Homework 4

CSCE 5640: Operating System Design, Fall 2024

Due 11/30/2024 at 11:59pm

This is the last homework of the course. All programs must compile and execute on the CSE machines! This is an individual assignment.

1. (10) Consider a paging system with the page table stored in memory.
    a. If a memory reference takes 60 nanoseconds, how long does a paged memory reference take?
    b. If we add TLBs, and if 80 percent of all page-table references are found in the TLBs, what is the effective memory reference time? (Assume that finding a page-table entry in the TLBs takes 3 nanoseconds, if the entry is present.)

    In answering the above two questions, briefly justify your answers.

    Keys:

    a. Since the page table is stored in memory, it will take two memory accesses for every memory reference. That is, one for page table reference and another for the actual data in the memory after identifying the frame number. Therefore, it will take **2 * 60 ns = 120 ns** for a paged memory reference.
    b. Since the TLB hit is 80%, 20% of the virtual addresses mapping to physical address will be required to check the page table which is in the main memory. Also, finding a page-table entry in the TLBs takes 3 ns for every TLB hit. Therefore, the effective memory reference time = **0.8 * (3 + 60) ns + 0.2 * (2 * 60) ns = 74.4 ns.**

2. (30) Assume that a system has a 32-bit virtual address with a 2-KB page size. Write a C++ program that takes a virtual address (in decimal) on the command line and has it output the page number and offset for the given address. Your program should handle error input as well.
    Sample outputs:
    $./addresses
    Usage: ./address virtual_address

    $./addresses -45
    Please provide a valid virtual address.

    $./addresses 19986
    The address 19986 contains:
    Binary: 00000000000000000100111000010010
    Number of bits for page number = 21
    Number of bits for page offset = 11
    Page number binary = 00000000000000001001
    Page offset binary = 11000010010
    Page number = 9
    Page offset = 1554

    Writing this program will require using the appropriate data type to store 32 bits. Use unsigned data types as well.

Comment your code and submit it in a file named **paging.cpp.**

3. (20) Assume that memory contains three holes of 10 MB each. A sequence of 14 requests for 1MB each is to be processed. For each of the memory allocation methods listed below, determine the sizes of each of the remaining holes after all 14 requests have been satisfied (assume that the direction of searching the holes is from top to bottom or left to right):
    a. first fit
    b. best fit
    c. worst fit

Keys:

a. First fit:

After 14 requests for 1 MB each: first hole will be used completely, 4 MB of the second hole will be used with remaining 6 MB unused. All of the last hole is unused.

So, sizes of holes: first = 0 MB, second = 6 MB, third = 10 MB.

b. Best fit:

After 14 requests for 1 MB each: first hole will be used completely, 4 MB of the second hole will be used with remaining 6 MB unused. All of the last hole is unused.

So, sizes of holes: first = 0 MB, second = 6 MB, third = 10 MB.

c. Worst fit:

After 14 requests for 1 MB each: 5 MB of the first hole will be used, 5 MB of the second hole will be used, and 4 MB of the third hole will be used.

So, sizes of holes: first = 5 MB, second = 5 MB, third = 6 MB.

4. (10) Consider the working set model with Δ=3. Given the following reference string of a process p:
    y x x x x x x y y u x x x y z y z w w z x x w w

    a. What is the largest working set process p will ever have?
    b. What is the smallest working set process p will ever have (not counting the first Δ references)?

Keys:

1. {*y, u, x*} = 3. *Note other sets with the same number of references are also available.*
2. {x} = 1

5. (30) Consider the following page reference string:
    3, 1, 4, 2, 5, 4, 1, 3, 5, 2, 0, 1, 1, 0, 2, 3, 4, 5, 0, 1

Assuming demand paging with three frames, how many page faults would occur for the following replacement algorithms? (Show your results similar to Figure 10.12, 10.14, 10.15)

a. FIFO replacement

b. Optimal replacement
c. LRU replacement

Keys:

a. FIFO:

3, 1, 4, 2, 5, 4, 1, 3, 5, 2, 0, 1, 1, 0, 2, 3, 4, 5, 0, 1

3	3	3	2	2		2	3		3	3	1				1	1	5	5	5
	1	1	1	5		5	5		2	2	2				3	3	3	0	0
		4	4	4		1	1		1	0	0				0	4	4	4	1

Total page faults: 15.

b. Optimal:

3, 1, 4, 2, 5, 4, 1, 3, 5, 2, 0, 1, 1, 0, 2, 3, 4, 5, 0, 1

3	3	3	2	5			5		2	2					3	4	5		
	1	1	1	1			1		1	1					1	1	1		
		4	4	4			3		3	0					0	0	0		

Total page faults: 11.

c. LRU:

3, 1, 4, 2, 5, 4, 1, 3, 5, 2, 0, 1, 1, 0, 2, 3, 4, 5, 0, 1

3	3	3	2	2		1	1	1	2	2	2				2	2	5	5	5
	1	1	1	5		5	3	3	3	0	0				0	4	4	4	1
		4	4	4		4	4	5	5	5	1				3	3	3	0	0

Total page faults: 16.

Submission:

- We will be using an electronic homework submission on Canvas to make sure that all students submit their programming tasks on time. You will submit both the homework 4 files to the Homework 4 Dropbox on Canvas by the due date and time.
- Make sure you submit the following files for this homework (do not submit a zip file):
  - **paging.cpp**
  - **hw4sol.docx (or pdf)** (which contains answers to questions # 1, 3, 4, and 5)
    - **Please write your answer in a word document instead of handwriting.**
- Program submissions will be checked using a code plagiarism tool against other solutions, including those found on the Internet, so please ensure that all work submitted is your own. Any student determined to have cheated will receive an **'F'** in the course and will be reported for an academic integrity violation.
- As a safety precaution, do not edit your program (using vim or nano) after you have submitted your program where you might accidentally re-save the program, causing the timestamp on your file to be later than the due date. If you want to look (or work on it) after submitting, make a copy of your submission and work off that copy. Should there be any issues with your submission, this timestamp on your code on the CSE machines will be used to validate when the program is completed.