PAGE REPLACEMENT ALGORITHM

BCSE303P Operating Systems

Name: Chillara V L N S Pavana Vamsi

Register Number: 21BCE5095

1. Develop page replacement algorithms (FIFO, Optimal, LRU) as a menu driven program and perform the replacements as per the menu choice selected by the user for the frame size -3. Assume your own frame reference string of size n and display the count of page fault

Code:

```
#include <stdio.h>
#include <stdbool.h>
#define s 3
//1 2 3 4 2 1 5 6 2 1 2 3 7 6 3 2 1 2 3 6
bool search(int a[], int n)
{
    for (int i = 0; i < s; i++)
    {
        if (a[i] == n)
        {
            return true;
        }
    }
    return false;
}
void pr_frame(int pg, int a[])
{
    printf("%d->frame:", pg);
    for (int i = 0; i < s; i++)
    {
        printf("%d ", a[i]);
    }
    printf("\n");
}
void fifo()
{
    printf("------------Implimenting FIFO Page replacement algorithm--------------\n");
    int n;
    printf("String Length:");
    scanf("%d", &n);
        printf("Enter page reference string:");
    int page[n];
    // 2 3 1 0 3 0 1 4 5 6
    // 2 3 1 7 3 7 1 4 5 6
    for (int i = 0; i < n; i++)
    {
        scanf("%d", &page[i]);
    }
    int frame[s];
    for (int i = 0; i < s; i++)
        frame[i] = -1;
    int hit = 0, fault = 0;
    for (int i = 0; i < n; i++)
    {
        if (search(frame, page[i]) == false) // if value dont exist
        {
            printf("Fault: ");
```

```
            int flag = 0;
            for (int i = 0; i < s; i++)
            {
                if (frame[i] != -1) // some value
                    flag++;
            }
            if (flag == s)
            {
                for (int i = 0; i < s - 1; i++)
                {
                    frame[i] = frame[i + 1];
                }
                frame[s - 1] = page[i];
                fault++;
            }
            else
            {
                frame[flag] = page[i];
                fault++;
            }
        }
        else
        {
            printf("Hit: ");
            hit++;
        }
        pr_frame(page[i], frame);
    }
    printf("\nFault:%d\n", fault);
    printf("Hit:%d\n", hit);
}
void lru()
{
    printf("-------------Implimenting LRU Page replacement algorithm--------------\n");
    int n;
    printf("String Length:");
    scanf("%d", &n);

    printf("Enter page reference string:");
    int page[n];
    for (int i = 0; i < n; i++)
    {
        scanf("%d", &page[i]);
    }
    int frame[s];
    for (int i = 0; i < s; i++)
        frame[i] = -1;
    int hit = 0, fault = 0;
    for (int i = 0; i < n; i++)
    {
        int flag = 0;
        for (int i = 0; i < s; i++)
```

```
        {
            if (frame[i] != -1) // some value
                flag++;
        }
        if (search(frame, page[i]) == false) // if value dont exist//fault
        {
            printf("Fault: ");

            if (flag == s)
            {
                for (int i = 0; i < s - 1; i++)
                {
                    frame[i] = frame[i + 1];
                }
                frame[s - 1] = page[i];
                fault++;
            }
            else
            {
                frame[flag] = page[i];
                fault++;
            }
        }
        else
        {
            printf("Hit: ");
            int pos;
            for (int j = 0; j < s; j++)
            {
                if (page[i] == frame[j])
                    pos = j;
            }

            int temp = frame[pos];
