CSE306: Spring 2020

Project 2 – Memory Management

Out: March 10
Due: April 7

What to do. Implement the *Memory* module of OSP2.

NOTE: You must use Java 8 or else your project will not link with OSP-2!!! Make sure that both

```
    javac -version
    java -version
```

say it's version 1.8.X_YYY

The project files are found on the Blackboard in the Assignments section.

Details of what you are supposed to do are given in the OSP2 manual, Chapter 5. The additional requirements are:

1. You have to implement a *modified* 2-handed clock (M2HC) page replacement algorithm similar to (but not exactly) the one described on pages 380-381 (UNIX and Solaris Memory Management) in the textbook. You also have to use the dirty-bit optimization (if you do not optimize for clean frames, OSP2 issues warnings, which is *not* OK and will cost points).

2. The M2HC page replacement algorithm works as follows.

   (a) The *first hand*, called the *cleaner*, sweeps all eligible for replacement frames[1] at regular intervals of 4000 clock ticks and decrements the *use counts* of these frames by 1 (but not less than 0). A use count is like a use bit, but it can have more than 1 bit (e.g., 2 bits) and thus can be used to count to more than 1. You must maintain the use count for every frame yourself. It should be incremented by 1 (to a maximum of 2) on every reference to the frame (and decremented by the cleaner, as explained). If a frame already has the use count = 0 then swap that page out, making it clean. To arrange for an asynchronous process to run at regular intervals, use the OSP2 feature called *daemon*. Read about daemons in the OSP2 manual.

   (b) At page faults, if no free frame is available, the *second hand*, called the *chooser*, sweeps the eligible frames and selects a *clean* page with the use count of 0 for replacement.

[1] You have to figure out which pages are eligible for replacement based on your study of the subject of memory management.

If no such frame, it selects a dirty frame with the use count 0. If *that* fails, it chooses some eligible frame at will.

Your objective in this project is to get your implementation to run under OSP2 without errors and warnings with the parameter files that appear in the `Misc` subdirectory (the files with the `.osp` extension). As explained earlier, occasional errors (every 7-10th run) should not be a cause of concern, if they are unrelated to memory management.

Statistics. Compare and explain your statistics with respect to those produced by Demo.jar. (Demo.jar uses a very simplistic algorithm for page replacement: it scans the entire frame table and chooses the first replaceable frame.) The statistics of interest are:

- The number of pages swapped in and out
- CPU utilization
- Service time per thread (avg turnaround time)
- Normalized service time per thread (avg normalized turnaround time)

These statistics are part of the snapshots in the file OSP.log, which is produced during the runs. The meaning of these statistics is explained both in the textbook and in the OSP2 manual.

Important notice on the use of Git. You **must** use Git to maintain your project files **and** you must make frequent commits to your repository. The Git repository must be in *Github classroom*. Follow this link to create a repository in this course's classroom for Project 2:

`https://classroom.github.com/a/bKQTom47`

After a few clicks you will get a repository named `cse306-project-2-`*YourGuthubId*. The repository will automatically become private and you should not attempt to change that. **This repository is different from that of Project 1.**

We will be checking your repository and your commit logs to make sure that substantial activity has been taking place over a period of time. If there are less than 6 **nontrivial** commits, significant penalty will apply. Other than that, same requirements to your code as in Project 1.

How to submit. Zip-up your main branch and submit the zip file via Blackboard. Github has a button for that: *Clone or download/Download ZIP*. Normally, the zip file will have a top folder with a name like `cse306-project-2-`*YourGuthubId*`-master`, but if not then make sure that your submitted zip file has a top folder with such a name. **Submission is through the Blackboard only.** Make sure your Github version is the same as in your Blackboard submission. If the two disagree, penalty will apply. Do **not** include the OSP.log file, Demo.jar, and the .class files generated by `javac` in the submitted .zip file. The easiest way to make sure they are not included is to not commit them into your Git repository.

Test to make sure that your .zip file contains everything that is necessary for the javac/java commands, described in Section 1.8 of the manual, to succeed.

This project is to be done **individually**. **Each** source file **must** include

- your name (as in Solar)
- student Id
- the following pledge:
  I pledge my honor that all parts of this project were done by me individually, without collaboration with anyone, and without consulting any external sources that provide full or partial solutions to a similar project.
  I understand that breaking this pledge will result in an "F" for the entire course.

Failure to include either of these items will require a re-submission and incur significant penalty.

GOOD LUCK!