Batch: D2 **Roll No.: 16010122323**

Experiment / assignment / tutorial No. 6

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of the Staff In-charge with date

TITLE: Implementation of LRU Page Replacement Algorithm.

AIM: The LRU algorithm replaces the least recently used that is the last accessed memory block from user.

Expected OUTCOME of Experiment: (Mention CO/CO's attained here)

Books/ Journals/ Websites referred:

1. Carl Hamacher, Zvonko Vranesic and Safwat Zaky, "Computer Organization", Fifth Edition, TataMcGraw-Hill.

2. William Stallings, "Computer Organization and Architecture: Designing for Performance", Eighth Edition, Pearson.

Pre Lab/ Prior Concepts:

It follows a simple logic, while replacing it will replace that page which has least recently used out of all.

a) A hit is said to be occurred when a memory location requested is already in the cache.

b) When cache is not full, the number of blocks is added.

c) When cache is full, the block is replaced which is recently used

Algorithm:

1. Start
2. Get input as memory block to be added to cache
3. Consider an element of the array
4. If cache is not full, add element to the cache array
5. If cache is full, check if element is already present
6. If it is hit is incremented
7. If not, element is added to cache removing least recently used element
8. Repeat step 3 to 7 for remaining elements
9. Display the cache at very instance of step 8
10. Print hit ratio

11. End

Example:

```c
#include <stdio.h>
#include <limits.h>

int checkHit(int incomingPage, int queue[], int occupied) {
    for (int i = 0; i < occupied; i++) {
        if (incomingPage == queue[i])
            return 1; // Page hit
    }
    return 0; // Page miss
}

void printFrames(int queue[], int occupied) {
    for (int i = 0; i < occupied; i++)
        printf("%d\t\t\t", queue[i]);
}

int main() {

    printf("Name: Vedansh Savla\n");
    printf("Roll Number: 16010122323\nDivision: D2\n");
    printf("---------------------------------------\n");
    printf("COA exp 6: Implementation of LRU Page Replacement Algorithm.\n");
    printf("Implementation details:\n");
    printf("---------------------------------------\n");
    printf("LRU Page Replacement Algorithm :\n");

    int n;

        int frames;
    printf("Enter the number of frames: ");
    scanf("%d", &frames);

    printf("Enter the number of pages: ");
    scanf("%d", &n);

    int incomingStream[n];
    printf("Enter the page references:\n");
    for (int i = 0; i < n; i++) {
        scanf("%d", &incomingStream[i]);
    }

    int queue[frames];
    int distance[frames];
    int occupied = 0;
    int pagefault = 0;
```

```
    int hits = 0; // Counter for page hits
    int misses = 0; // Counter for page misses

    printf("Page\t Frame1 \t Frame2 \t Frame3\n");

    for (int i = 0; i < n; i++) {
        printf("%d:  \t\t", incomingStream[i]);

        if (checkHit(incomingStream[i], queue, occupied)) {
            printFrames(queue, occupied);
            hits++;
        } else if (occupied < frames) {
            queue[occupied] = incomingStream[i];
            pagefault++;
            occupied++;
            printFrames(queue, occupied);
            misses++;
        } else {
            int max = INT_MIN;
            int index;

            for (int j = 0; j < frames; j++) {
                distance[j] = 0;
                for (int k = i - 1; k >= 0; k--) {
                    ++distance[j];
                    if (queue[j] == incomingStream[k])
                        break;
                }

                if (distance[j] > max) {
                    max = distance[j];
                    index = j;
                }
            }
            queue[index] = incomingStream[i];
            printFrames(queue, occupied);
            pagefault++;
            misses++;
        }
        printf("\n");
    }

    printf("Page Faults: %d\n", pagefault);
    printf("Hits: %d\n", hits);
    printf("Misses: %d\n", misses);
    float hitRatio = (float)hits / (float)(hits + misses);
    printf("Hit Ratio: %.2f\n", hitRatio * 100);
```

```
    return 0;
}
```

OUTPUT :

```
/tmp/6KN3twByON.o
Name: Vedansh Savla
Roll Number: 16010122323
Division: D2
----------------------------------------
COA exp 6: Implementation of LRU Page Replacement Algorithm.
Implementation details:
----------------------------------------
LRU Page Replacement Algorithm :
Enter the number of frames: 3
Enter the number of pages: 5
Enter the page references:
4
2
1
4
5
Page     Frame1      Frame2      Frame3
4:          4
2:          4           2
1:          4           2           1
4:          4           2           1
5:          4           5           1
Page Faults: 4
Hits: 1
Misses: 4
Hit Ratio: 20.00
```

Post Lab Descriptive Questions

1. Define hit rate and miss ratio?

Hit Rate : The chief measurement of a cache, which is the percentage of all accesses that are satisfied by the data in the cache. Also known as "hit ratio." See cache and hits. 1.

Miss ratio : A miss ratio is the flip side of this where the cache misses are calculated and compared with the total number of content requests that were received.

2. What is the need for virtual memory?

Virtual memory serves two purposes. First, it allows us to extend the use of physical memory by using disk. Second, it allows us to have memory protection, because each virtual address is translated to a physical address . Less number of I/O would be needed to load or swap each user program into memory.

Conclusion : Implementation of LRU Page Replacement Algorithm method is understood.

Date: Signature of faculty in-charge