            if (flag == s)
            {

                for (int i = pos; i < s - 1; i++)
                {
                    frame[i] = frame[i + 1];
                }
                frame[s - 1] = temp;
            }
            else
            {
                frame[flag] = temp;
            }
            hit++;
        }
        pr_frame(page[i], frame);
    }
    printf("\nfault:%d\n", fault);
```

```
    printf("\nHit:%d", hit);
}
void optimal()
{
    printf("------------Implimenting Optimal Page replacement algorithm------------\n");
    int n;
    printf("String Length:");
    scanf("%d", &n);

    printf("Enter page reference string:");
    int page[n]; // = {2, 3, 1, 0, 3, 0, 1, 4, 5, 6};
    for (int i = 0; i < n; i++)
    {
        scanf("%d", &page[i]);
    }
    int frame[s];
    for (int i = 0; i < s; i++)
        frame[i] = -1;
    int hit = 0, fault = 0, k = 0;
    for (int i = 0; i < n; i++)
    {
        if (search(frame, page[i]) == false) // if value dont exist
        {
            printf("Fault: ");
            fault++;
            int flag = 0;
            for (int i = 0; i < s; i++)
            {
                if (frame[i] == -1) // some value
                    flag++;
            }
            if (flag <= s && flag != 0)
            {
                frame[k] = page[i];
                k++;
            }
            else
            {
                int flag2 = 0;
                int flag1 = 0;
                int max = -1, pos;
                for (int x = 0; x < s; x++) // frame
                {
                    flag1 = 0;
                    for (int y = i + 1; y < n; y++) // future string
                    {

                        if (page[y] == frame[x]) // fram
                        {
                            if (y > max)
                            {
                                max = y;
```

```
                                pos = x;
                            }
                            flag1 = 1;
                            break;
                        }
                    }
                    if (flag1 == 0) // not in future
                    {
                        flag2 = 1;
                        for (int i = x; i < s - 1; i++)
                        {
                            frame[i] = frame[i + 1];
                        }
                        frame[s - 1] = page[i];
                        break;
                    }
                }
                if (flag2 != 1)
                    frame[pos] = page[i];
            }
        }
        else
        {
            printf("Hit: ");
            hit++;
        }
        pr_frame(page[i], frame);
    }
    printf("\nFault:%d\n", fault);
    printf("Hit:%d\n", hit);
}
int main()
{
    printf("Menu:\n1.FIFO\n2.LRU\n3.Optimal\nEnter your choice:");
    int choice;
    scanf("%d", &choice);
    switch (choice)
    {
    case 1:
        fifo();
        break;
    case 2:
        lru();
        break;
    case 3:
        optimal();
        break;

    default:
        printf("Wrong Input!");
        break;
    }
```

```
    return 0;
}
```

Output:

```
Menu:
1.FIFO
2.LRU
3.Optimal
Enter your choice:1
---------------Implimenting FIFO Page replacement algorithm----------------
String Length:20
Enter page reference string:1 2 3 4 2 1 5 6 2 1 2 3 7 6 3 2 1 2 3 6
Fault: 1->frame:1 -1 -1
Fault: 2->frame:1 2 -1
Fault: 3->frame:1 2 3
Fault: 4->frame:2 3 4
Hit: 2->frame:2 3 4
