Programming Assignment 2: Page Replacement Algorithm
CSC 4103: Operating Systems, Spring 2019

Due: April 4th, Thursday (by 11:59 PM)
Total Points: 10

Objective
To implement a page replacement algorithm used by operating systems.

Background
If the total memory demand exceeds the physical memory capacity, the operating system needs to replace pages from memory based on the locality principle. The CLOCK (i.e., Second-Chance) algorithm is an approximation of the well-known Least Recently Used (LRU) algorithm, which evicts the least recently accessed pages. The CLOCK algorithm uses a reference bit associated with each page to efficiently estimate the recency of page references and identify the victim pages for eviction.

Programming Task
Write a program that implements the CLOCK page replacement algorithm. Use the page reference sequence in the input file to drive your program. Report the number of page faults and latencies incurred by the page faults (see details below). The number of available page frames can be set as specified by an input parameter of your program. Assume that demand paging is used and the page frames are initially free. More details about the CLOCK algorithm can be found in the lecture notes and the textbook.

Programming Language
C/C++ or Java.

Detailed Requirements

(1) The program should accept two input parameters to specify:
   a. The memory size (the number of available page frames)
   b. The input page reference file that contains the page references.
      Example: `$ ./replace 20 pageref.txt`

   Page Reference File Format: Each line has two fields. The first field is either "R" or "W", meaning a read or a write to the page, respectively. The second field is the page number. Thus, each line of the input file specifies a page reference. Our course website provides two sample input page reference files:
   http://www.csc.lsu.edu/~fchen/class/csc4103-sp19/assignments/pageref.txt
   http://www.csc.lsu.edu/~fchen/class/csc4103-sp19/assignments/pageref-simple.txt

(2) The program should generate the following output information:
   a. The total number of page references.
   b. The total number of page faults.
   c. The total number of time units for swapping in pages.
   d. The total number of time units for swapping out pages.

(3) Page fault costs: The program needs to calculate the "cost" of page faults in terms of time units. In particular, upon a page hit (the page is found in memory), the reference cost is 0; upon a page miss (the page is not found in memory), it takes 5 time units to load (swap in) the page. When evicting a victim page, if the victim page is modified (i.e., the page has been written in memory), it takes 10 time units to write (swap out) the modified victim page to the swap space. We assume the pages are all initially in the swap space and not in memory.

(4) Your submission should include an README file to clearly explain how to compile and run your code, such as the command, parameters, and expected input and output, and any other necessary information. In the README file, please also provide your full name, LSU ID, and email address.

Before you submit:

(1) Compile and test-run your code on the classes.csc.lsu.edu server. Your code should be written in the programming languages as specified above, and be compilable and runnable in the classes server's Linux environment. Windows code will not be accepted.
(2) Submit your work as instructed before the deadline and verify your submission as instructed below. The verify tool will display your submission date/time.
(3) Late submissions will be penalized by 10% per day late and no more than 3 days late.

Submitting Your Work

All files you submit must have a header with the following:

Name:	Your Name (Last, First)
Project:	PA-2 (Page Replacement Algorithm)
File:	filename
Instructor:	Feng Chen
Class:	cs4103-sp19
LogonID:	cs4103xx

You need to use the server "**classes.csc.lsu.edu**" to work on the assignment. You can login to your account in the server using SSH. Create a directory **prog2** (by typing **mkdir prog2)** in your home directory, and then you create your program or source code in **prog2**.

Note that do NOT include any directory in **prog2**.

Make sure that you are in the **prog2** directory while submitting your program. Submit your assignment to the grader by typing the following command:

~cs4103_chf/bin/p_copy 2

This command copies everything in your prog2 directory to the grader's account. Check whether all required files have been submitted successfully:

~cs4103_chf/bin/verify 2