ASSIGNMENT 3

What's new in SOS2?

- A NAÏVE memory manager *(memman.c)*
- A timer interrupt handler *(timer.c)*
- Processes with states *(kernel_only.h)*
- One more system call – `sleep` *(kernelservices.c)*
- A scheduler (this assignment) *(scheduler.c)*

Background

The background material for this assignment is in Chapter 4 of "Understanding a Simple Operating System." **You must read it (and the previous chapters) before attempting this assignment.**

Assignment objective

The file *scheduler.c* has three incomplete functions with **`TODO`** blocks. Finish these blocks to complete the assignment.

- **`PCB *add_to_processq(PCB *p)`** : inserts a PCB into the circular doubly linked list of existing processes; returns a pointer to the added PCB
- **`PCB *remove_from_processq(PCB *p)`** : removes a PCB from the existing doubly linked list of processes; returns a pointer to the next PCB in the list, if any, otherwise `NULL`
- **`void schedule_something()`** : the SOS2 scheduler

Note: Ignore the TODOs outside these three functions; they are for me.

User programs

(/userprogs)

Two new user programs are included in this assignment.

1. `p1.out` stored in sector 1200 and 604 bytes long; see *userprogs/p1.c*
2. `p2.out` stored in sector 1300 and 604 bytes long; see *userprogs/p2.c*

Setup

Download the *ASG3.zip* file from the assignment page. Extract the folder. In SOS2, use `./create` to create the `SOS.dsk` file (just like you did for SOS0 and SOS1). Create a new machine called SOS2 in VirtualBox, and use the *SOS.vmdk* file for the hard drive. Once SOS boots, you can run one of the three available programs. For example, to run *p1.c* and *p2.c* concurrently, type

```
%run 1200 2
%run 1300 2
```

However, nothing will happen for now since the processes do not get added to the queue (`add_to_processq` is empty) and the scheduler (the `schedule_something` function) always dispatches the console. But once you finish the assignment, you should see both programs executing concurrently (you will see output with mixed `A`s and `B`s).

Submission

Submit in Canvas the modified *scheduler.c* file, and a README file containing any information you feel the GTA should know before grading your program. Comment your program well (include your name).

Grading

The assignment is worth **100 points** – 50 points for `schedule_something`, 20 points each for `add_to_processq` and `remove_from_processq`, and 10 miscellaneous points for code clarity, commenting, etc. The assignment will be graded within the SOS development environment. It is your responsibility to ensure that the submitted code runs in that environment (irrespective of where you wrote or tested your code). Failure to do so will result in full penalty.

The late policy is available in the course syllabus. You must work alone on this assignment.