MICRO-PROJECT REPORT

**“Least Recently Used (LRU) Page Replacement Algorithm’’**

Under the subject

Operating System [22516]

Submitted by

|  |  |
| --- | --- |
| **Sr No.** | **Student name** |
| 1. | Shruti Vishnu Gatkal. |

Under guidance of

**Mrs. S.B. Patil**

Department of Computer Engineering

Government Polytechnic, Karad.

MICRO- PROJECT REPORT

**“Implement Least Recently Used (LRU) Page Replacement Algorithm”**

1. **Rationale**

In operating systems page replacement is used for memory management, page replacement algorithms are needed to decide which page needed to be replaced when new page comes in. Whenever a new page is referred and not present in memory, page fault occurs and Operating System replaces one of the existing pages with newly needed page. Different page replacement algorithms suggest different ways to decide which page to replace. The target for all algorithms is to reduce number of page faults.

In **L**east **R**ecently **U**sed (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. In this project we implement Least Recently Used page replacement algorithm using C Language.

1. **Aim of Micro-project**
2. To learn page replacement concept in operating system.
3. To understand different page replacement algorithms.
4. To understand working of Least Recently Used (LRU) algorithm.

1. **Course outcome Achieved**
2. Apply page replacement algorithm.
3. Calculate page faults from the given page references.
4. **Literature Review**
5. [www.wikipedia.com/operating](http://www.wikipedia.com/operating) : Page replacement, local and global replacement, Belady’s anomaly.
6. Operating System by Prashant B. Jawalkar: LRU Algorithm theory, Advantages and Limitations of LRU.
7. [www.geeksforgeeks.com](http://www.geeksforgeeks.com) : Referred for implementation ideas.

1. **Actual Procedure Followed** 
   1. **Page Replacement in Operating System:**

In [operating system](https://en.wikipedia.org/wiki/Operating_system) [paging](https://en.wikipedia.org/wiki/Paging) using for [virtual memory](https://en.wikipedia.org/wiki/Virtual_memory) [management](https://en.wikipedia.org/wiki/Memory_management), page replacement algorithms decide which memory pages to page out, sometimes called swap out, or write to disk, when a [page](https://en.wikipedia.org/wiki/Page_(computer_memory)) of memory needs to be allocated. [Page replacement](https://en.wikipedia.org/wiki/Paging) happens when a requested page is not in memory and a free page cannot be used to satisfy the allocation, either because there are none, or because the number of free pages is lower than some threshold.

When the page that was selected for replacement and paged out is referenced again it has to be paged in (read in from disk), and this involves waiting for I/O completion. This determines the quality of the page replacement algorithm. The less waiting time for page, better the algorithm. A page replacement algorithm looks at the limited information about accesses to the pages provided by hardware, and tries to guess which pages should be replaced to minimize the total number of page misses, while balancing this with the costs (primary storage and processor time) of the algorithm itself.

* Terminologies related to Page Replacement:

1. Page Fault: When the page (data) requested by the program is not available in the main memory then it is need to replace where page faults occurred.
2. Page Hit: When the page requested by the program is available in the main memory it is called as page hit.
3. Local Replacement: The local page replacement selects page that belongs from same process for replacement.
4. Global Replacement: In global replacement processor is free to select any page from memory which belongs to any process.
5. Belady’s anomaly: In operating system, Belady’s anomaly is the phenomenon in which increasing the number of page frames results in an increase in the number of [page faults](https://en.wikipedia.org/wiki/Page_fault) for certain memory access patterns.

* Different Algorithms used for Page Replacement:

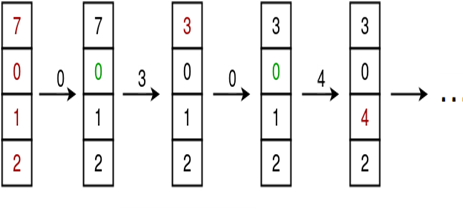
1. FIFO (First In First Out) Page Replacement Algorithm.
2. Optimal Page Replacement Algorithm.
3. Least Recently Used (LRU) Page Replacement Algorithm.
   1. **Least Recently Used (LRU) Page Replacement Algorithm:**

In **L**east **R**ecently **U**sed (LRU) algorithm is a Greedy algorithm where the page to be replaced is least recently used. The idea is based on locality of reference.

If we use the recent past as an approximation of the near future then we would replace that page which has not been used for longest period of time. This is a least recently used algorithm.

* For Example:

1. Let consider the page reference string is 7 0 1 2 0 3 0 4 2 3 0 3 2.
2. Initially we have 4 page slots or frames.
3. Initially all slots are empty, so when 7 0 1 2 are allocated to the empty slots —>**4 Page fault.**
4. 0 is already present there, so —> **0 Page fault.**
5. When 3 come it will take the place of 7 because it is least recently used —>**1 Page fault**
6. 0 is already in memory, so **—> 0 Page fault.**
7. 4 will takes place of 1 **—> 1 Page Fault**
8. Now for the further page reference string —>**0 Page fault** because they are already available in the memory.

