Submission

Turn in the code for this homework by uploading your project to a public repository on GitHub. While you may discuss this homework assignment with other students, you must complete the work on your own.

To complete your submission, print the following sheet, fill out the spaces below, and submit it to Titanium by the deadline. Failure to follow the instructions exactly will incur a **10%** penalty on the grade for this assignment.

CPSC 351 Project: Virtual Memory Manager, due 6 May 2021

Your name: Carson Carpenter

Repository: https://github.com/carsoncarpenter7/CS-351-Virtual-Memory.git

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	
☑	☐	**Created functions that correctly calculate the offset and page of a given virtual address**
☑ This is implemeneted when pagefaulting.	☐ mostly but still has small errors	**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**
☑	☐	**Given a given logical address, checks the page table to find the corresponding physical address**
☑	☐	**Correctly reads the given physical address for the char value stored there**
☐	☑	**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.**
☑	☐	**Implemented a Translation Lookaside Buffer (TLB) to store the most recently read-in page, AND checks the TLB first when decoding a logical address.**
☐	☑	**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..**
☑	☐	**Follows this flow diagram when has a TLB hit: Check TLB → Gets frame and offset → reads value from physical memory**
☑	☐	**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**
☑	☐	**Page-fault rate -- the percentage of address references that resulted in page faults.**

☑ I think so? Hit rate is 0 so not sure if value is updating	☐	**TLB hit rate -- the percentage of address references that were resolved in the TLB**
☐	☑	**Now modify your program so that it has only 128 page frames of physical memory (but still has 256 entries in the page table)**
☐	☑	**Program now keeps track of the free page frames, as well as implementing a pagereplacement policy using either FIFO or LRU**
☑	☐	**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.