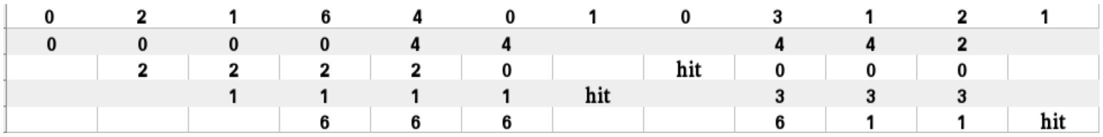
OS lab-6
Readme

Harshita Kalani (B20CS019)

1. Unzip the zipped file **B20CS019_OS_lab6.**
2. Open the file in any code editor and run the following command:
   a. **sh pg.sh or**
   b. **g++ page_replacement.cpp -o pg**
      ./pg
3. Enter the following:
   a. Number of frames.
   b. Number of page references
   c. Page references
4. There are 3 functions in the **page_replacement**.**cpp** file which represent the following 3 page replacement algorithms.

   Page Replacement Algorithms in OS that use paging for memory management, page replacement algorithms are needed to decide which page needs to be replaced when a new page comes in. Whenever a new page is referred and not present in memory, page fault occurs and the OS replaces one of the existing pages with a newly needed page. Different page replacement algorithms suggest different ways to decide which page to replace. The target for all algorithms is to reduce the number of page faults.

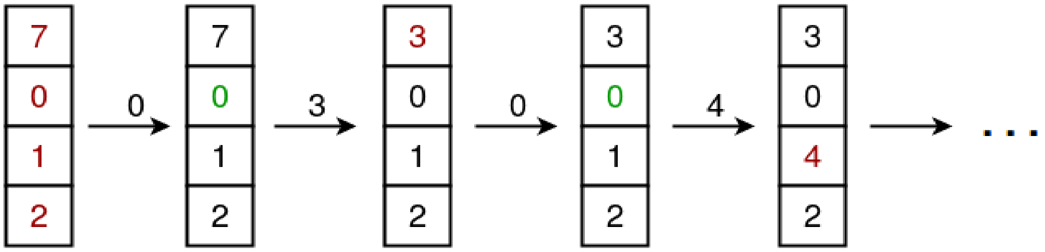
1. **FIFO page replacement algorithm**
   a. This is the simplest page replacement algorithm. In this algorithm, the operating system keeps track of all pages in the memory in a queue, the oldest page is in the front of the queue. When a page needs to be replaced, the page in the front of the queue is selected for removal.
   b. Consider the following reference string: 0, 2, 1, 6, 4, 0, 1, 0, 3, 1, 2, 1. Using FIFO page replacement algorithm –



0	2	1	6	4	0	1	0	3	1	2	1
0	0	0	0	4	4			4	4	2	
	2	2	2	2	0		hit	0	0	0	
		1	1	1	1	hit		3	3	3	
			6	6	6			6	1	1	hit

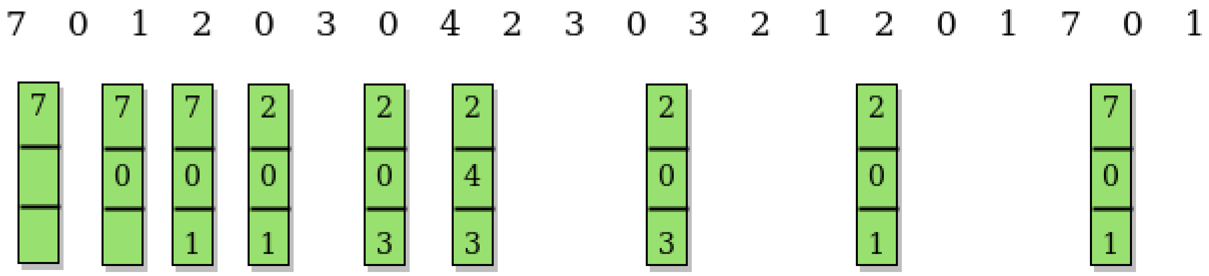
   So, total number of page faults = 9. Given memory capacity (as number of pages it can hold) and a string representing pages to be referred, write a function to find number of page faults
   c. **Belady's anomaly:**
   Belady's anomaly proves that it is possible to have more page faults when increasing the number of page frames while using the First in First Out (FIFO) page replacement algorithm. For example, if we consider reference string 3, 2, 1, 0, 3, 2, 4, 3, 2, 1, 0, 4 and 3 slots, we get 9 total page faults, but if we increase slots to 4, we get 10 page faults.

2. **Lru(least recently used) page replacement**
    a. The Least Recently Used (LRU) algorithm is a Greedy algorithm where the page to be replaced is least recently used. The idea is based on locality of reference, the least recently used page is not likely.
    b. Let say the page reference string 7 0 1 2 0 3 0 4 2 3 0 3 2 . Initially we have 4 page slots empty.



Total Page faults = 6

3. **Optimal page replacement**
    a. In the optimal page replacement algorithm, the OS replaces the page that will not be used for the longest period of time in future. Examples :
    b. If the referred page is already present, increment the hit count.
    c. If not present, find a page that is never referenced in future. If such a page exists, replace this page with a new page. If no such page exists, find a page that is referenced farthest in future. Replace this page with a new page.


7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1
7
7
0
7
0
1
2
0
1
2
0
3
2
4
3
2
0
3
2
0
1
7
0
1