Fault: 1->frame:3 4 1
Fault: 5->frame:4 1 5
Fault: 6->frame:1 5 6
Fault: 2->frame:5 6 2
Fault: 1->frame:6 2 1
Hit: 2->frame:6 2 1
Fault: 3->frame:2 1 3
Fault: 7->frame:1 3 7
Fault: 6->frame:3 7 6
Hit: 3->frame:3 7 6
Fault: 2->frame:7 6 2
Fault: 1->frame:6 2 1
Hit: 2->frame:6 2 1
Fault: 3->frame:2 1 3
Fault: 6->frame:1 3 6

Fault:16
Hit:4
```

```
Menu:
1.FIFO
2.LRU
3.Optimal
Enter your choice:2
---------------Implimenting LRU Page replacement algorithm---------------
String Length:20
Enter page reference string:1 2 3 4 2 1 5 6 2 1 2 3 7 6 3 2 1 2 3 6
Fault: 1->frame:1 -1 -1
Fault: 2->frame:1 2 -1
Fault: 3->frame:1 2 3
Fault: 4->frame:2 3 4
Hit: 2->frame:3 4 2
Fault: 1->frame:4 2 1
Fault: 5->frame:2 1 5
Fault: 6->frame:1 5 6
Fault: 2->frame:5 6 2
Fault: 1->frame:6 2 1
Hit: 2->frame:6 1 2
Fault: 3->frame:1 2 3
Fault: 7->frame:2 3 7
Fault: 6->frame:3 7 6
Hit: 3->frame:7 6 3
Fault: 2->frame:6 3 2
Fault: 1->frame:3 2 1
Hit: 2->frame:3 1 2
Hit: 3->frame:1 2 3
Fault: 6->frame:2 3 6

fault:15

Hit:5
```

```
Menu:
1.FIFO
2.LRU
3.Optimal
Enter your choice:3
---------------Implimenting Optimal Page replacement algorithm---------------
String Length:20
Enter page reference string:1 2 3 4 2 1 5 6 2 1 2 3 7 6 3 2 1 2 3 6
Fault: 1->frame:1 -1 -1
Fault: 2->frame:1 2 -1
Fault: 3->frame:1 2 3
Fault: 4->frame:1 2 4
Hit: 2->frame:1 2 4
Hit: 1->frame:1 2 4
Fault: 5->frame:1 2 5
Fault: 6->frame:1 2 6
Hit: 2->frame:1 2 6
Hit: 1->frame:1 2 6
Hit: 2->frame:1 2 6
Fault: 3->frame:3 2 6
Fault: 7->frame:3 7 6
Hit: 6->frame:3 7 6
Hit: 3->frame:3 7 6
Fault: 2->frame:3 6 2
Fault: 1->frame:3 1 2
Hit: 2->frame:3 1 2
Hit: 3->frame:3 1 2
Fault: 6->frame:1 2 6

Fault:11
Hit:9
```

2. Develop page replacement algorithms FIFO, LRU using two threads and let them return the page faults of those algorithms with the frame size 4. Assume your own frame reference string of size n.

```c
#include <stdio.h>
#include <stdbool.h>
#include <pthread.h>
#define s 4
#define n 20
struct details
{
    int frame[s];
    int page[n];
    int hit;
    int fault;
};// 1 2 3 4 2 1 5 6 2 1 2 3 7 6 3 2 1 2 3 6
bool search(int a[], int n1)
{
    for (int i = 0; i < s; i++)
    {
        if (a[i] == n1)
        {
            return true;
        }
    }
    return false;
}
void pr_frame(int pg, int a[])
{
    printf("%d->frame:", pg);
    for (int i = 0; i < s; i++)
    {
        printf("%d ", a[i]);
    }
    printf("\n");
}
void *fifo(void *a)
{
    struct details *fif = (struct details *)a;
    printf("------------Implimenting FIFO Page replacement algorithm-------------\n");
    fif->hit = 0;
    fif->fault = 0;
    for (int i = 0; i < n; i++)
    {
        if (search(fif->frame, fif->page[i]) == false) // if value dont exist
        {
            printf("Fault: ");
            int flag = 0;
            for (int i = 0; i < s; i++)
            {
                if (fif->frame[i] != -1) // some value
                    flag++;
```

```
            }
            if (flag == s)
            {
                for (int i = 0; i < s - 1; i++)
                {
                    fif->frame[i] = fif->frame[i + 1];
                }
                fif->frame[s - 1] = fif->page[i];
                fif->fault++;
            }
            else
            {
                fif->frame[flag] = fif->page[i];
                fif->fault++;
            }
        }
        else
        {
            printf("Hit: ");
            fif->hit++;
        }
        pr_frame(fif->page[i], fif->frame);
