GEBZE TECHNICAL UNIVERSITY

COMPUTER ENGINEERING

OPERATING SYSTEM LECTURE

HOMEWORK 2

REPORT 2

AHMET FURKAN KURBAN

1801042674

JUNE 19

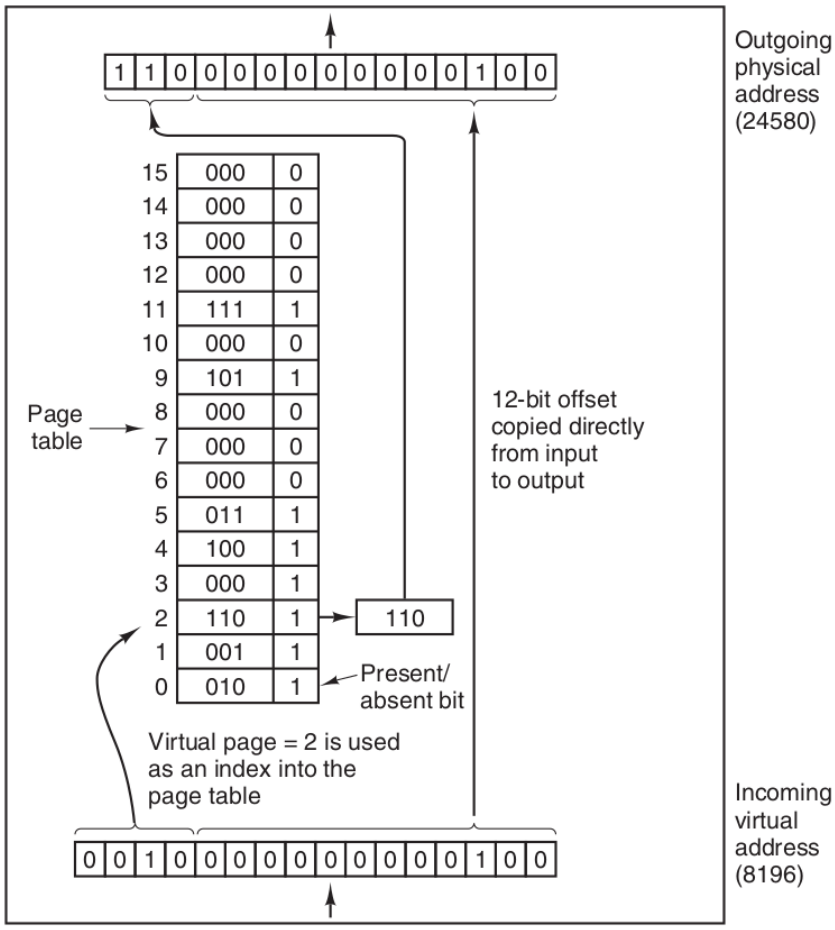
MAIN STRUCTURE :

Virtual Memory :

The basic idea behind virtual memory is that each program has its own address space, which is broken up into chunks called pages. Each page is a contiguous range of addresses. These pages are mapped onto physical memory, but not all pages have to be in physical memory at the same time to run the program.

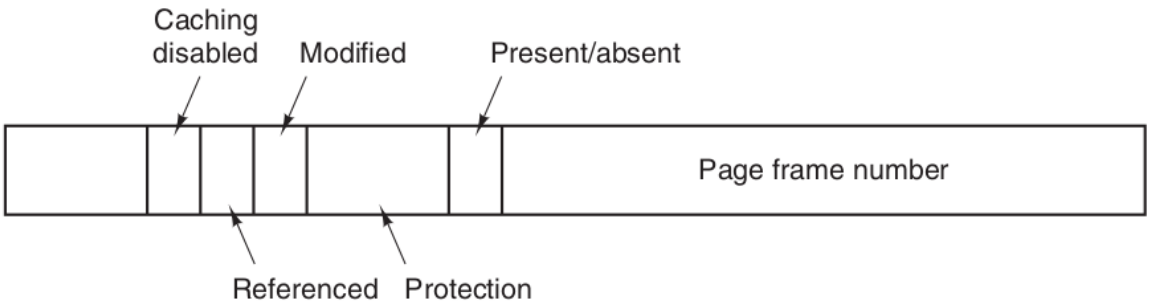
Page Tables:

The mapping of virtual addresses onto physical addresses can be summarized as follows: the virtual address is split into a virtual page number (high-order bits) and an offset (low-order bits).



