Khristian Brooks
Operating Systems
Homework 1

Summary

Language Choice:

My choice of language was python. Since, this project will be iterated on in future homeworks, I wanted to use a language with large support alongside of readable and maintainable code. Python seemed the best solution for this. Because it has a large community there is a lot of useful packages for such as the sockets and queue packages that made this assignment a little easier to implement. If I had used a language like C, I would have to construct my own queue. Alongside that, unlike other programming languages, Python emphasizes on code readability and allows one to use simple English keywords.

IPC method:

The IPC method that I chose was sockets. I prefer this solution over it's alternatives because of it's simple implementation using python and it's lack of side effects unlike it's shared memory contemporaries.

Sockets vs. Shared Memory:

Unlike shared memory solutions, such as shared RAM or shared files, passing messages via sockets is straightforward. When dealing with the shared RAM solution, you have to be cautious not to create data race conditions that come with writing into RAM with two writers - in our case the CPU and Scheduler will both have to write into the memory space which can cause them to overwrite the data. In the Shared File solution, your attention will have to be on increasing efficiency. Since each read from a file requires a read from the Hard Disk Drive (HDD).

Sockets vs. Other Memory Passing Alternatives:

The memory passing solution seemed to be a better solution then shared memory. However, I personally prefer the use of sockets. Pipes would have been a nice solution because the assignment only needed local communication for the files, however, there is more documentation and support for using sockets in Python. Another viable solution for passing messages was using a remote procedure call, however, I wanted a concise solution and creating a remote procedure call would take a lot of time and energy.

Khristian Brooks
Operating Systems
Homework 1

Appendix

CPU Emulator Output:

```
Executing:              ls,1000,ready,1,0,360,360
Sending back to Scheduler: ls,1000,ready,1,0,360,160
Executing:              wc,1001,ready,3,0,120,120
Sending back to Scheduler: wc,1001,ready,3,0,120,-80
Executing:              cat,1002,ready,2,0,430,430
Sending back to Scheduler: cat,1002,ready,2,0,430,230
Executing:              sed,1003,ready,1,0,411,411
Sending back to Scheduler: sed,1003,ready,1,0,411,211
Executing:              ps,1004,ready,1,0,360,360
Sending back to Scheduler: ps,1004,ready,1,0,360,160
Executing:              grep,1005,ready,2,0,430,430
Sending back to Scheduler: grep,1005,ready,2,0,430,230
Executing:              python,1006,ready,4,0,560,560
Sending back to Scheduler: python,1006,ready,4,0,560,360
Executing:              ls,1000,ready,1,0,360,160
Sending back to Scheduler: ls,1000,ready,1,0,360,-40
Executing:              cat,1002,ready,2,0,430,230
Sending back to Scheduler: cat,1002,ready,2,0,430,30
Executing:              sed,1003,ready,1,0,411,211
Sending back to Scheduler: sed,1003,ready,1,0,411,11
Executing:              ps,1004,ready,1,0,360,160
Sending back to Scheduler: ps,1004,ready,1,0,360,-40
Executing:              grep,1005,ready,2,0,430,230
Sending back to Scheduler: grep,1005,ready,2,0,430,30
Executing:              python,1006,ready,4,0,560,360
Sending back to Scheduler: python,1006,ready,4,0,560,160
Executing:              cat,1002,ready,2,0,430,30
Sending back to Scheduler: cat,1002,ready,2,0,430,-170
Executing:              sed,1003,ready,1,0,411,11
Sending back to Scheduler: sed,1003,ready,1,0,411,-189
Executing:              grep,1005,ready,2,0,430,30
Sending back to Scheduler: grep,1005,ready,2,0,430,-170
Executing:              python,1006,ready,4,0,560,160
Sending back to Scheduler: python,1006,ready,4,0,560,-40
```