    }
}
void *lru(void *b)
{
    struct details *lr = (struct details *)b;
    printf("------------Implimenting LRU Page replacement algorithm--------------\n");
    lr->hit = 0, lr->fault = 0;
    for (int i = 0; i < n; i++)
    {
        int flag = 0;
        for (int i = 0; i < s; i++)
        {
            if (lr->frame[i] != -1) // some value
                flag++;
        }
        if (search(lr->frame, lr->page[i]) == false) // if value dont exist//fault
        {
            printf("Fault: ");
            if (flag == s)
            {
                for (int i = 0; i < s - 1; i++)
                {
                    lr->frame[i] = lr->frame[i + 1];
                }
                lr->frame[s - 1] = lr->page[i];
                lr->fault++;
            }
            else
            {
                lr->frame[flag] = lr->page[i];
```

```
                // printf("%d ",page[i]);
                lr->fault++;
            }
        }
        else
        {
            printf("Hit: ");
            int pos;
            for (int j = 0; j < s; j++)
            {
                if (lr->page[i] == lr->frame[j])
                    pos = j;
            }
            int temp = lr->frame[pos];
            if (flag == s)
            {

                for (int i = pos; i < s - 1; i++)
                {
                    lr->frame[i] = lr->frame[i + 1];
                }
                lr->frame[s - 1] = temp;
            }
            else
            {
                lr->frame[flag] = temp;
            }
            lr->hit++;
        }
        pr_frame(lr->page[i], lr->frame);
    }
}
int main()
{
    struct details fi, lr;
    printf("Enter page reference string:");
    for (int i = 0; i < n; i++)
    {
        scanf("%d", &fi.page[i]);
    }
    for (int i = 0; i < s; i++)
        fi.frame[i] = -1;
    lr = fi;
    pthread_t t1, t2;
    pthread_create(&t1, NULL, (void *)fifo, (void *)&fi);
    pthread_join(t1, NULL);
    printf("\nFIFO:\nFault:%d\nHit:%d\n", fi.fault, fi.hit);
    pthread_create(&t2, NULL, (void *)lru, (void *)&lr);
    pthread_join(t2, NULL);
    printf("\nLRU:\nFault:%d\nHit:%d\n", lr.fault, lr.hit);
    return 0;
}
```

```
Enter page reference string:1 2 3 4 2 1 5 6 2 1 2 3 7 6 3 2 1 2 3 6
---------------Implimenting FIFO Page replacement algorithm----------------
Fault: 1->frame:1 -1 -1 -1
Fault: 2->frame:1 2 -1 -1
Fault: 3->frame:1 2 3 -1
Fault: 4->frame:1 2 3 4
Hit: 2->frame:1 2 3 4
Hit: 1->frame:1 2 3 4
Fault: 5->frame:2 3 4 5
Fault: 6->frame:3 4 5 6
Fault: 2->frame:4 5 6 2
Fault: 1->frame:5 6 2 1
Hit: 2->frame:5 6 2 1
Hit: 3->frame:1 3 7 6
Fault: 2->frame:3 7 6 2
Fault: 1->frame:7 6 2 1
Hit: 2->frame:7 6 2 1
Fault: 3->frame:6 2 1 3
Hit: 6->frame:6 2 1 3

FIFO:
Fault:14
Hit:6
---------------Implimenting LRU Page replacement algorithm----------------
Fault: 1->frame:1 -1 -1 -1
Fault: 2->frame:1 2 -1 -1
Fault: 3->frame:1 2 3 -1
Fault: 4->frame:1 2 3 4
Hit: 2->frame:1 3 4 2
Hit: 1->frame:3 4 2 1
Fault: 5->frame:4 2 1 5
Fault: 6->frame:2 1 5 6
Hit: 2->frame:1 5 6 2
Hit: 1->frame:5 6 2 1
Hit: 2->frame:5 6 1 2
Fault: 3->frame:6 1 2 3
Fault: 7->frame:1 2 3 7
Fault: 6->frame:2 3 7 6
Hit: 3->frame:2 7 6 3
Hit: 2->frame:7 6 3 2
Fault: 1->frame:6 3 2 1
Hit: 2->frame:6 3 1 2
Hit: 3->frame:6 1 2 3
Hit: 6->frame:1 2 3 6

LRU:
Fault:10
Hit:10
```

3. Develop second chance page replacement algorithm with the frame size 4. Assume your own frame reference string of size n. Display the page fault and hit percentage

