Sean Hinchee

Section G

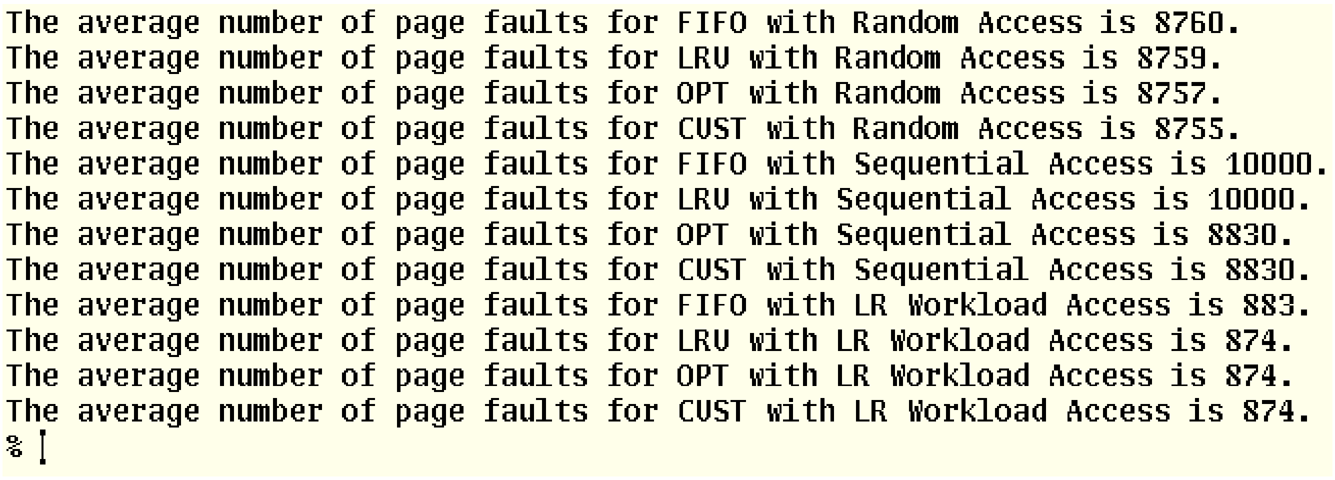
Lab 7

4/5/18

Memory Management

In this lab I dealt with memory management algorithms and worked to write an efficient one of my own. I decided after some extensive testing that I was not sure if there was a better algorithm than my LRU implementation that I could write for my CUST implementation on the LR sequence. My LR implementation finds the oldest element and returns that ID. The description for the LR sequence states that it has a 90% chance to select from the five youngest pages. A simple way to get around this is to select the oldest page in the available pool to remove, as this will not enter the five most recently used pages in the near future.

Overall I felt I learned a great deal and that my numbers reflected the trends I expected in the lab description. My numbers are displayed below:



```
The average number of page faults for FIFO with Random Access is 8760.
The average number of page faults for LRU with Random Access is 8759.
The average number of page faults for OPT with Random Access is 8757.
The average number of page faults for CUST with Random Access is 8755.
The average number of page faults for FIFO with Sequential Access is 10000.
The average number of page faults for LRU with Sequential Access is 10000.
The average number of page faults for OPT with Sequential Access is 8830.
The average number of page faults for CUST with Sequential Access is 8830.
The average number of page faults for FIFO with LR Workload Access is 883.
The average number of page faults for LRU with LR Workload Access is 874.
The average number of page faults for OPT with LR Workload Access is 874.
The average number of page faults for CUST with LR Workload Access is 874.
%
```

I was unable to beat my LRU algorithm with my CUST algorithm, but I believe that this is because my LRU algorithm was written too precisely, but I am not sure.