# [도전과제] LRU 알고리즘 구현

20185309
황명원

## 1.코드

```
#include<stdio.h>
main()
{
    int p[50],c=0,c1,f,i,j,k=0,n,r,t,b[20],c2[20];
    int hit=0;
    int array2_num=0;

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

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

    int q[n][f]; // Frame 안에 들어있는 값을 순서대로 저장하기 위해 2 차원 배열로 저장
    int hit_array[n]; //hit 되면 hit_array 의 값을 1 로 변경, 나머지는 0


    for(i=0 ; i<sizeof(hit_array)/sizeof(int) ; i++)
        hit_array[i]=0;

    printf("Enter process:");
    for(i=0;i<n;i++)
        scanf("%d",&p[i]);

    for(i=0 ; i<n ; i++)
    {
        for(j=0 ; j<f ; j++)
            q[i][j]=0;
    } //frame 을 저장할 2 차원 배열을 다 0 으로 초기화



    q[0][k]=p[k];
    q[1][k]=q[0][k];//바뀌지 않은 새로운 값도 다음 배열에 저장
```

```
    c++; //페이지 폴트 수 세는 변수
    k++;
    for(i=1;i<n;i++)
    {
        c1=0;
        for(j=0;j<f;j++)
        {
            if(p[i]!=q[array2_num][j])
                c1++;
        }
        if(c1==f)
        {
            c++;
            if(k<f) // 프레임에 들어간 수가 3 보다 작을때 차례로
프레임수까지 넣는다
            {
                array2_num++;
                q[array2_num][k]=p[i];
                for(j=0 ; j<f ; j++)
                    q[array2_num+1][j]=q[array2_num][j];
                k++;
            }
            else //프레임에 들어간 수가 3 보다 크거나 같을때
             {
                for(r=0;r<f;r++)
                {
                    c2[r]=0;
                    for(j=i-1;j<n;j--)
                    {
                        if(q[array2_num][r]!=p[j])
                            c2[r]++;
                        else
                            break;
                    }
                }
                for(r=0;r<f;r++)
                    b[r]=c2[r];
                for(r=0;r<f;r++) //제일 오래된 페이지 찾기
                {
                    for(j=r;j<f;j++)
                    {
                        if(b[r]<b[j])
                        {
                            t=b[r];
                            b[r]=b[j];
                            b[j]=t;
                        }
                    }
```

```
                }
                 array2_num++;
                for(r=0;r<f;r++) //찾은 오래된 페이지를 없애고
그자리에 새로 넣기
                {
                    if(c2[r]==b[0])
                        q[array2_num][r]=p[i];
                    else
                        q[array2_num][r]=q[array2_num-1][r];
//바뀌지 않은 새로운 값도 다음 배열에 저장
                }
            }
        }
        else
        {
            array2_num++;
            for(r=0;r<f;r++)
            {
                q[array2_num][r]=q[array2_num-1][r];
            }
            hit++;
            hit_array[i]++;
        }
    }

    printf("Process  ");
    for(i=0 ; i<n ; i++)
        printf("%d  ",p[i]);
    printf("\n");

    for(i=0 ; i<f ; i++)
    {
        printf("Frame  %d",i);
        for(j=0 ; j<array2_num+1 ; j++)
        {
            if(q[j][i]==0)
                printf("  E");
            else
            printf("  %d",q[j][i]);
        }
        printf("\n");
    }
    for(i=0 ; i<f ; i++)
        printf("   ");
    for(i=0 ; i< sizeof(hit_array)/sizeof(int); i++)
    {
        printf("  %d",hit_array[i]);
    }
```

```
    printf("\nHit  %d",hit);
    printf("\nPage Fault %d",c);
}
```