****

Total number of page fault: 6

* Advantages of Least Recently Used Page Replacement Algorithm:

1. It is good approximation to optimal algorithm.
2. LRU is actually quite efficient.
3. LRU is very feasible to implement.

* Limitations of Least Recently Used Page Replacement Algorithm:
  1. LRU algorithms required additional data structure and hardware support for actual implementation.
  2. Actual implementation of LRU is critical.
  3. It suffers from Belady’s anomaly.
* Source Code:

#include<stdio.h>

int findLRU(int time[], int n)

{

int i, minimum = time[0], pos = 0;

for(i = 1; i < n; ++i)

{ if(time[i] < minimum)

{

minimum = time[i];

pos = i;

}

}

return pos;

}

void main()

{

int no\_of\_frames, no\_of\_pages, frames[10], pages[30], counter = 0, time[10], flag1, flag2, i, j, pos, faults = 0;

printf("Enter number of frames: ");

scanf("%d", &no\_of\_frames);

printf("Enter number of pages: ");

scanf("%d", &no\_of\_pages);

printf("Enter reference string: ");

for(i = 0; i < no\_of\_pages; ++i)

{

scanf("%d", &pages[i]);

}

for(i = 0; i < no\_of\_frames; ++i)

{

frames[i] = -1;

}

for(i = 0; i < no\_of\_pages; ++i)

{

flag1 = flag2 = 0;

for(j = 0; j < no\_of\_frames; ++j)

{

if(frames[j] == pages[i])

{ counter++;

time[j] = counter;

flag1 = flag2 = 1;

break;

}

}

if(flag1 == 0)

{

for(j = 0; j < no\_of\_frames; ++j)

{

if(frames[j] == -1)

{ counter++;

faults++;

frames[j] = pages[i];

time[j] = counter;

flag2 = 1;

break;

}

}

}

if(flag2 == 0)

{

pos = findLRU(time, no\_of\_frames);

counter++;

faults++;

frames[pos] = pages[i];

time[pos] = counter;

}

printf("\n");

for(j = 0; j < no\_of\_frames; ++j)

{

printf("%d\t", frames[j]);

}

}

printf("\n\nTotal Page Faults = %d", faults);

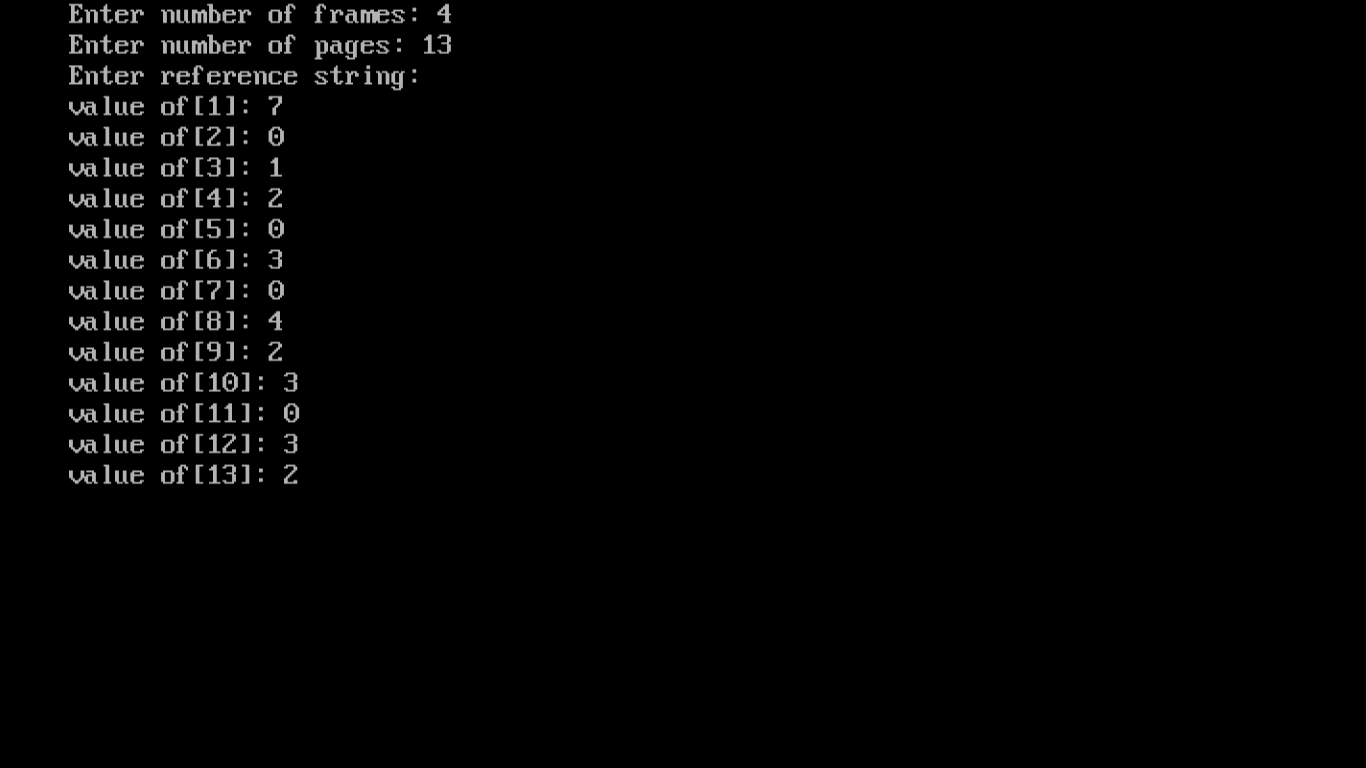
getch();

