CSE 312 HW2 REPORT

EMRE SEZER
1901042640

I created 6 dynamic arrays with malloc provided by the OS (Source code of the video on the pdf file). 3 arrays representing disks, 1 memory array, 1 array for lru, 1 page table.

I wrote 3 different sort algorithms. Bubble sort, quick sort and insertion sort.

When it tries to reach an element of the array. First, it checks if seeked element is at the memory. It checks by searching the page table from 0th element to last element. If it finds it at the memory, it returns that value. If it can’t find it at the memory, it finds an available page table entry by using one of the 3 algorithms I will explain next. Then, founded page table entry is modified with the value and the index of the seeked element of the array.

When it tries to change value of an element of the array. First, it checks if seeked element is at the memory. It checks by searching the page table from 0th element to last element. If it finds it at the memory, it changes that value and changes modified bit to true. If it can’t find it at the memory, it finds an available page table entry by using one of the 3 algorithms I will explain next. Then, founded page table entry is modified with the change value and the index of the array.

!!!!!!!

Main point is first check memory, if page is there no more actions is needed. If it is not there replace a page using an algorithm (depends on the method) and put the new page’s information at the position of the replaced page at the page table.

I wrote getValue and setValue functions. getVaulue is for getting the i’th value of the array at the disk. setVaulue is for changing the i’th value of the array at the disk with a new one. At sort functions I used these 2 functions for getting and setting.

!!!!!!!

Page Replacement Algorithms:

First In First Out Algorithm:

First it checks each page table entry’s valid bit. It stores that page on the first page entry with valid false valid bit. If it can’t find one, it increments fifoCounter by 1. If fifoCounter is exceeds the size it calculates mod of the fifoCounter and sets to fifoCounter. If replaced page’s modified bit is true. Then, that page table’s entry’s index th elemens of the array at the disk is modified with the value on the memory spot that page table’s entry points.

Second Chance Algorithm:

First it checks each page table entry’s valid bit. It stores that page on the first page entry with valid false valid bit. If it can’t find one, it checks the page table element that secondChancePointer points. It checks that page table entry’s referenced bit. If it is false, replaces that page with the new desired one. Else, it increments the secondChancePointer by 1 and changes that page table entry’s referenced bit to false. It repeats those steps until it finds a page table entry with false referenced bit. If replaced page’s modified bit is true. Then, that page table’s entry’s index th element of the array at the disk is modified with the value on the memory spot that page table’s entry points. Whenever a new page is put on the page table that page’s referenced bit is set to true.

Least Recently Used Algorithm:

It holds a counter for each page table entry at the page table in an array. When a page is referenced, its counter in the array is increased by 1. When that page is replaced its counter is set to 0.
First it checks each page table entry’s valid bit. It stores that page on the first page entry with valid false valid bit. If it can’t find one, it finds the page table entry with the least counter and replaces that page.
If replaced page’s modified bit is true. Then, that page table’s entry’s index th element of the array at the disk is modified with the value on the memory spot that page table’s entry points.

Page Table:

Page table is an array with 5 page table entries. Each page table entry points to an address at the memory array. Page table entry represents a page. When a page replacement is needed. New page's information is put to the replaced page's position at the page table. Simply, it overwrites the information, no extra work is done for page replacement.

Its variables are explained below:

Page Table Entry:

address: It holds the memory address that this page table entry points.

