Anisha Aggarwal
CIS 415
Assignment 3

1. OSC 7.16
    a. Increase *Available* – It can be made safely because there are more resources becoming available to the processes. If we were safe before increasing *Available*, we could not become unsafe when adding resources.
    b. Decrease *Available* – It can't be made safely because we might be removing resources that are needed by processes. Before decreasing Available, the safe state was based off of the number of resources available to the processes, so by decreasing them we may move into an unsafe circumstance.
    c. Increase *Max* for 1 process – It can't be made safely if the new *Max* is greater than *Available*. However, even if the new *Max* is less than *Available*, we will have run the safety algorithm to check if it is safe.
    d. Decrease *Max* for 1 process – It can be made safely because if we were safe with a higher *Max*, we can't become unsafe when decreasing the *Max*.
    e. Increase the number of processes – It can't be made safely because the new number of processes may require a number of resources greater than *Available*.
    f. Decrease the number of processes – It can be made safely because by removing processes, we are decreasing number of resource requests.

2. OSC 8.13
    a. External fragmentation

        Both **pure segmentation** and **contiguous memory allocation** cause external fragmentation. In pure segmentation, each segment varies in size and loaded into the memory contiguously. In contiguous memory allocation, each process is loaded into the memory contiguously. In both of these, holes are created in the memory that sometimes becomes unusable. **Pure paging** doesn't create external fragmentation because the processes can use any free memory frame available.
    b. Internal fragmentation

        Due to fact that **pure paging** has fixed page sizes, this makes internal fragmentation possible because if a process is only occupying a small chunk of the page, another process can't be allocated to occupy the rest of the page causing internal fragmentation. Since both **pure segmentation** and **contiguous memory allocation**, size is allocated based on the size, internal fragmentation would not occur.
    c. Ability to share code across process

        In **pure paging,** the code can be shared by having the page table entries mapping to the same frame number. In **pure segmentation**, the code can also be shared by having the segmentation table entries mapped to the same base address. In **contiguous memory**

allocation, there is no way to share the code because there is no mapping system.

3. OSC 8.25
   a. If a memory reference takes 50 nanoseconds, how long does a paged memory reference take?
      100 nanoseconds because there will be 2 memory references. One when we are accessing the page table and the other when we have gotten the frame number from the page table and accesses the memory frame.
   b. If we add TLBs, and 75% of all page-table references are found in the TLBs, what is the effective memory reference time? (page-table entry in TLBs takes 2 nanoseconds if entry is present)
      First the MMU will search in the TLB, which will cause a constant 2 nanoseconds adder. Knowing that 75% of all page-table references that will be found, they will take 50 nanoseconds because it will have to make 1 memory reference. The other 25% of page-table references will be misses, so it will take 100 nanoseconds for the 2 memory references.

      2 nanoseconds + (.75(50 nanoseconds) + .25(100 nanoseconds))
      = 64.5 nanoseconds

4. OSC 9.14
   a. TLB miss with no page fault – this can occur
      Even though there is a TLB miss, the page can still be in the memory and hence there will not be a page fault.
   b. TLB miss and page fault – this can happen
      When there is a TLB miss, it means that the page translation was not found in the TLB. If it is also not found in the memory, there will be a page fault.
   c. TLB hit and no page fault – this can happen
      If there is a TLB hit, then no page fault will occur because that means we have found the page number mapped to a frame number.
   d. TLB hit and page fault – this can't happen
      If there is a TLB hit, it means that the page number is mapped to a frame number. Therefore there will not be a page fault.

5. OSC 9.21 (reference string: 7, 2, 3, 1, 2, 5, 3, 4, 6, 7, 7, 1, 0, 5, 4, 6, 2, 3, 0, 1)
   a. LRU replacement

7	7	7	1	1	1	3	3	3	7	7	7	7	5	5	5	2	2	2	1
	2	2	2	2	2	2	4	4	4	4	1	1	1	4	4	4	3	3	3
		3	3	3	5	5	5	6	6	6	6	0	0	0	6	6	6	0	1
7	2	3	1	2	5	3	4	6	7	7	1	0	5	4	6	2	3	0	1

page faults = 18

b. FIFO replacement

7	7	7	**1**	1	1	1	1	**6**	6	6	6	**0**	0	0	**6**	6	6	**0**	1
	2	2	2	2	**5**	5	5	5	**7**	7	7	7	**5**	5	5	**2**	2	2	**1**
		3	3	3	3	3	**4**	4	4	4	**1**	1	1	**4**	4	4	**3**	3	3
7	2	3	1	2	5	3	4	6	7	7	1	0	5	4	6	2	3	0	1

page faults = 17

c. Optimal replacement

7	7	7	**1**	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	2	2	2	2	**5**	5	5	5	5	5	5	5	5	**4**	**6**	**2**	**3**	3	3
		3	3	3	3	3	**4**	**6**	**7**	7	7	**0**	0	0	0	0	0	0	0
7	2	3	1	2	5	3	4	6	7	7	1	0	5	4	6	2	3	0	1

page faults = 13