CPSC 351 Project: Virtual Memory Manager, due 28 Nov 2020

Your name: Hai Nguyen

Repository *(print)*: **https://github.com/hai-nguyen93/cpsc351-virtual-mem-mgr**

Verify each of the following items and place a checkmark in the correct column. Each item incorrectly marked will incur a 5% penalty on the grade for this assignment

Finished	Not finished	
X		Created functions that correctly calculate the offset and page of a given virtual address
X		Created a page table, that contains the frame of a given page, and which will page fault if the desired page is not in memory (this will happen: (A) when the program is first run and physical memory is empty, and (B) if only half as many physical frames as pages in the page table
X		Given a given logical address, checks the page table to find the corresponding physical address
X		Correctly reads the given physical address for the char value stored there
X		Goes to the BACKING_STORE and reads in the corresponding page into a free frame in physical memory. If there are only 128 frames, it must replace a frame to do this.
X		Implemented a Translation Lookaside Buffer (TLB) to store the most recently read-in page, AND checks the TLB first when decoding a logical address.
X		Do following when reading a logical address that is not in the TLB/Page table: Check TLB → (TLB miss) Check Page Table → (Page table miss) Page fault → read page from BACKING_ STORE → updates physical memory → updates Page table → updates TLB → reads value from physical memory..
X		Follows this flow diagram when has a TLB hit: Check TLB → Gets frame and offset → reads value from physical memory
X		Do following when has a TLB miss but a Page table hit → Check TLB → (TLB miss) → Checks Page table → Updates TLB → Gets frame and offset → reads value from physical memory
X		**Page-fault rate** -- the percentage of address references that resulted in page faults.
X		**TLB hit rate** -- the percentage of address references that were resolved in the TLB
X		Now modify your program so that it has only 128 page frames of physical memory (but still has 256 entries in the page table)
X		Program now keeps track of the free page frames, as well as implementing a **pagereplacement policy using either FIFO or LRU**
X		Project directory pushed to new GitHub repository listed above

Fill out and print this page, and submit it on Titanium on the day this project is due.