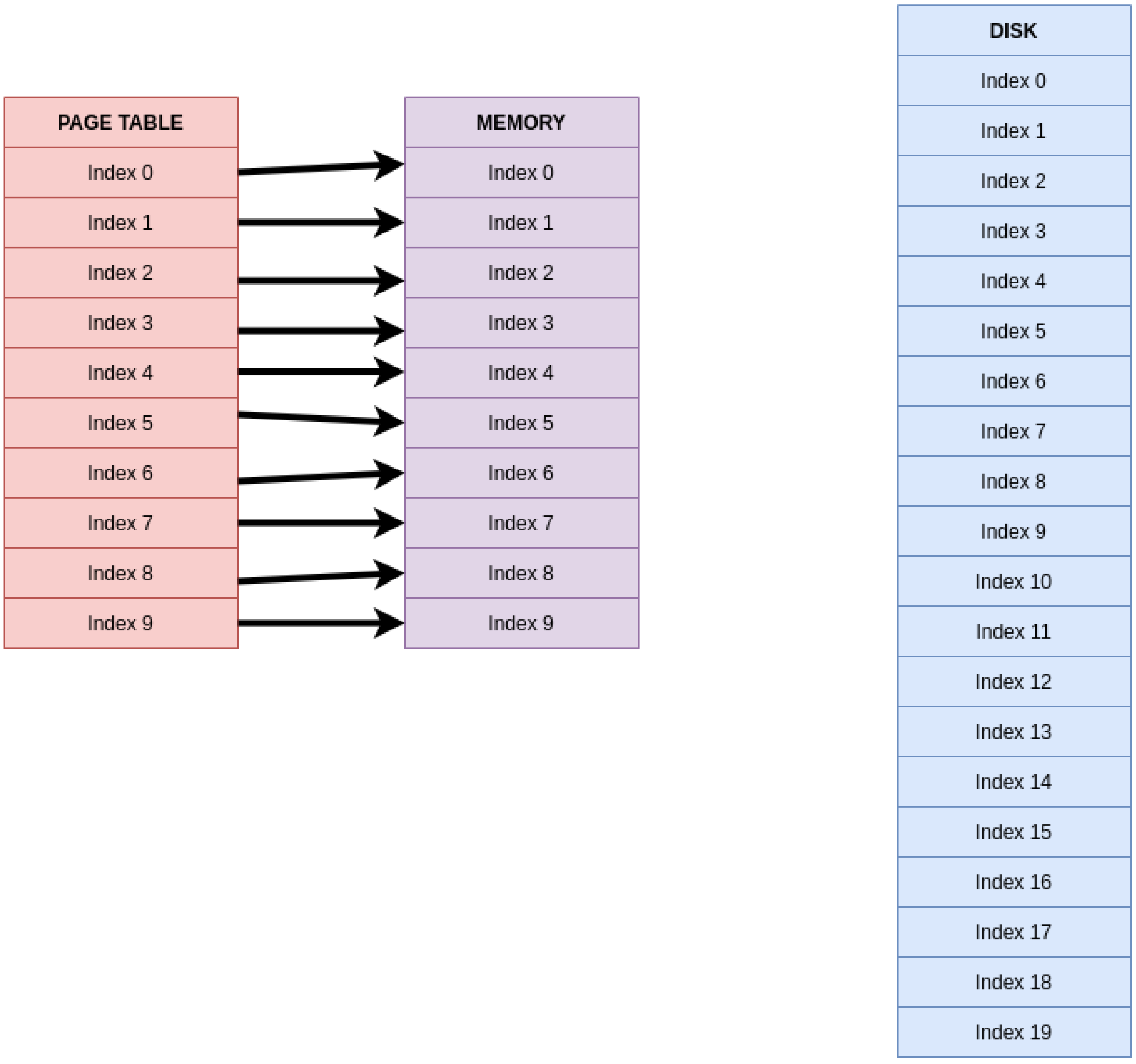
Index: It holds the index of the array disk which this page table entry keeps. For example page table entry holds 3 as an index for arrDisk1[3].

Referenced: It holds the information that if this page table entry is referenced.

Modified: It holds the information that if this page table entry is modified.