Structure of a Page Table Entry:

The most important field is the Page frame number. After all, the goal of the page mapping is to output this value. Next to it we have the Present/absent bit. If this bit is 1, the entry is valid and can be used. If it is 0, the virtual page to which the entry belongs is not currently in memory. Accessing a page table entry with this bit set to 0 causes a page fault.



When a page fault occurs, the operating system has to choose a page to evict (remove from memory) to make room for the incoming page. If the page to be removed has been modified while in memory, it must be rewritten to the disk to bring the disk copy up to date. If, however, the page has not been changed the disk copy is already up to date, so no rewrite is needed.

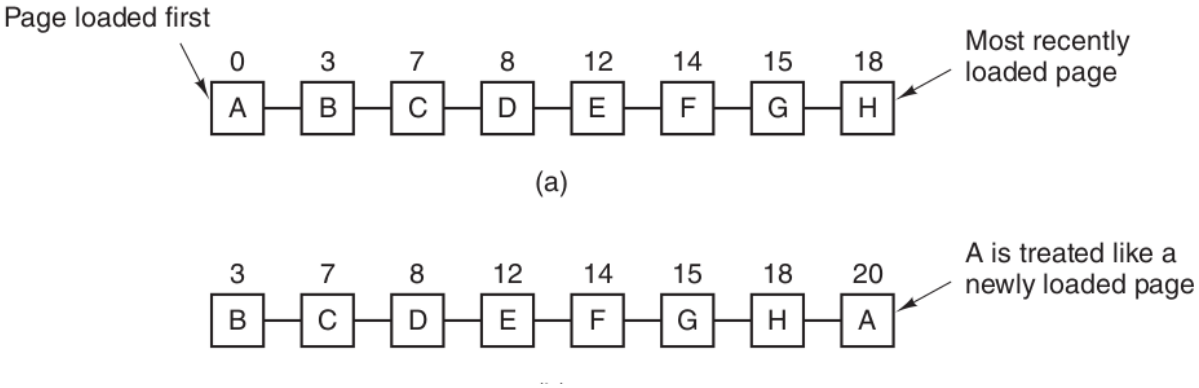
The page to be read in just overwrites the page being evicted.

1-FIFO :

FIFO (First-In, First-Out) algorithm. To illustrate how this works, consider a supermarket that has enough shelves to display exactly k different products. One day, some company introduces a new convenience food instant, freeze-dried, organic yogurt that can be reconstituted in a microwave oven. It is an immediate success, so our finite supermarket has to get rid of one old product in order to stock it.

