1 Final Project

Over the next several weeks you will work to implement many of the core features of a real operating system. Although this is all in the same folder you should be able to work on each weeks lab without needing to have the previous ones finished.

2 Scheduling

This first week is going to be desiging several scheduling algorithms and compairing their response times and other features. First, **please** take a look around the provided code in the project folder. Nearly all of this has been finshed for you; however, it is good to look through it and understand how the different parts work.

2.1 Sourcing the enviornment

We first need to setup the enviornment we are working in. Since this is a fairly large project and setting up that enviornment takes several steps we can group all of that into one operation. Note that this operation is only good in the terminal that it is run in. Each time you open a new terminal to work on the project you will have to run this following line in the root directory of the project.

```
$ source source.me
```

It will print out a message about "is a directory" which can be safely ignored.

2.2 Compiling the libraries

This project consists of two libraries each of which need to be compiled and installed.

2.2.1 Linked List

The linked list library provides a full featured linked list with itterators. This libary is used extensively throughout the entire project and should be the first library installed. For documentation on the linked list libary look at `include/llist/isu_llist.h` after installing.

To compile and install the linked list library from the root directory of the project run

```
$ cd lib/llist
$ make
$ make install
```

2.2.2 Module

The module library provides dynamic loading of modules into the system at run time. Modules must support and `install()` function which is called during the install process and returns a void pointer which is the access to the module during runtime. For documentation on modules look at `include/module/isu_module.h` after installing.

The linked list library is a dependency of the module library so make and install that one first. The from the root directory of the project run the commands

```
$ cd lib/module
$ make
$ make install
```

2.3 Compiling the algorithm

One of the scheduling algorithms has already been finished for you as an example. `cd` into the `scheduler\fcfs\` directory and run `make`. next `cd` back to the `project` directory and run `make`. To test it, run the following command:

```
$ ./sched_test scheduler/fcfs/sched-fcfs.mod
```

When this finishes it will create two files: one called `fcfs.log` which show information about each task along with the overall averages, and one called `fcfs.json` which you can copy the contents of into http://www.wavedrom.com/editor.html to see a Gantt chart of the scheduler.

2.4 Tasks for this lab

Your tasks for this lab are to implement the Shortest Remaining Task Next, Round Robin, and Priority Round Robin schedulers. You can test multiple schedulers using the `sched_test` program by suplying each as a command line argument. The expected outputs from each scheduler are given in the `answers` directory to compair with. *note, I can't garuntee that the answers are correct, I finished them up at 5AM and have not verfied them.* Comment in your lab report the outputs of each algorithm and how they compare with each other. Is it what you would expect? Which do you consider the best?

2.5 Going futher

After you have finished the scheduling algorithm you should now play around with the numbers to see how it effects the response times. In the `init_test_framework` function in `sched_test.c` you can chage the rate which new processes enter the system by changing the modulo value on the `start_time +=` line, and the length of each process by chaging the `length =` line. Take care to make sure the length is never `< 1` as this will cause problems with the test program. Also try changing the quantum of round robin algorithm to see how that effects the system. Comment on what you do and your observations on the effects in your lab report.

3 Extra Credit

For extra credit you can implement your own scheduling algorithm. Feel free to get as creative as you would like on this and folow the given examples to get started. One option is priority round robin with different quantum for each priority level.

4 License