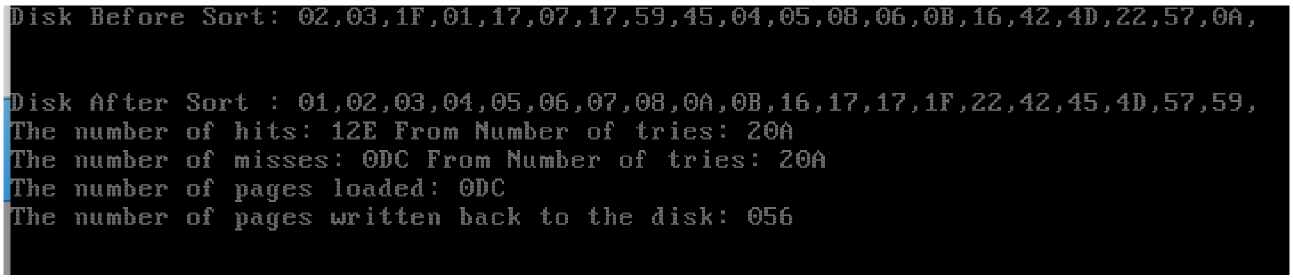
Valid: It holds the information that if this page table entry is valid.

VISUALS FOR THE MEMORY MANAGEMENT



SCREENSHOTS:

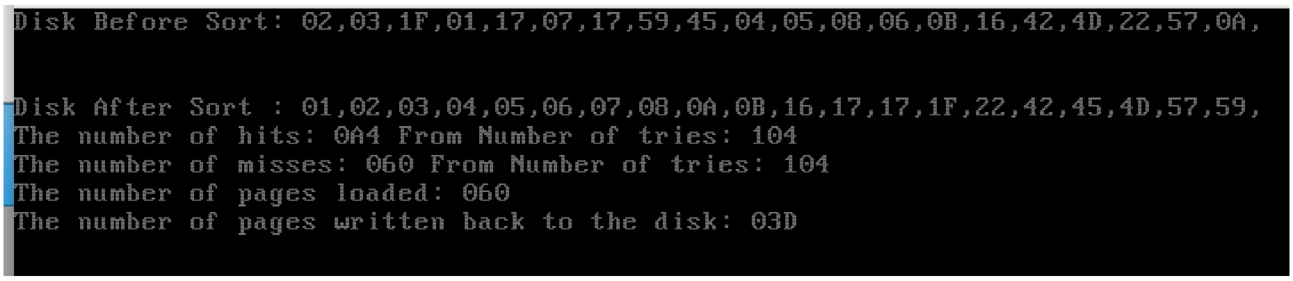
Bubble Sort With FIFO and Not Sorted Array 1



```
Disk Before Sort: 02,03,1F,01,17,07,17,59,45,04,05,08,06,0B,16,42,4D,22,57,0A,


Disk After Sort : 01,02,03,04,05,06,07,08,0A,0B,16,17,17,1F,22,42,45,4D,57,59,
The number of hits: 12E From Number of tries: 20A
The number of misses: 0DC From Number of tries: 20A
The number of pages loaded: 0DC
The number of pages written back to the disk: 056
```

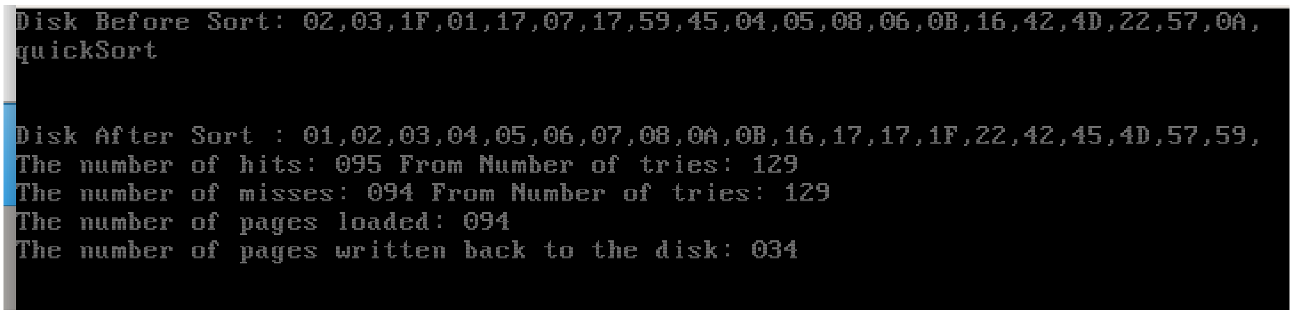
Insertion Sort With Second Chance and Not Sorted Array 1



```
Disk Before Sort: 02,03,1F,01,17,07,17,59,45,04,05,08,06,0B,16,42,4D,22,57,0A,


Disk After Sort : 01,02,03,04,05,06,07,08,0A,0B,16,17,17,1F,22,42,45,4D,57,59,
The number of hits: 0A4 From Number of tries: 104
The number of misses: 060 From Number of tries: 104
The number of pages loaded: 060
The number of pages written back to the disk: 03D
```