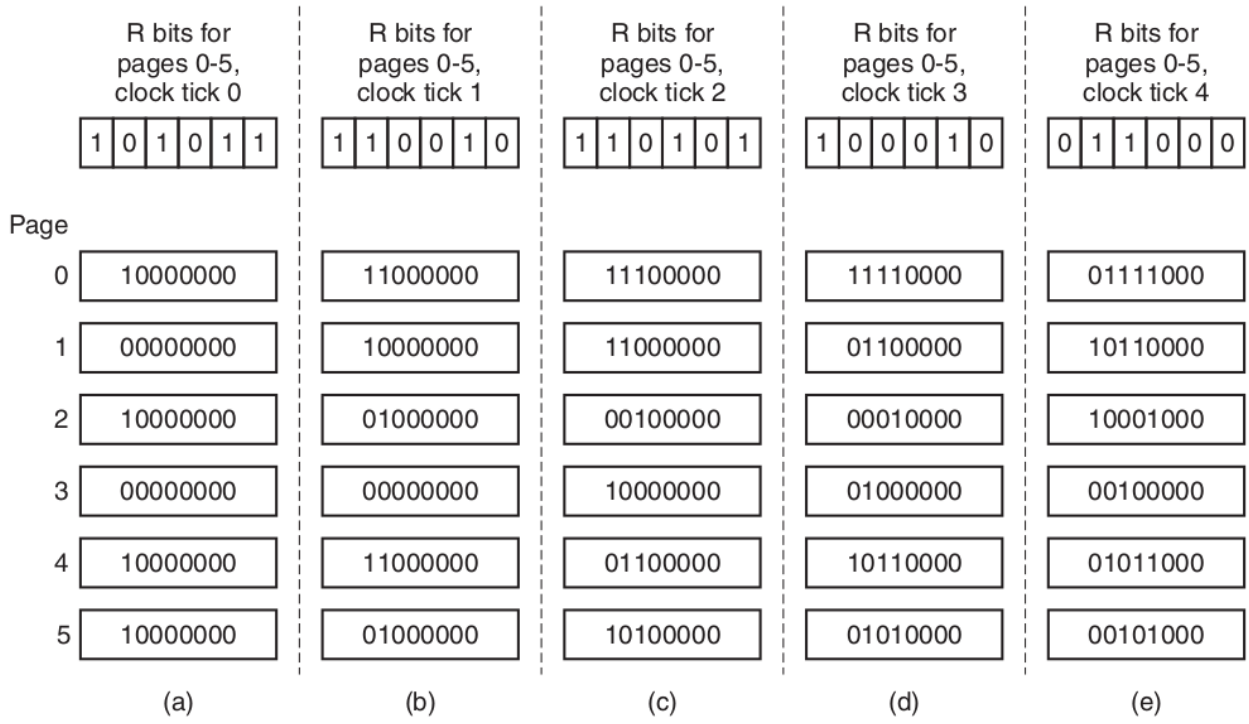
2-SECOND CHANCE :

A simple modification to FIFO that avoids the problem of throwing out a heavily used page is to inspect the R bit of the oldest page. If it is 0, the page is both old and unused, so it is replaced immediately. If the R bit is 1, the bit is cleared, the page is put onto the end of the list of pages, and its load time is updated as though it had just arrived in memory. Then the search continues. Implemented using queue structure.



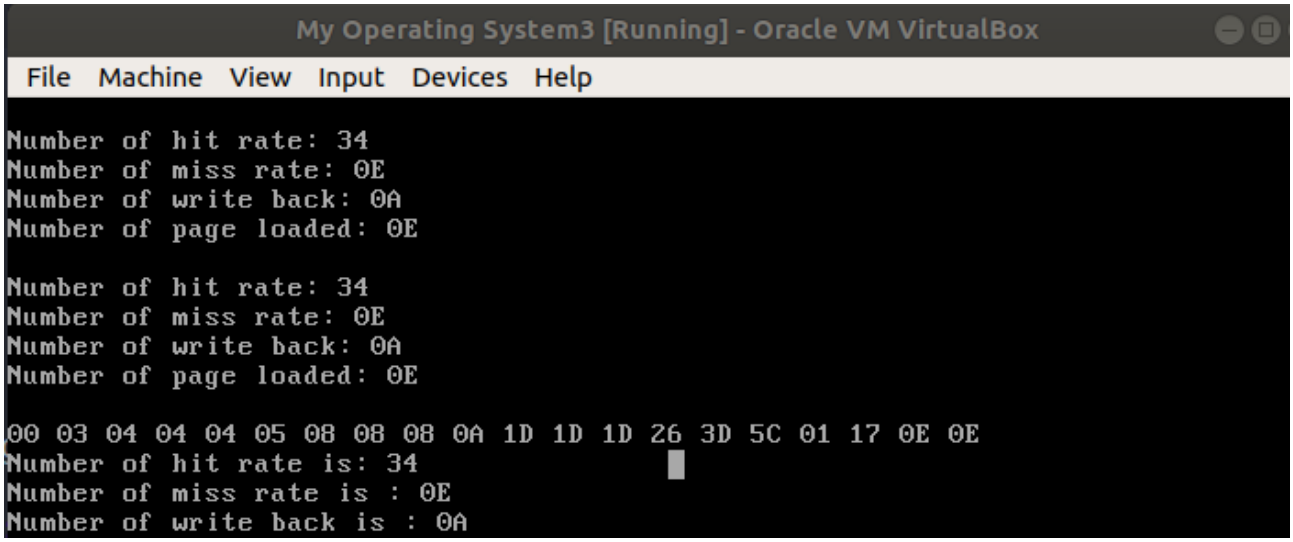
3- LRU :

