School of Electrical Engineering and Computer Science
COMP2240/6240 - Operating Systems
Assignment 3 (15%)

Submit using Blackboard by **11:59 pm**, **Friday 30th October 2015**

In assignment 1, we assumed that the system had an infinite amount of memory. In this assignment, the operating system has a limited amount of memory and this needs to be managed to meet process demands. You will write a program that simulates *least recently used (LRU)* and *clock* page replacement algorithms for a virtual memory store.

- You are to simulate a system that uses paging (with no segmentation), where each page contains a single instruction.
- The system is to use a Round Robin short-term scheduling algorithm with time quantum of 3. Executing a single instruction takes 1 unit of time, and swapping in a page takes 6 units of time (if a page required by a process is not in main memory, the process must be put into its blocked state until the required page is available).
- The system has 30 frames available in main memory. During execution, the processor will determine if the page required for the currently running process is in main memory. If the page is in main memory, the processor will access the instruction and continue. If the page is not in main memory, the processor will issue a page fault and block the process until the page has been transferred to main memory.
- When a page fault occurs, the interrupt routine will handle the fault by placing an I/O request into a queue, which will later be processed by the I/O controller to bring the page into main memory. This may require replacing a page in main memory using a page replacement policy. Other processes should be scheduled to run while such an I/O operation is occurring.

You are to compare the performance of the **LRU** and **clock** page replacement algorithms for the following allocation/replacement strategies:

1. Fixed allocation with local replacement strategy

2. Variable allocation with global replacement strategy

For the fixed allocation strategy, all processes will receive the same number of frames (i.e. divide the total number of frames by the number of processes).

For the variable allocation strategy, processes that experience page faults will gradually grow in size.

Please use the basic versions of policies introduced in lectures. The simulation will be strictly demand paging, where pages are only brought into main memory when they are requested. Assume that all pages are read-only, so no page needs to be written back to disk.

Input and Output

Input to your program should be via **command line** arguments, where each argument is the name of a file that specifies the execution trace of a process. All processes start execution at time 0, and are entered into the system in the order of the command line arguments (with the first argument being the earliest process).

```
mysimulator process1.txt process2.txt process3.txt
```

Since we assume that each page contains a single instruction, an execution trace is simply a page request sequence.

For example: (process1.txt)

```
begin
1
3
2
1
5
4
3
end
```

This specifies a process that first reads page 1, then page 3, and so on until end is reached.

For each allocation/replacement strategy, the simulation should produce a summary showing, for each process, the total number of page faults, the time the page faults were generated, and the total turnaround time (including I/O time).

User Interface:

There are no marks allocated for using or not using a GUI – the choice is yours.

Programming Language:

The preferred programming language is Java 6.
Permissible programming languages are Java 6, Java 7, Java 8, C (gcc), C++ (g++).
If you wish to use any language other than the preferred programming language, you must first notify the course demonstrator (see post in Blackboard discussion board for more details).

Deliverable:

1. Program source code and a README file containing any special instructions required to compile and run the source code. If programmed in Java, your main class should be c9999999A3 (where c9999999 is your student number) i.e. your program can be executed by running "java c9999999A3". If programming in other languages, your code should compile to an executable named "c9999999A3".

2. Brief 1 page (A4) review of how you tested your program and a comparison of the page replacement methods based on the results from your program and any interesting observations.

Please submit all files and the 1 page report plus a copy of the official assignment cover sheet in a ZIPPED folder through Blackboard. The folder name should be "c9999999A3" and the zip file should be name as c9999999A3.zip (where c9999999 is your student number).

NOTE: Assignments submitted after the deadline (**11:59 pm Friday 30th October 2015)** will have the maximum marks available reduced by 10% per 24 hours.