Srinivas Simhan
CIS 450 - Homework #2
Due: 4/11/19

1. **Question 1**
    a. Bits in Logical Address
        i. (8 * 1024) = ((2^3) * (2^10)) = (2^13) ===> 13 bits in logical address
    b. Bits in Physical Address
        i. (32 * 1024) = ((2^5) * (2^10)) = (2^15) ===> 15 bits in physical address
2. **Question 2**
    a. 20000
        i. 4KB
            1. Page: 20000/4096 ~ 4
            2. Offset: 20000 - (4*4096) = 3616
        ii. 8KB
            1. Page: 20000/8192 ~ 2
            2. Offset: 20000 - (2*8192) = 3616
    b. 32768
        i. 4KB
            1. Page: 32768/4096 ~ 8
            2. Offset: 32768- (8*4096) = 0
        ii. 8KB
            1. Page: 32768/8192 ~ 4
            2. Offset: 32768- (4*8192) = 0
    c. 60000
        i. 4KB
            1. Page: 60000/4096 ~ 14
            2. Offset: 60000- (14*4096) = 2656
        ii. 8KB
            1. Page: 60000/8192 ~ 7
            2. Offset: 60000 - (7*8192) = 2656
    d. 90000
        i. 4KB
            1. Page: 90000/4096 ~ 21
            2. Offset: 90000 - (21*4096) = 3984
        ii. 8KB
            1. Page: 90000/8192 ~ 4
            2. Offset: 90000- (10*8192) = 8080
    e. Table with results (page, offset):

	4 KB (4096)	**8 KB** (8192)
20000	(4, 3616)	(2, 3616)
32768	(8,0)	(4, 0)
60000	(14, 2656)	(7, 2656)
90000	(21, 3984)	(10, 8080)

Srinivas Simhan
CIS 450 - Homework #2
Due: 4/11/19

3. **Question 3**
    a. If one-level paging is used, the number of page table entries needed in the table page is:
        i. ((2^32) / (2^12)) = 1,048,576 pages.
    b. You will need three page table entries due to needing one in the top-level table, and one in each of the lower-level tables
4. **Question 4**
    a. 0, 430
        i. 219 + 430 = 649
    b. 1, 010
        i. 2300 + 10 = 2310
    c. 2, 500
        i. Illegal reference
    d. 3, 400
        i. 1327 + 400 = 1727
    e. 4, 112
        i. Illegal reference
5. **Question 5**
    a. Improving CPU Utilization
        i. Install a faster CPU
            1. Won't help, the CPU is already not being used completely.
        ii. Install a bigger paging disk
            1. Won't help, size of disk isn't an issue. The issue comes from the need for processes to access memory, hence causing page faults
        iii. Increase the degree of multiprogramming
            1. Won't help, we already have too many processes running
        iv. Decrease the degree of multiprogramming
            1. Will help, it reduces the number of processes trying to access memory
        v. Install more main memory
            1. Will help, there's less that needs to be swapped to the disk since it can be held in the RAM
        vi. Install a faster hard disk
            1. Will help, allows pages to be swapped out faster
        vii. Add prepaging to the page-fetch algorithm
            1. Depends, it might load in correct pages and improve utilization, or it might load in wrong pages and provide no solution to the issue
        viii. Increase the page size
            1. Will help, each process has pages allocated to it, but it won't be helpful to those with few pages allocated to them since the number of page faults won't change

Srinivas Simhan
CIS 450 - Homework #2
Due: 4/11/19

6. **Question 6**

	1 Frame	**2 Frames**	**3 Frames**	**4 Frames**	**5 Frames**	**6 Frames**	**7 Frames**
LRU	20	18	15	10	8	7	7
FIFO	20	18	16	14	10	10	7
MIN	20	15	11	8	7	7	7

7. **Question 7**
    a. Contiguous
        i. Sequential
            1. Works well, file is stored contiguously (adjacent) on the disk
            2. Sequential access means that it iterates over the contiguous
        ii. Random
            1. Works well, easy to find the adjacent block containing the position that needs to be found on the disk
    b. Linked
        i. Sequential
            1. Works well, have to follow from on block on the disk to the next
        ii. Random
            1. Doesn't work, will need to follow multiple disk blocks before finding destination
    c. Indexed
        i. Sequential
            1. Works well, iterates to specific index easily
        ii. Random
            1. Works well, iterates to specific index easily
8. **Question 8**

Block	1	10	200	500	1,000	2,200	10,000	100,000	1,000,000	1,400,000	5,000,000
Reads	1	1	2	3	3	3	3	4	4	4	4