CS 3413

Assignment 8

Due Date: November 25th, 2019 at 9:30 am

ASSIGNMENT IS TO BE COMPLETED INDIVIDUALLY BY ALL STUDENTS!

Your solution is to be written in C and submitted via D2L.

This question consists of writing a program that will examine how the number of frames allocated to a process can be increased (or decreased) as the process performs its work.

The program will take as input (from stdin) a list of pages that are being accessed by the process as it does it's work (sample input file provided). As each page is accessed, you need to assign the page a frame to be stored. The frame is chosen according to the 3 different page replacement algorithms we discussed in class (FIFO, LRU and Optimal). Upon starting, the process is given *n* frames to be used. So, the command line for your program is:

```
./a.out -{f|l|o} –n 10
```

where **f** is for FIFO, **l** is for LRU and **o** is for optimal. **10** indicates that the process will start with 10 frames. Note: Since the input is page numbers, then you DO NOT have to do logical to physical address conversions and maintain page directory/tables!

While the process starts with *n* frames allocated, it does not stay this way. When a page fault occurs you need to determine if your process is allocated an additional frame by the kernel … or if the kernel takes away a frame from your process. For this, you are to add two more command line options:

```
./a.out -{f|l|o} –n 10 –low 20 –high 5
```

the reciprocal of the low and high numbers give the lower and upper bound respectively for the page fault frequency. In this case, the lower bound is 0.05 and the upper bound is 0.2. If when a page fault occurs the page fault frequency is above the upper bound then the process gets an additional frame from the kernel (ie. *n* increases by 1 and the page is loaded in the new frame – no page out occurs). However, if the page fault occurs and the page fault frequency is below the lower bound then the process loses a frame to the kernel. Therefore, 2 pages are removed from frames – the first page is removed and that frame is given back to the kernel and a second page is removed to load the page requested (ie. *n* decreases by 1). If the page fault frequency is between the bounds, then we do just a normal page in/out and *n* does not change.

Your program will print the following output.

When a page fault results in the process increasing or decreasing the number of frames that it has from the kernel (where x is the line number in the input file and y is the value of n):

```
Input Line x: n = y
```

When your program finishes processing the input, it prints the following summary of execution (where x, f and n are the actual values):

```
Total number of Page Faults: x
Final Page Fault Frequency: f
Final number of frames allocated to process: n
```

Hint: If your program runs correctly, f should be equal to total number of page faults divided by the number of input lines read ☺