Project - 2
CSE 421/521 – Operating Systems
Due: November 19 @11:59pm, 2012

1. Preparation

Before beginning your work, please read the following carefully:

- Chapter 9 from Silberschatz
- Lecture slides on Virtual Memory and Page Replacement Algorithms

2. Programming Task: Implement a Simple Virtual Memory Manager

The objective of this project is to implement a simple virtual memory manager "**virtualmem**" in C/C++ on a UNIX-based platform.

SYNOPSIS: virtualmem [**−h**] [**-f** *available-frames*] [**−r** *replacement-policy*] [**−i** *input_file*]

DESCRIPTION: virtualmem is a simple virtual memory manager. It takes a sequence of page references as an input, as well as the number of available frames. It performs the placement of these pages to the available frames using the page replacement policy specified by the user.

2.1 OPTIONS:

−h **:** Print a usage summary with all options and exit.

−f *available-frames* **:** Set the number of available frames. By default it should be 5.

−r *replacement-policy* **:** Set the page replacement policy. It can be either
FIFO (First-in-first-out)
LFU (Least-frequently-used)
LRU-STACK (Least-recently-used stack implementation)
LRU-CLOCK ((Least-recently-used clock implementation – second chance alg.).
LRU-REF8 (Least-recently-used Reference-bit Implementation, using 8 reference bits)
The default will be **FIFO**.

−i *input file* **:** Read the page reference sequence from a specified file. If not given, the sequence should be read from STDIN (ended with ENTER).

2.2 OUTPUT:

You should also implement the **Optimal** page replacement algorithm, and compare the performance of the replacement-policy chosen by the user (above) to the Optimal algorithm, in terms of number of page replacements for the given string.

i.e:

$ virtualmem –f 10 –r LFU –i myinputfile

# of page replacements with LFU : 118
# of page replacements with Optimal : 85
% page replacement penalty using LFU : 38.8%

Total time to run LFU algorithm : 1214 msec
Total time to run Optimal algorithm : 1348 msec
LFU is 9.9% faster than Optimal algorithm.

3. What to Submit?

- There will be no teams for this project, and it will be implemented individually. You need to prepare a **tar package** containing all source files of the project, and call it <yourlastname>.tar. This package should also include a Makefile and README file. The whole package should compile when the tester simply types make in the source code directory. Your README file should contain details and options on how to compile and run the server, if there are any.

  This package should be **emailed** to Prof. Tevfik Kosar (tkosar@buffalo.edu) and cc'ed to the TAs Ying Yang (yyang25@buffalo.edu) and Weida Zhong (weidazho@buffalo.edu) by **November 19st @11:59pm.**

- You also need to write a report explaining the design of your virtual memory management system. The report should especially focus on the implementation of the page replacement algorithms (i.e. the data structures you have used etc). The report should not exceed 3 pages. You should append your README file (Appendix 1) and your source code (Appendix 2) to this report.

  A hardcopy of this report should be submitted to Prof. Kosar at the beginning of the class on **November 20th, @9:30am.**