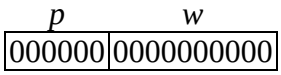
CSC 262
Homework #10
Due in class Wednesday, April 25.

1. Consider the following simple paged virtual memory system, with 16-bit virtual addresses divided as follows: bits 10-15 are the page name (*p*) and bits 0-9 are the offset into the page (*w*). The physical memory is 16K words in size.



a. How large are the pages on this system? How many (virtual) pages are there? How many page frames?

Now assume that at some point in time, the first lines of the page table look as follows:

Page	*Resident*	*Modified*	*Permission*	*Frame*
0:	1	0	0	0000
1:	0	0	0	0001
2:	0	0	1	1100
3:	0	0	1	0101
4:	1	1	1	1100
5:	0	0	1	0010
6:	1	0	1	1010
…	…	…	…	…

In this table, *Resident* indicates whether a page is currently in the physical memory; *Modified* indicates whether a page in memory has been written to and therefore needs to be copied back into the backing store when the page is evicted. *Permission* indicates whether the current process has permission to access the page. Finally, *Frame* gives the frame number where the page is stored. (If the page is not resident then the value is meaningless.) Multiplying the value in *Frame* by the frame size gives the base address for the frame. The table itself is organized so that the entry for page 0 is at the top and the entry for page 7 is at the bottom.

For the following questions, compute the physical address that will be generated from a given virtual name. If a hardware fault is generated instead, state kind of fault that will take place.

b. Virtual address 6633 (= 0001101000000001 binary).
c. Virtual address 4351 (= 0001000011111111 binary).
d. Virtual address 1365 (= 0000010101010101 binary).
e. Virtual address 4095 (= 0000111111111111 binary).
f. Virtual address 4096 (= 0001000000000000 binary).
g. Virtual address 0 (= 0000000000000000 binary).

2. As we did in class, compute the state of memory and page faults (if any) generated by each request, for the reference string shown below, using each of the replacement policies requested. Assume that exactly four page frames are available, and that they are empty at the start of the simulation.

w = 1,2,3,4,1,5,2,6,1,3,4,2,1,5,7,6,7,5,4,7,6,3,7,6

a. FIFO

b. LFU (with LRU as tiebreaker)

c. LRU

d. CLOCK (Assume the use bit is set when a page is brought in on a fault. See Stallings p. 357)

e. OPT