A good approximation to the optimal algorithm is based on the observation that pages that have been heavily used in the last few instructions will probably be heavily used again soon. Conversely, pages that have not been used for ages will probably remain unused for a long time. This idea suggests a realizable algorithm: when a page fault occurs, throw out the page that has been unused for the longest time. This strategy is called LRU (Least Recently Used) paging.

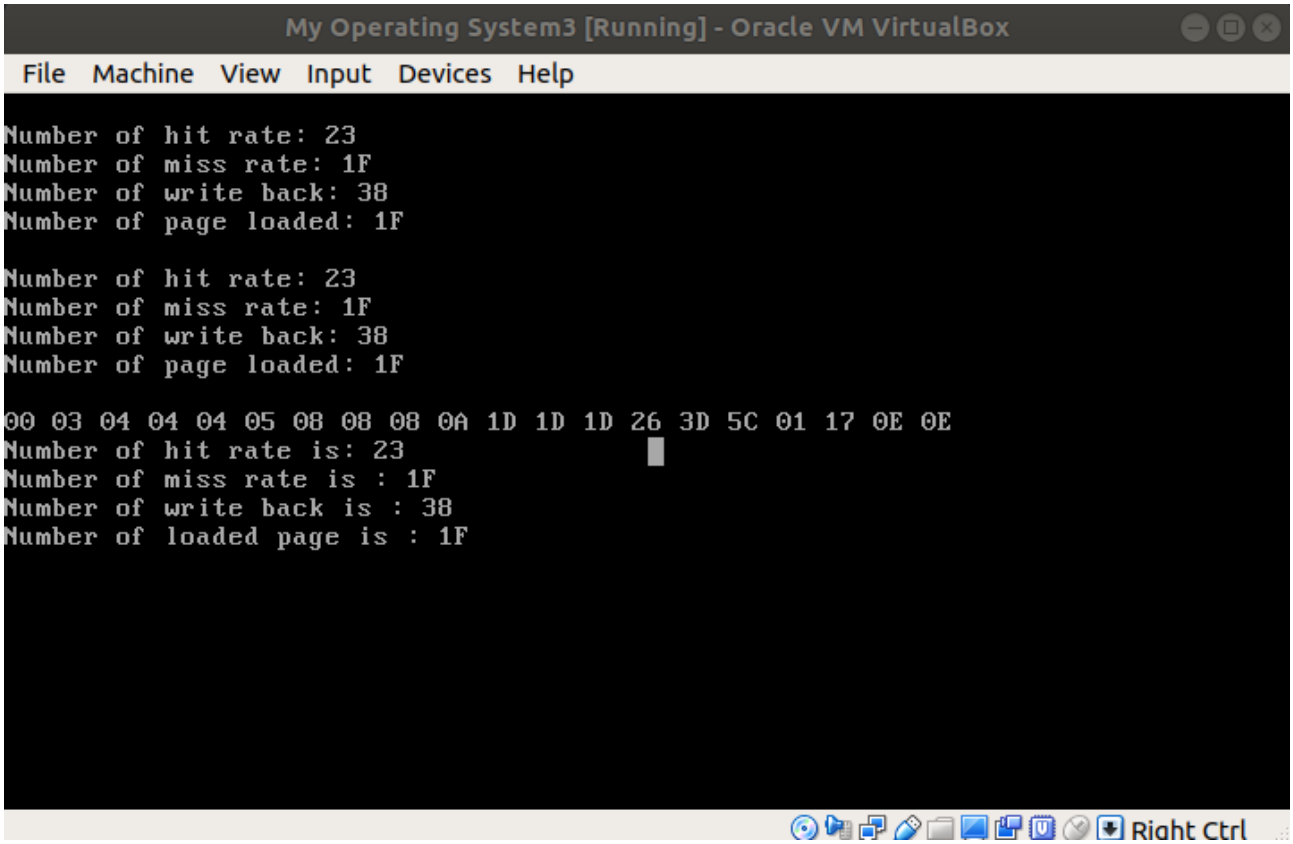


TEST RESULTS :

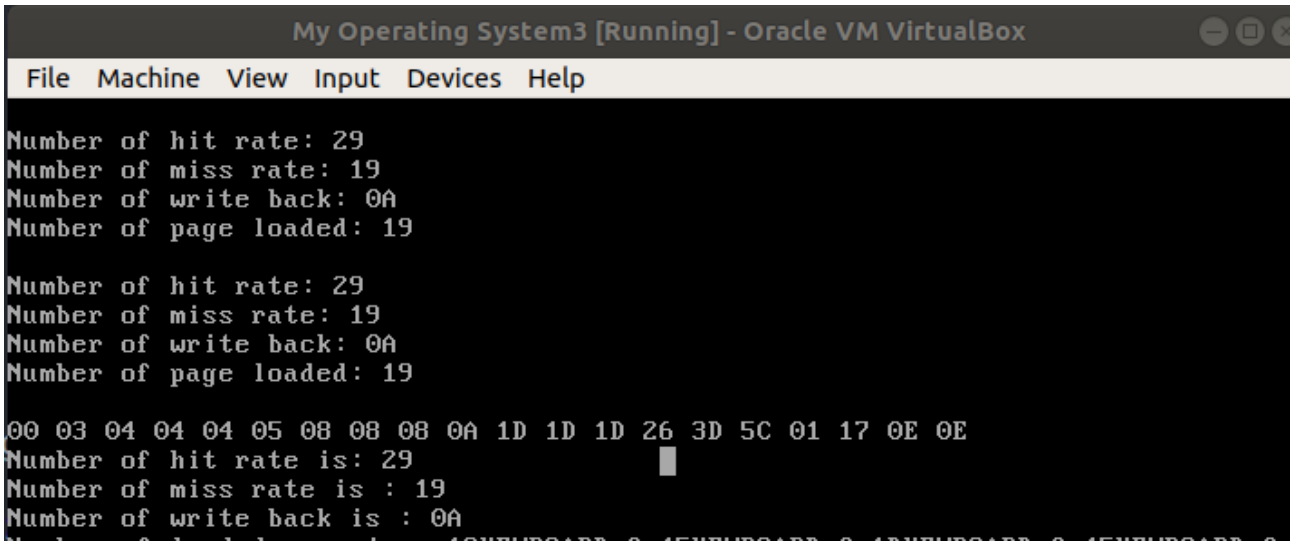
BUBBLE SORT ALGORITHM WITH FIFO:



