Problem Set 3

Due: March 15 at 11:59 PM

The goal of this problem set is to help you understand how the various page replacement algorithms work is detail. you are to implement in C the following functions:

```
int Random( char *fileName);
int FIFO( char *fileName);
int LRU( char *fileName);
int ApproximateLRU( char *fileName);
int Optimal( char *fileName);
```

Each of them tries to open the file with the given file name, If the file cannot be opened, then the function returns -1. Otherwise, it simulates the appropriate paging method on the file and returns the number of page faults that occur. The file is in the following format:

```
#3
0
1
2
3
0
1
4
0
1
2
3
4
```

The number on the first line is the number of physical pages that you are to assume (3 in this case). Each subsequent line of the file is an integer that "names" a virtual page. In the terminology of the text, this file is the "reference string". Assume that, at startup, none of the virtual pages are present in the memory.

All the methods are standard as described in the text and in the class. For *ApproximateLRU*, use the aging method of the text and set the timer interval to 100; that is, update the counters every 100 page references.

Test your functions any way you want. However, you must run the functions on the files data1, data2 ... in the directory */u/cs318/3* on phoenix. Also, present in the directory is a sample "driver" routine driver.c which will be used to test your functions. You can use the files in that directory as a starting point.

Submit all the code, makefile and a README file on phoenix. The README file should contain the results of running your code and your conclusions. Also, *coding style counts*. So, please be neat, use mnemonic variable names and comment tricky code.