MIDWESTERN STATE UNIVERSITY

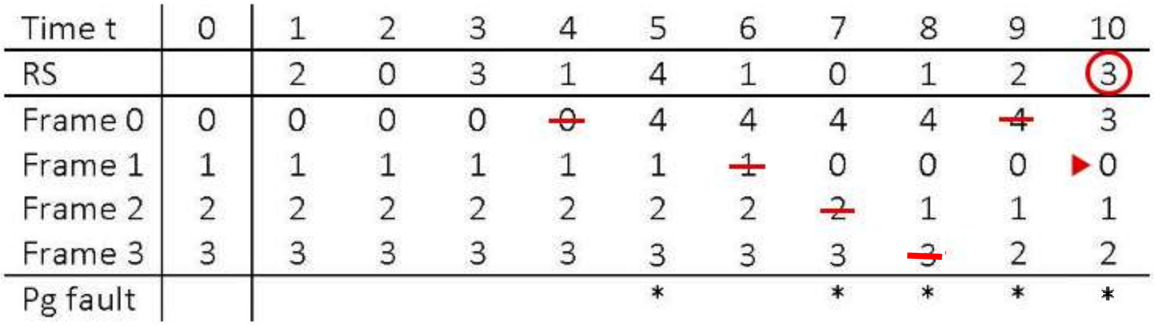
DEPARTMENT OF COMPUTER SCIENCE

CMPS 4103- Introduction to Operating Systems
Fall semester 2022

Mini project #4 – Paging - due date 11/29

Problem.
You just learned about page replacement algorithms and you would like to know which algorithm will be more effective in the operating system that you will develop after graduation. To make it simple, you decided to compare the results of two of those algorithms: FIFO and Second chance. Your mission is to write a small simulation program under the following assumptions: there are a maximum of 64 virtual pages running in a physical memory with 10 frames. You will use a random number generator to determine which address in the range 0 to 4095 is the next to be accessed (12 bit addresses). You will extract the page number from that address and if the page is already in the physical memory, nothing happens (make sure its reference bit is marked one), otherwise if a frame is available then the page will be assigned to that frame, finally if there are no frames available you will then apply the replacement algorithm to decide which frame to use. To measure the effectiveness of the two algorithms, you will run the simulation for 10,000 pages and report the number of page faults for each algorithm. Any access to a page counts as a reference, including the initial load. You will assume that the reference bit is reset only when running the second chance algorithm. You will hand in the printout of your source code and the output values on the due date (no late work will be accepted). You can use Visual Studio, Cygwin or any other system you may have access to but the programming language must be C or C++.

Example: in a four frame physical memory (frames 0 t0 3) and 14 page requests in the range 0 to 4. FIFO, shown below, produces 9 page faults (four requests happened at time zero in the order 0,1,2,3, all page faults, then times 5, 7, 8, 9, 10)



Time t	0	1	2	3	4	5	6	7	8	9	10
RS		2	0	3	1	4	1	0	1	2	3
Frame 0	0	0	0	0	~~0~~	4	4	4	4	~~4~~	3
Frame 1	1	1	1	1	1	1	~~1~~	0	0	0	▶0
Frame 2	2	2	2	2	2	2	2	~~2~~	1	1	1
Frame 3	3	3	3	3	3	3	3	3	~~3~~	2	2
Pg fault						*		*	*	*	*

Using the second chance algorithm (not shown in the diagram above), at time 0, there were four page faults in the order 0, 1, 2, 3. At time 5, page 0 has been referenced (time 0 and 2) so it goes to the end of the queue (reference erased), same happens with pages 1, 2 and 3, so page 0 gets to the front again and it is replaced. At time 7, page 0 is back, page 1 has been referenced between time 4 and 6, so it goes to end of the line, page 2 is replaced (its reference bit had been reset at time 5). Next page fault will be at time 9, the head of the list is page 3, which was not referenced since time 5, so 3 is replaced by 2. At time 10, 3 is back. The head of the line is page 4, which will be replaced. The result is 8 page faults (four initial ones at time zero and then at time 5, 7, 9, and 10).