CS423 - MP2 - Group 1
Seylom Ayivi, Son Nguyen, Casey Piper

Implementation description:
Our implementation consists of

- Initialization
  - Created a linked list that stored augmented process control blocks
  - Created /proc/mp2 proc entry directory and /proc/mp2/status proc entry file, giving the callback read and write functions for the status file
  - Created kernel dispatcher thread to handle scheduling of task based on priorities
- Callbacks
  - /proc/mp2/status file write: copy buffer from user space and parse the buffer in order to retrieve the command (R,Y,D - for register, yield and deregister) as well as their respective parameters.
  - /proc/mp2/status file read: loop through linked list and copy process information (such as pid,period and computation) to the buffer.
  - Register: The registration process takes process parameters, compute the projected CPU utilization and decided whether to add the process to the list of augmented process or not.
  - Yield:This method provide the system with a mean to perform context switches whether voluntarily requested by the executing application or by the timer interrupt of a given process..
  - Deregister: The process is removed from the list of augmented process control blocks, its timer is killed and the utilization of the processor is updated.
- Cleanup
  - Loop through, remove, and deallocate process control blocks as well as deleting their respective timers stop the dispatcher thread and remove proc entry

Design Decisions:

- Locking
  - Spinlock are used when performing looping operations (whether read or write)
- Computation of the admission control:
  - We first keep a variable for the processor utilization
  - For each newly process registration request we compute the new utilization and accept or reject the process based on the new projected utilization.
  - When a process is deregistered, the utilization is decreased by the amount of utilization the process was using.

Tests:

- The test are performed by creating a test applications registering themselves with a periodic job and voluntarily performing context switch during their job execution. The stack trace of the test are a good indication of what happened during the execution of the program, yields, timer interrupt, process registration and unregistration.