CS570 Summer 2017 Assignment 2

This page last modified 6 June, 2017

You shall build a simulation of the page replacement function of the page fault interrupt handler routine for a paged memory management system. Additionally, you shall analyze three page replacement algorithms using this simulation.

1. You shall create a program which implements the following Page Replacement Algorithms:
   - OPT (Optimal)
   - Second Chance
   - Clock

   This program shall simulate a process/thread accessing its pages during execution. The sequence of pages the page/thread accesses will be contained in a file, named “pages.txt” which you shall create. This file shall be located in the same directory as the executable and shall contain a series of positive integers representing the pages (by it’s page number) which the process/thread is accessing as it runs. The limit of number of pages any process/thread may have is 99. The pages.txt file will contain a single line of text where each number is separated by a space character.

2. The program shall perform the following:
   - Upon startup, prompt the user for the number of frames in main memory
   - Read the sequence of pages accessed from the file "pages.txt"
   - Run each of the algorithms (OPT, **Second Chance**, and Clock) on the same input string from the step above
   - Printout the number of faults each algorithm generates to the terminal.

3. Perform an analysis of the page replacement algorithms:
   - Use the simulation and page replacement algorithms developed in steps 1 and 2 above.
   - Develop at least two different input data sets for analyzing the algorithms. The sequences of pages in these two sets should be designed such that they demonstrate the differences in the **OPT vs Second Chance** vs Clock algorithms.
   - Run the input sets through the simulation, collect the data. You should use at least two different memory systems (different numbers of frames) on each data set.
   - Perform an analysis of the data (include controls, variables, outcomes, and input characterizations).
   - Document your experiment and findings in your README file. -- Note, your README file shall contain the usual requirements for a README file (see READMEformat.pdf file posted on Blackboard) and it shall also include a summary of your findings from your analysis.

I will test your program by compiling it and executing it on edoras. Your program shall be written such that it compiles and executes cleanly when using the gcc/g++ compiler on edoras. Note - you must use a Makefile and you must use functions/methods (not just one or three) to implement the code for this project.

You shall create a sub-directory named "a2" in your home directory. In it, you shall place all of your project files, including your Makefile and a README file. Your source files SHALL CONTAIN sufficient comments for making the source easy to read. Points will be taken off for poorly (or non) commented source. Name the executable "a2".

- Create ~/a2 by hand.
- Create all necessary project files. Put them into ~/a2.
- The Makefile shall create an executable named "a2" in this same directory (~/a2).
- The system call "system()" will NOT be allowed
- You are working individually or in teams of two this assignment.
- You may use the gcc, or g++ compiler on this assignment.

The assignment is due 1800 on Monday, 19 June 2017

TURNING IN YOUR WORK:
Follow the turn-in procedures posted in the course's Blackboard (as stated in the Assignment_Turnin.pdf file, i.e. one of you upload a tarball with all project files to the Assignment on Blackboard).