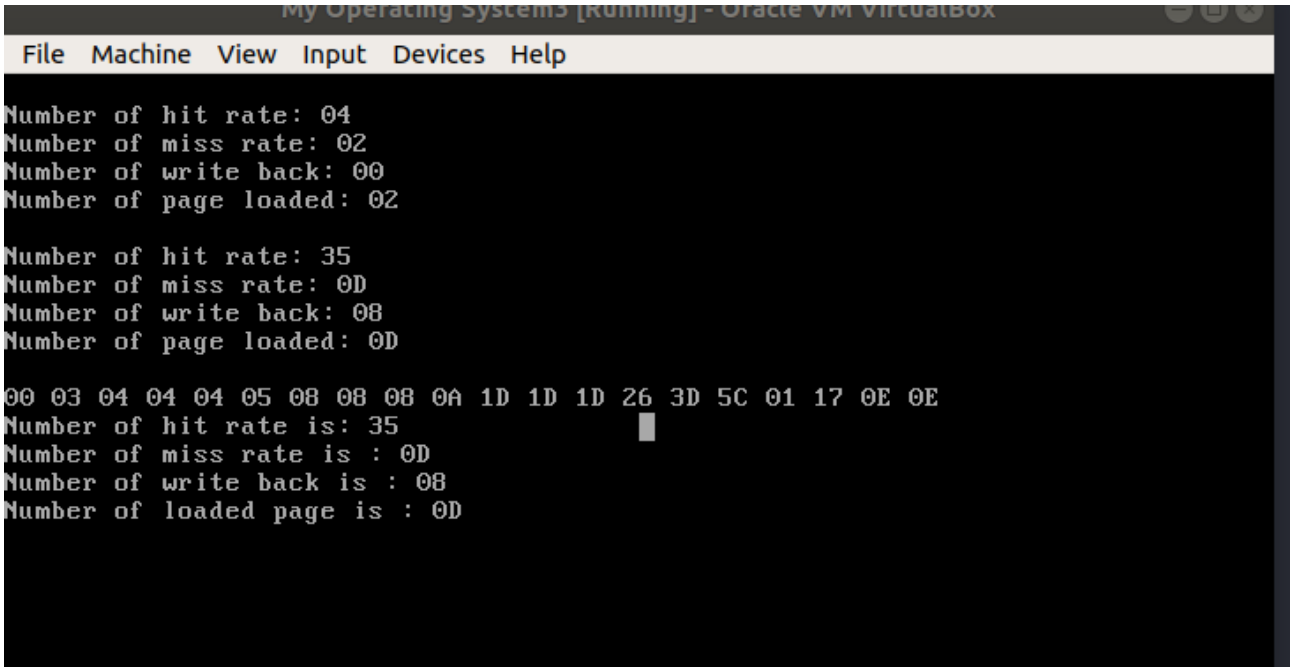
BUBBLE SORT ALGORITHM WITH SECOND CHANCE:



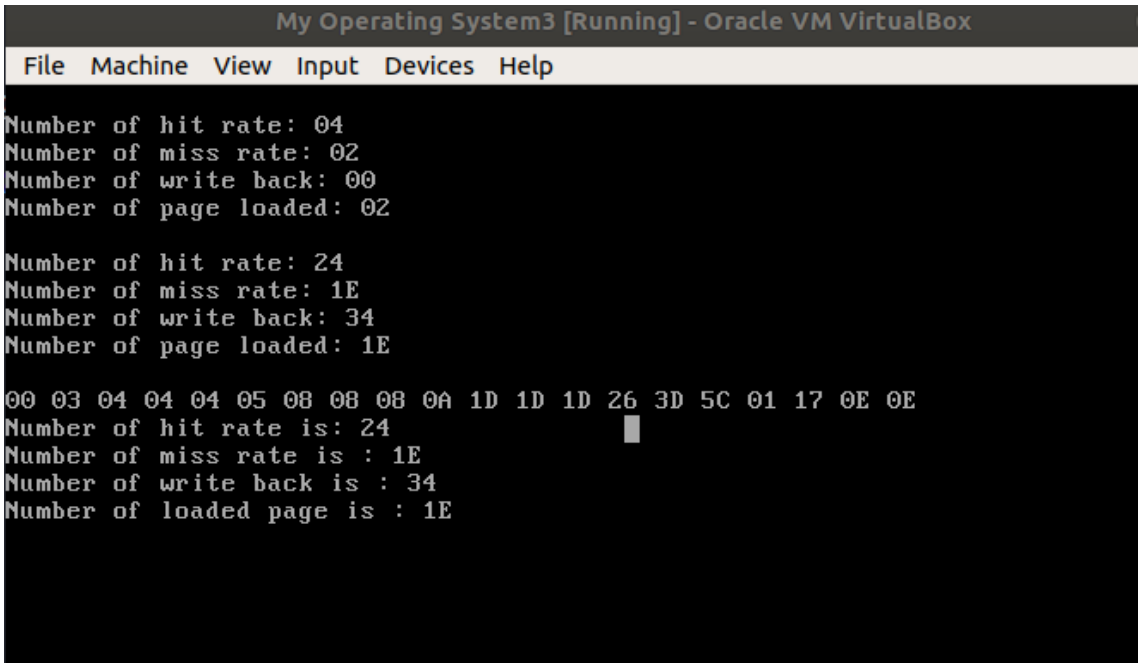
BUBBLE SORT ALGORITHM WITH LRU:



