CSE 120: Homework 3

Merrick Qiu

Nachos VM Worksheet

1. 4
2. 8
3. (a) pageTable[0].vpn = 0
   (b) pageTable[0].ppn = 2
   (c) pageTable[1].vpn = 1
   (d) pageTable[1].ppn = 0
   (e) pageTable[2].vpn = 2
   (f) pageTable[2].ppn = 6
   (g) pageTable[3].vpn = 3
   (h) pageTable[3].ppn = 4
4. 128*2 = 256 bytes
5. 128*6 = 768 bytes
6. It is in virtual page 2, which is in physical page 6.
7. 298-256 = 42 bytes
8. 128*6 + 42 = 810 bytes
9. Since the virtual and physical pages aren't always the same, memory that is contiguous in the virtual address space might not be contiguous in the physical address space, so we would need to check if there are page boundaries for the array we are copying.

Question 2

1. 200 ns
2. 0.75*100 + 0.25*200 = 125 ns
3. 0.995*100 + 0.005*200 = 100.5 ns

Question 3

1. 22 bits are left.
2. We can take the first 22 bits and then shift by 10 bits(page size is 2^{10})

$$\begin{aligned} 0x0000FFFF &= 00000000000000001111111111111111 \\ &\implies 00000000000000001111110000000000 \\ &\implies 00000000000000000000000000111111 \\ &= 0x3F = 63 \end{aligned}$$

3. The offset is the last 10 bits which is 0b1111111111 = 0x3FF = 1023.
4. 4*1024 = 4096 bytes
5. 4096 + 1023 = 5119 bytes

Question 4

1. $2^{44-16} = 2^{28}$
2. A page can have 2^14 entries so we need 14 bits to index into a page table. The offset will be 16 bits since that is the size of a page. Since there are two tables, we have a total of 14 + 14 + 16 = 44 bits.
3. 4 gigabytes is 2^32 bytes which is 2^16 pages. Since page can hold 2^14 entries, we need 4 pages, plus 1 root page table. This is $2^16 + 5$ page frames.

Question 5

For a 2 nanosecond instruction, it would be $2 + 10000000 * \frac{1}{20000000} = 2.5ns$. For a 1 nanosecond instruction, it would be 1.5 ns

Question 6

There are a total of 8 page faults for FIFO

Reference History	Queue	Faults
4	4	1
42	42	2
427	427	3
4272	427	4
42725	4275	4
427253	2753	5
4272533	2753	5
42725332	2753	5
427253323	2753	5
4272533231	7531	6
42725332312	5312	7
427253323126	3126	8

There are a total of 7 page faults for LRU

Reference History	Queue	Faults
4	4	1
42	42	2
427	427	3
4272	472	4
42725	4725	4
427253	7253	5
4272533	7253	5
42725332	7532	5
427253323	7523	5
4272533231	5231	6
42725332312	5312	6
427253323126	3126	7

Question 7

The working set page replacement algorithm tries to keep the pages that the processes regularly need, the working set, so that page faults are not too burdensome.

Question 8

1. The CPU speed is not what is stopping the utilization from increasing.
2. Probably wont help since the storage of the disk is not the issue
3. Multiprogramming will probably decrease utilization since they will require more memory
4. Decreasing multiprogramming will free up memory and increase cpu utilization

5. Increasing main memory will increase cpu utilization by decreasing the amount of page faults
6. It will help
7. Can help if the algorithm works correctly
8. Doesn't change much.