2.파일 생성 및 실행파일 만들기

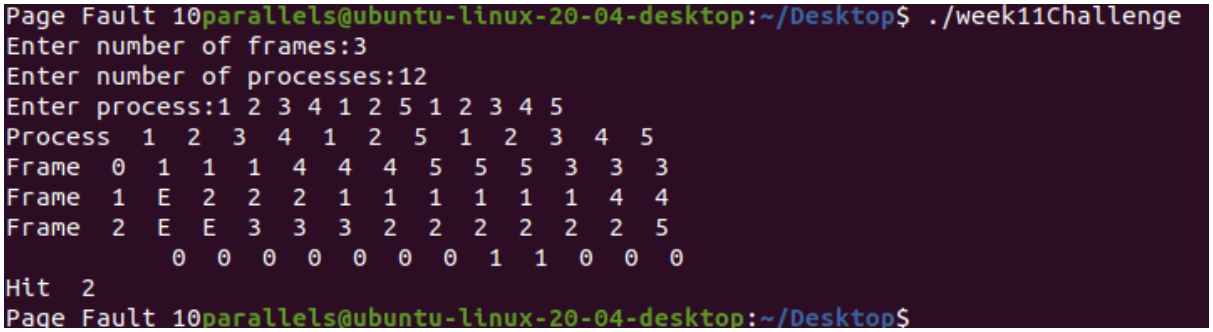
```
parallels@ubuntu-linux-20-04-desktop:~/Desktop$ vi week11Challenge.c
parallels@ubuntu-linux-20-04-desktop:~/Desktop$ gcc -o week11Challenge week11Cha
llenge.c
week11Challenge.c:2:1: warning: return type defaults to ‘int’ [-Wimplicit-int]
    2 | main()
      | ^~~~
```

3. 실행결과
(도전과제에 나온 예시만으로 결과를 확인하지 않고 기본과제에 나와있는 예시 레퍼런스 스트링으로도 결과를 확인해 봤습니다.)

Frame:3, process 개수:10,
입력하는 process: 7 5 9 4 3 7 9 6 2 1

```
parallels@ubuntu-linux-20-04-desktop:~/Desktop$ ./week11Challenge
Enter number of frames:3
Enter number of processes:10
Enter process:7 5 9 4 3 7 9 6 2 1
Process  7  5  9  4  3  7  9  6  2  1
Frame  0  7  7  7  4  4  4  9  9  9  1
Frame  1  E  5  5  5  3  3  3  6  6  6
Frame  2  E  E  9  9  9  7  7  7  2  2
          0  0  0  0  0  0  0  0  0  0
Hit  0
Page Fault 10parallels@ubuntu-linux-20-04-desktop:~/Desktop$
```

Frame:3,process 개수:12,
입력하는 process: 1 2 3 4 1 2 5 1 2 3 4 5




4.코드 설명
: 저는 기본과제의 코드를 충실히 이해한 바탕으로 도전과제를 해결하고 싶어서 도전과제의 코드를 이해하고 그것을 바탕으로 코드를 수정했습니다. (즉 LRU 알고리즘으로 구현 했고 인터넷 자료를 참조하지 않았습니다.)

출력결과처럼 코드를 짜고 싶어서 기본과제에서 풀었던 방식대로는 도전과제의 출력결과 처럼 출력을 할 수가 없었습니다. 처음부터 Frame 안에 어떤 레퍼런스 스트링이 들어오는지 과정을 출력해야 하기 때문에 저는 배열을 이용해 출력해보자 했습니다. 하지만 기본과제에서도 배열을 사용해서 출력을 했으니 이걸 한번에 다 담아서 출력 하려면 또 다른 배열을 사용해 저장해야 했었는데 저는 그걸 이차원 배열을 이용해 코드를 작성해 봤습니다.

기본과제 코드를 바탕으로 설명을 이어가자면 q 로 frame 안에 들어갈 수를 정했는데 저는 이 q 를

이차원배열로 바꾸었고 어디서 hit 이 됐는지를 저장하기 위해 hit_array 라는 배열을 만들었습니다.(이때 hit_array 는 단순히 0 과 1 로만 표현하므로 일차원으로 만들었습니다.) 처음에는 q[0][0]에 값이 들어왔다면 다음 Frame 에 들어오는 레퍼런스 스트링은 그 다음 차원인 q[1][]에서 나타내고 싶었습니다. 다음 배열에서도 똑같이 q[1][0]에 q[0][0]이 들어와야 q[1][]에서 문제없이 이어서 Frame 에 값이 들어오는 것을 순서대로 표현할 수 있어서 였습니다.

이제 이러한 과정을 page 개수만큼 실행 하도록 for 문안에서 코드를 수행시킵니다.(기본과제 랑 다르게 이차원 배열을 사용하므로 그것에 맞게 코드를 수정했습니다.)

이제 저는 이차원배열을 이용해 Frame 안에 레퍼런스 스트링이 들어오는 것을 도전과제의 출력결과와 동일하게 출력할 수 있었고 그 코드를 입력해 수행하였습니다.