EXPERIMENT - 8 :PAGE REPLACEMENT ALGORITHM

Page Replacement Algorithms in Operating Systems (OS)

In an operating system, page replacement is referred to a scenario in which a page from the main memory should be replaced by a page from secondary memory. Page replacement occurs due to page faults. The various page replacement algorithms like FIFO, Optimal page replacement, LRU, LIFO, and Random page replacement help the operating system to decide which page to replace.

1).First in First out Page Replacement Algorithm

This is the first basic algorithm of Page Replacement Algorithms. This algorithm is basically dependent on the number of frames used. Then each frame takes up the certain page and tries to access it. When the frames are filled then the actual problem starts. The

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

int main() {
    // Get the number of pages
    int num_pages;
    printf("Enter the number of pages: ");
    scanf("%d", &num_pages);

    // Allocate memory for the page reference string
    int *page_reference = (int *) malloc(num_pages * sizeof(int));

    // Get the page reference string
    printf("Enter the page reference string:\n");
    for (int i = 0; i < num_pages; i++) {
        scanf("%d", &page_reference[i]);
    }

    // Get the memory capacity
    int capacity;
    printf("Enter the memory capacity: ");
    scanf("%d", &capacity);

    // Initialize an array to represent the page frames
    int *page_frames = (int *) malloc(capacity * sizeof(int));
    for (int i = 0; i < capacity; i++) {
        page_frames[i] = -1; // Set all page frames to -1 to indicate that they are empty
    }

    // Initialize a variable to keep track of the index of the next page frame to replace
    int replace_index = 0;
```

fixed number of frames is filled up with the help of first frames present. This concept is fulfilled with the help of Demand Paging After filling up of the frames, the next page in the waiting queue tries to enter the frame. If the frame is present then, no problem is occurred. Because of the page which is to be searched is already present in the allocated frames. If the page to be searched is found among the frames then, this process is known as Page Hit. If the page to be searched is not found among the frames then, this process is known as Page Fault. When Page Fault occurs this problem arises, then the First In First Out Page Replacement Algorithm comes into picture. The First In First Out (FIFO) Page Replacement Algorithm removes the Page in the frame which is allotted long back. This means the useless page which is in the frame for a longer time is removed and the new page which is in the ready queue and is ready to occupy the frame is allowed by the First In First Out Page Replacement.

```c
    int num_faults = 0;
  for (int i = 0; i < num_pages; i++) {
        int in_frame = 0;
        for (int j = 0; j < capacity; j++) {
            if (page_frames[j] == page_reference[i]) {
                in_frame = 1;
                break;}}
        if (!in_frame) {
            page_frames[replace_index] = page_reference[i];
            replace_index = (replace_index + 1) % capacity;
            num_faults++;
        }

        // Print the current state of the page frames
        printf("Page Frames: ");
        for (int j = 0; j < capacity; j++) {
            printf("%d ", page_frames[j]);
        }
        printf("\n");
    }

    // Print the total number of page faults
    printf("Total Page Faults: %d\n", num_faults);

    // Free the allocated memory
    free(page_reference);
    free(page_frames);

    return 0;
}
```

```
E/OS-LAB/OS-8/fifo ; exit;
Enter the number of pages: 7
Enter the page reference string:
4
423
346
2
3
35
235
Enter the memory capacity: 2
Page Frames: 4 -1
Page Frames: 4 423
Page Frames: 346 423
Page Frames: 346 2
Page Frames: 3 2
Page Frames: 3 35
Page Frames: 235 35
Total Page Faults: 7
```

2) . Least Recently Used (LRU) Replacement Algorithm

This is the Second basic algorithm of Page Replacement Algorithms. This algorithm is basically dependent on the number of frames used. Then each frame takes up a certain page and tries to access it. When the frames are filled then the actual problem starts. The fixed number of frames is filled up with the help of the first frames present. This concept is fulfilled with the help of Demand Paging After filling up the frames, the next page in the waiting queue tries to enter the frame. If the frame is present then, no problem has occurred. Because the page which is to be searched is already present in the allocated frames. If the page to be searched is found among the frames then, this process is known as Page Hit. If the page to be searched is not found among the frames then, this process is known as Page Fault. When Page Fault occurs this problem arises, then the Least Recently Used (LRU) Page Replacement Algorithm comes into the picture.

```c
#include<stdio.h>

void main(){
int n,i,j,cnt=0,x,val;
printf("enter the size of the pg table");
scanf("%d",&n);
int a[n];
int f=0,l;
printf("enter the size of demand paging list");
scanf("%d",&l);
int ent[l];
printf("enter the entries");
for(i=0;i<l;i++)
{
scanf("%d",&ent[i]); }
for(j=0;j<n;j++) {
a[i]=-1; }
for(i=0;i<l;i++) { f=0;
for(j=0;j<n;j++) {
if(a[j]==ent[i]) { val=a[j];
for(x=j;x<n-1;x++) {
a[x]=a[x+1]; }
a[x]=val; f=1; break;
}
} if(f==0) {
for(j=0;j<n-1;j++) {
a[j]=a[j+1]; }
a[j]=ent[i];
cnt++; }
else
continue;
}
printf("page fault is %d ",cnt); }
```

```
enter the size of the pg table5
enter the size of demand paging list 5
enter the entries 2
1
2
3
4
page fault is 2
```