

Operating Systems

CMSC 422 and MSCS 515

*i*Project Three

Starting with and including all functionality of *i*Project two, add new features as specified below.

OS Functional Requirements

- Allow the user to load multiple (at least three) programs into memory at once. [required]
- Add a shell command to let the user set the RR quantum measured in clock ticks. [required]
 - Default to 6 clock ticks. [required]
- Provide “reset” functionality to allow the user to “start over” without reloading. [required]
 - Clear the memory, PCBs, Ready queue, etc. [required]
- Display the Ready queue and its PCBs’ contents in real time. [required]
- Add a shell command to display the process ids of all active processes. [required]
- Add a shell command to kill an active process given its PID. [optional]

Source Code Requirements

- Store multiple programs in memory, each in their own area allocated by the OS. [required]
 - Add base and limit registers... [required]
 - to your memory access hardware simulation. [required]
 - to your PCB object. [required]
 - Enforce the memory boundaries all the time. [required]
- Create a Ready queue to hold the PCBs. [required]
- Instantiate a PCB for each loaded program and place it in the Ready queue. [required]
- Synchronize the CPU execution cycles with clock ticks. [required]
- Develop a CPU scheduler module in the OS. [required]
 - Use Round Robin (RR) scheduling with the user-specified quantum. [required]
- Make the OS control the CPU with the CPU scheduler as it executes. [required]
 - Log all scheduling events. [required]
- Context switch with software interrupts. Be sure to update... [required]
 - the mode bit as appropriate. [required]
 - the PCBs. [required]
 - the Ready queue. [required]
- Detect and gracefully handle errors in the user machine language code. [optional]
 - Invalid op codes [optional]
 - Missing operands [optional]
 - Out of bounds access (attempts) [optional]
 - other errors as we dream up silly or malicious testing techniques [silly]

Hints

- Refactor everything according to the architecture we worked out in class.

Style, Art, and Science Requirements

- Your code must separate structure from presentation, be professionally formatted, and use and demonstrate best practices
- Remember the value of comments and expressive identifier names and how much their presence and quality affect my grading.