San Jose State University

Department of Computer Science
Data Structures and Algorithms (CS 149-07)

Instructor: Ahmed Ezzat
Homework #4 (10 pts)

Swapping and Paging

The purpose of this assignment is to explore the various memory management algorithms for swapping and paging. You will write a series of small simulation programs in Java, C, or C++ (your choice). As in Assignment #2, you will generate simulated processes. You should not need to make any process management system calls.

Swapping

Assume main memory has 100 MB for swapping with variable-sized partitions. Processes have randomly and evenly distributed sizes of 5, 11, 17, and 31 MB. Processes have randomly and evenly distributed durations of 1, 2, 3, 4, or 5 seconds.

Write programs that simulate the **first fit, next fit, and best fit memory allocation algorithms**.

Run each algorithm 5 times simulating 1 minute each time to get an average of the number of processes successfully swapped into memory (but not necessarily completed) during the minute. Generate around 150 new random processes before each run, and allocate them into memory in the order that you generated them according to the algorithm. Once they're in memory, the processes run independently and simultaneously for their durations.

For each algorithm, print the average number of processes that were successfully swapped in. Each time a process is swapped in or completes (and therefore is removed from memory), print a memory map, e.g., **AAAAA.... bbbbbbbbbbb.. 33333** where the characters are the process names (one character per MB) and the dots are holes (one per MB). You can reuse process names that aren't currently in the map. Indicate which process entered or left. For an entering process, also print its size and duration.

For each algorithm, print the average number of processes (over the 5 runs) that were successfully swapped in.

Paging

You are running a process that consists of 10 pages numbered 0 through 9. Physical memory has 4 page frames. There are always 10 page frames available on disk. When the process starts, none of its pages are in memory.

The process makes random references to its pages. Due to **locality of reference**, after referencing a page i, there is a 70% probability that the next reference will be to page i, i-1, or i+1. i wraps around from 9 to 0. In other words, there is a 70% probability that for a given i, Δi will be -1, 0, or +1. Otherwise, |Δi| > 1.

Suggested procedure

- To compute the next i, first generate a random number r from 0 through 9.
- If 0 ≤ r < 7, then generate a random Δi to be -1, 0, or +1.
- For 7 ≤ r ≤ 9, randomly generate 2 ≤ Δi ≤ 8.

Simulate the page replacement algorithms FIFO, LRU, LFU, MFU, and random pick. Run each algorithm 5 times, 100 page references each time, to compute an average hit ratio of pages already in memory. For each reference, print the page numbers of the pages in memory and (if any) which page needed to be paged in and which page was evicted.

What to turn in

Email a zip file to Vaibhav Tupe, Email: vaibhav.tupe@sjsu.edu that contains:

- Your source files.
- Your output.

Email your answers to Vaibhav Tupe, Email: vaibhav.tupe@sjsu.edu. Your subject line should be **CS 149**-section **Assignment #4**, your group name for example **CS 149-7 Assignment #4, Group-n**. Cc all your group members so that the grader can do a “Reply all” when needed. This is a group assignment; all group members will receive the same score.