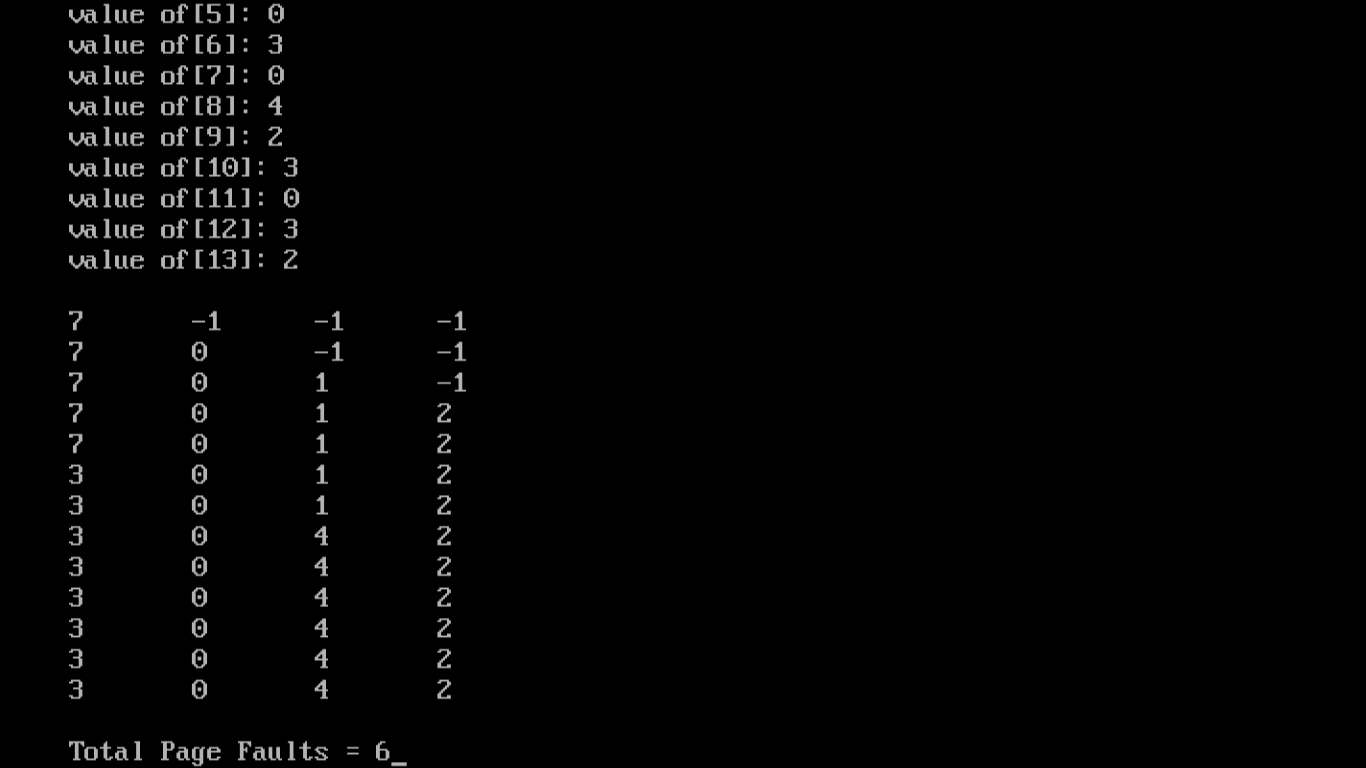
}

1. **Actual resources Used**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr.No | Name of Resource | Specification | Qty | Remarks |
| 1. | System | Intel (R) Core( TM ) i36006 U CPU @ 2.00GHz 2.00. | 1 |  |
| 2. | Software | Turbo C | 1 |  |
| 3. | Other Reference | Book-Operating System by Prashant B. Jawalkar.  [www.tutorialspoint.com/operatingsystem](http://www.tutorialspoint.com/operatingsystem)  [www.wikipedia.com](http://www.wikipedia.com) | 1 |  |

1. **Output of the Micro-Project**

****

****

1. **Skill Developed /learning out of this Micro-Project**

We studied page replacement concept of operating system. In which, we saw different algorithms for page replacement. By comparing all algorithms we select Least Recently Used (LRU) algorithm for implementation. By understanding its working, we implement this algorithm in C Programming language and also examine its execution along with looping statement. We also learned different terminologies in page replacement such as page fault, page hit. We also analyse advantages and limitation of LRU algorithm by comparing this algorithm with other algorithms.

1. **Application of the Micro-Project**
2. LRU Page Replacement algorithm used in operating system for main memory management.
3. This project is used to understand page replacement concept along with Least Recently Used Algorithm in operating system.

**\*\*\*\*\*\***