```c
#include <stdio.h>
#define s 4
// 0 4 1 4 2 4 3 4 2 4 0 4 1 4 2 4 3 4
int full = 0; // To check whether all frames are filled
int a[21];    // To take the input
int ref[s];   // This is for reference bits for each frame
int frame[s];
int repptr = 0; // Initialised to first frame
int count = 0;
int display()
{
    int i;
    printf("\nThe elements in the frame are: ");
    for (i = 0; i < full; i++)
        printf("%d  ", frame[i]);
}
int Pagerep(int ele)
{
    int temp;
    while (ref[repptr] != 0)
    {
        ref[repptr++] = 0;
        if (repptr == s)
            repptr = 0;
    }
    temp = frame[repptr];
    frame[repptr] = ele;
    ref[repptr] = 1; // The latest page reference, hence it is set to 1

    return temp;
}
int Pagefault(int ele)
{
    if (full != s)
    {
        ref[full] = 1; // All the ref bits are set to 1 as each frame is being filled
firstly
        frame[full++] = ele;
    }
    else
        printf("\tThe page replaced is %d", Pagerep(ele));
}
int Search(int ele)
{
    int i, flag;
    flag = 0;
    if (full != 0)
    {
        for (i = 0; i < full; i++)
            if (ele == frame[i])
```

```
            {
                flag = 1;
                ref[i] = 1; // When ever page reference occurs, it's rference bit is
set to 1
                break;
            }
    }
    return flag;
}
int main()
{
    int n, i;
    printf("The number of elements in the reference string are :");
    scanf("%d", &n);
    printf("Enter page refernce string:");
    for (i = 0; i < n; i++)
        scanf("%d", &a[i]);
    for (i = 0; i < n; i++)
    {
        if (Search(a[i]) != 1)
        {
            Pagefault(a[i]);
            display();
            count++;
        }
    }
    printf("\nThe number of page faults are %d\n", count);
    getchar();
    return 0;
}
```

```
The number of elements in the reference string are :18
Enter page refernce string:0 4 1 4 2 4 3 4 2 4 0 4 1 4 2 4 3 4

The elements in the frame are: 0
The elements in the frame are: 0  4
The elements in the frame are: 0  4  1
The elements in the frame are: 0  4  1  2       The page replaced is 0
The elements in the frame are: 3  4  1  2       The page replaced is 1
The elements in the frame are: 3  4  0  2       The page replaced is 3
The elements in the frame are: 1  4  0  2       The page replaced is 0
The elements in the frame are: 1  4  3  2
The number of page faults are 8
PS D:\VIT\Sem 2-2\OPERATING SYSTEMS\Assignments\Memory Management>
```

```
The number of elements in the reference string are :20
Enter page refernce string:1 2 3 4 2 1 5 6 2 1 2 3 7 6 3 2 1 2 3 6

The elements in the frame are: 1
The elements in the frame are: 1  2
The elements in the frame are: 1  2  3
The elements in the frame are: 1  2  3  4       The page replaced is 1
The elements in the frame are: 5  2  3  4       The page replaced is 2
The elements in the frame are: 5  6  3  4       The page replaced is 3
The elements in the frame are: 5  6  2  4       The page replaced is 4
The elements in the frame are: 5  6  2  1       The page replaced is 5
The elements in the frame are: 3  6  2  1       The page replaced is 6
The elements in the frame are: 3  7  2  1       The page replaced is 1
The elements in the frame are: 3  7  2  6       The page replaced is 7
The elements in the frame are: 3  1  2  6
The number of page faults are 12
PS D:\VIT\Sem 2-2\OPERATING SYSTEMS\Assignments\Memory Management>
```