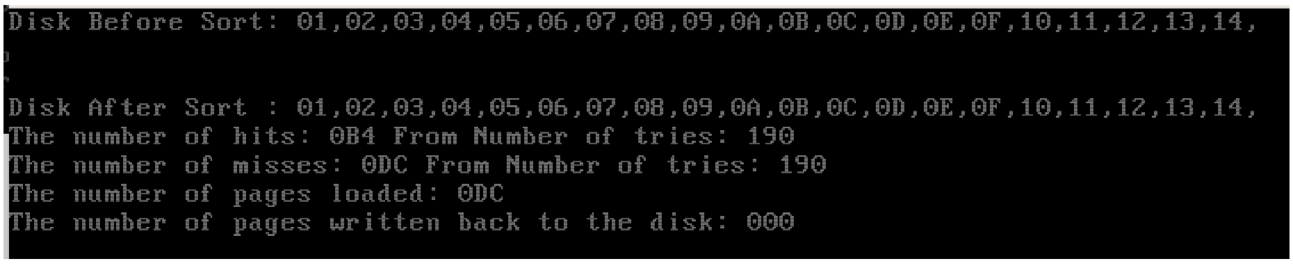
Quick Sort With LRU and Not Sorted Array 1



```
Disk Before Sort: 02,03,1F,01,17,07,17,59,45,04,05,08,06,0B,16,42,4D,22,57,0A,
quickSort


Disk After Sort : 01,02,03,04,05,06,07,08,0A,0B,16,17,17,1F,22,42,45,4D,57,59,
The number of hits: 095 From Number of tries: 129
The number of misses: 094 From Number of tries: 129
The number of pages loaded: 094
The number of pages written back to the disk: 034
```

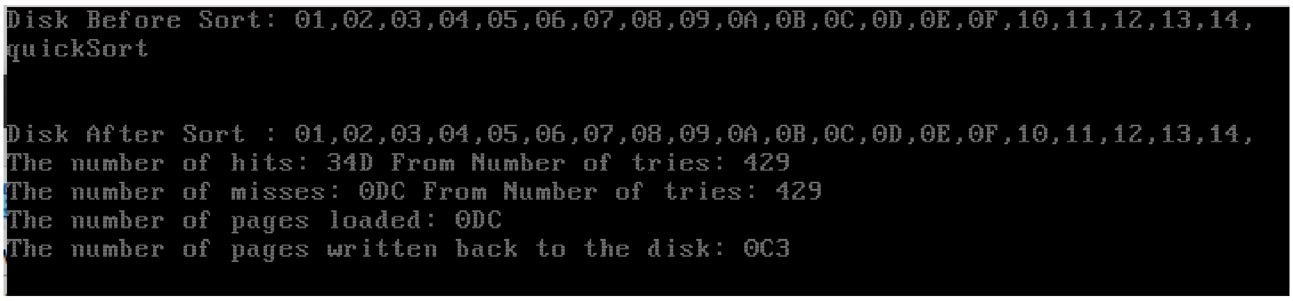
Bubble Sort With Second Chance and Sorted Array



```
Disk Before Sort: 01,02,03,04,05,06,07,08,09,0A,0B,0C,0D,0E,0F,10,11,12,13,14,

Disk After Sort : 01,02,03,04,05,06,07,08,09,0A,0B,0C,0D,0E,0F,10,11,12,13,14,
The number of hits: 0B4 From Number of tries: 190
The number of misses: 0DC From Number of tries: 190
The number of pages loaded: 0DC
The number of pages written back to the disk: 000
```