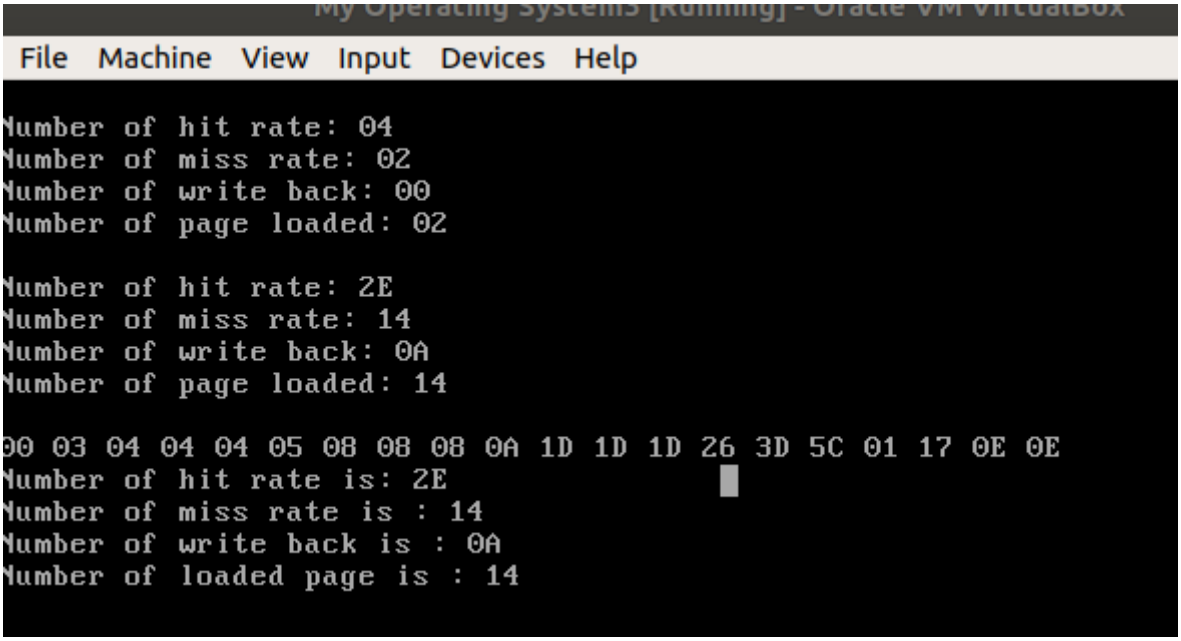
INSERTION SORT ALGORITHM WITH FIFO:



INSERTION SORT ALGORITHM WITH SECOND CHANCE:



INSERTION SORT ALGORITHM WITH LRU:



6- QUICK SORT :NOT IMPLEMENTED.