

**WIA2004 OPERATING SYSTEM**

**LAB 8 PROJECT REPORT**

**PAGE REPLACEMENT**

**GROUP MEMBERS**

|  |  |  |
| --- | --- | --- |
| **No.** | **Member Name** | **Matric Number** |
| **1.** | **MUHAMMAD IZZAT HAKEEM BIN ROSDI** | **17204856** |
| **2.** | **WAN MUHAMMAD AZIB BIN WAN ABDUL RAHIM** | **17207678** |
| **3.** | **MUHAMMAD AMIRUL AMIN BIN ARMAN** | **17202969** |
| **4.** | **JONE WAN GAN** | **S2031153** |
| **5.** | **LAWRENCE LEROY CHIENG TZE YAO** | **S2018935** |

**OBJECTIVE**

Write a C program to simulate page replacement algorithms

a) FIFO

**DESCRIPTION**

## **What are Replacement Algorithms?**

Generally, most operating systems use the method of paging for memory management. These algorithms are needed to decide of which page needs to be replaced when a new page comes into the picture or is demanded. The need for the demand occurs when the Operating Systems need any page for the processing which is not actually present in the main memory. This situation is also called Page Fault.

**Page Fault:** The page fault takes place when the main program accesses the memory page which is mapped into a virtual address space but is not loaded in physical memory.

When the Physical Memory is much smaller than the Virtual Memory in such a situation Page Fault happens.

## **What is FIFO Page Replacement Algorithm in C?**

FIFO which is also called First In First Out is one of the types of Replacement Algorithms. This algorithm is used in a situation where an Operating system replaces an existing page with the help of memory by bringing a new page from the secondary memory.

FIFO is the simplest among all algorithms which are responsible for maintaining all the pages in a queue for an operating system and keeping track of all the pages in a queue.

The older pages are kept in the front and the newer ones are kept at the end of the queue. Pages that are in the front are removed first and the pages which are demanded are added.

**CODE**

//Learnprogramo

#include <stdio.h>

int main()

{

int referenceString[10], pageFaults = 0, m, n, s, pages, frames;

printf("\nEnter the number of Pages:\t");

scanf("%d", &pages);

printf("\nEnter reference string values:\n");

for( m = 0; m < pages; m++)

{

printf("Value No. [%d]:\t", m + 1);

scanf("%d", &referenceString[m]);

}

printf("\n What are the total number of frames:\t");

{

scanf("%d", &frames);

}

int temp[frames];

for(m = 0; m < frames; m++)

{

temp[m] = -1;

}

for(m = 0; m < pages; m++)

{

s = 0;

for(n = 0; n < frames; n++)

{

if(referenceString[m] == temp[n])

{

s++;

pageFaults--;

}

}

pageFaults++;

if((pageFaults <= frames) && (s == 0))

{

temp[m] = referenceString[m];

}

else if(s == 0)

{

temp[(pageFaults - 1) % frames] = referenceString[m];

}

printf("\n");

for(n = 0; n < frames; n++)

{

printf("%d\t", temp[n]);

}

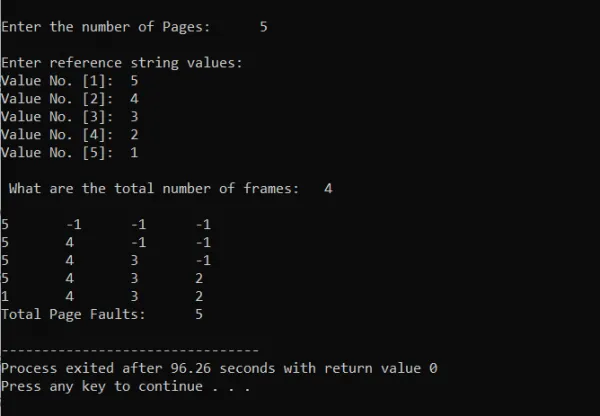
}

printf("\nTotal Page Faults:\t%d\n", pageFaults);

return 0;

}

**OUTPUT**

****