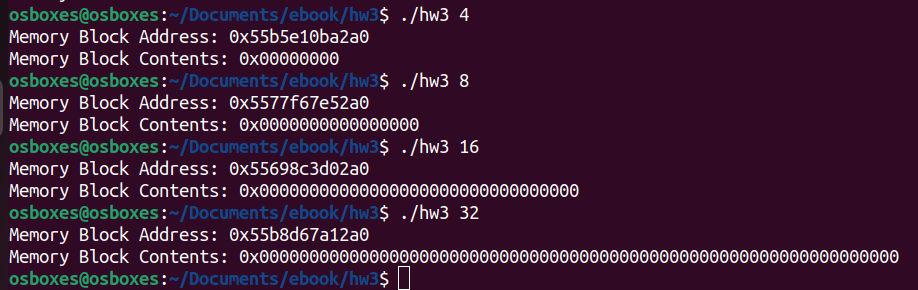
# Lists and Structures

Write a "C" structure definition for the process information.

**A struct.c file has been included in the submission zip.**

# C Pointers and References



# Scheduling Policies

*Over the years Linux has used a number of different processing scheduling algorithms. Two current ones are the CFS and BFS schedulers. For each of these algorithms describe, in a paragraph, the main features of the algorithm (are they priority based, time sliced, etc.)*

The **Completely Fair Scheduler (CFS)** is the scheduler generally used as the default scheduler for processes under the Linux kernel. Tasks are organized into a priority tree, and the leftmost task in the tree is the one that runs on the CPU. The scheduler continually moves nodes in the priority tree to the right as they are executed. This is what makes this scheduler “completely fair,” as eventually, all processes will be guaranteed allotted room on the processor.

The **Brain Fuck Scheduler (BFS)** is an alternative scheduler designed by Con Kolivas after a dispute with Ingo Molnar (the creator of CFS). Rather than a priority tree, BFS focuses on a simpler algorithm using a doubly-linked list acting as a process queue. In BFS, whenever a context switch occurs, the CPU will traverse the process queue until it finds a process with an adequately high priority, as defined by a “virtual deadline formula.” BFS often beats CFS in turnaround time, but because BFS traverses a linear queue rather than a tree as CFS does, its worst-case time complexity is O(n) while CFS’s is O(logn), and BFS experiences more “hiccups” in performance due to the queue traversal overhead.

Both BFS and CFS are *preemptive,* with currently running processes being replaced by new processes based on internal priority values handled by their respective schedulers.