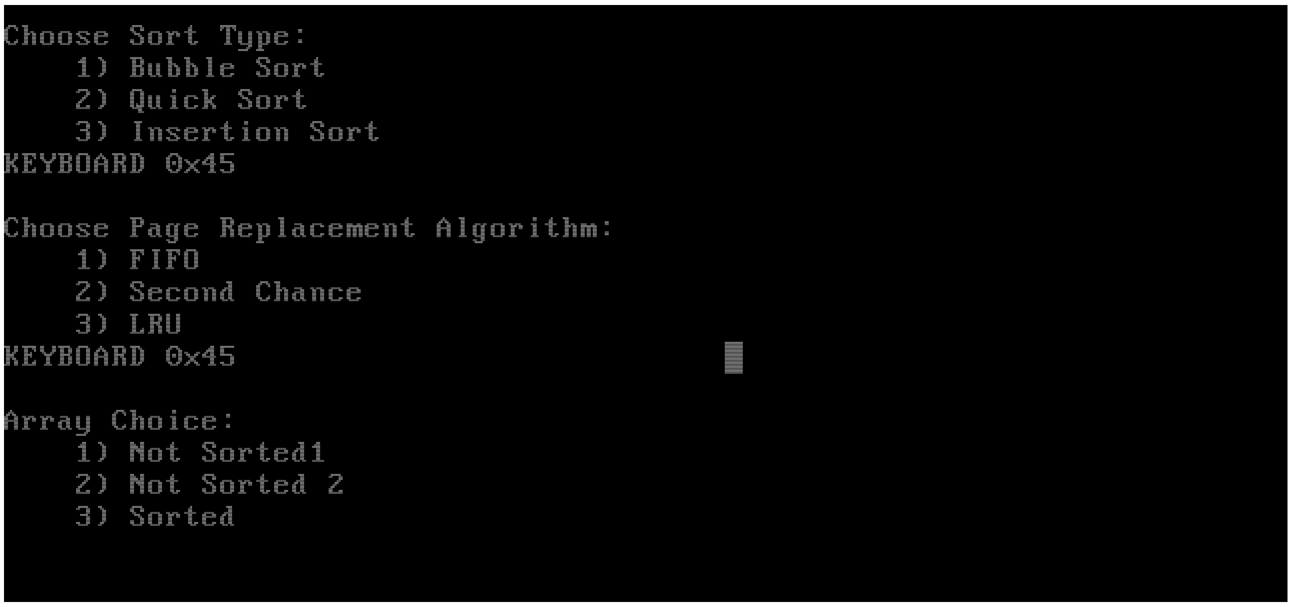
Quick Sort With FIFO and Sorted Array



```
Disk Before Sort: 01,02,03,04,05,06,07,08,09,0A,0B,0C,0D,0E,0F,10,11,12,13,14,
quickSort


Disk After Sort : 01,02,03,04,05,06,07,08,09,0A,0B,0C,0D,0E,0F,10,11,12,13,14,
The number of hits: 34D From Number of tries: 429
The number of misses: 0DC From Number of tries: 429
The number of pages loaded: 0DC
The number of pages written back to the disk: 0C3
```

User Interface



```
Choose Sort Type:
    1) Bubble Sort
    2) Quick Sort
    3) Insertion Sort
KEYBOARD 0x45

Choose Page Replacement Algorithm:
    1) FIFO
    2) Second Chance
    3) LRU
KEYBOARD 0x45

Array Choice:
    1) Not Sorted1
    2) Not Sorted 2
    3) Sorted
```