

## 프로세스 동기화 과제

20185309

황명원

소스코드:

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

int cnt;
int flag[2] = {0,0};
int turn = 0;

void *func0() {
    for (int i = 0; i < 100000; i++) {
        flag[0] = 1;

        turn = 1;

        while (flag[1] && turn==1) {}
        cnt++;
        printf("cnt1 :: %d\n", cnt);
        flag[0] = 0;
    }
}

void *func1() {
    for (int i = 0; i < 100000; i++) {
        flag[1] = 1;

        turn = 0;

        while (flag[0]&&turn==0) {}
        cnt++;
        printf("cnt2 :: %d\n", cnt);
        flag[1] = 0;
    }
}

int main()
{
    pthread_t thread1, thread2;
    int iret1, iret2;

    iret1 = pthread_create( &thread1, NULL, func0, NULL);
    iret2 = pthread_create( &thread2, NULL, func1, NULL);

    pthread_join( thread1, NULL);
    pthread_join( thread2, NULL);

    printf("Thread error: %d\n", 200000-cnt);
    exit(0);
}
```

## 분석내용:

임계 영역에 진입하려면 먼저 `flag[i] = j`로 하여 임계 영역에 들어가게 합니다.

`turn = j`로 설정하고 내부 `while`문을 수행하고

내부 `while`문에서, 프로세스 `j`가 임계구역에 들어갈 의사표시를 하지 않았다면 임계 영역에 들어갈 수 있습니다.

만약, 두 개의 프로세스가 동시에 임계구역에 진입하려고 한다면 둘 중에 하나만을 선택해야 하므로, 변수 `turn`이 역할을 발휘하고 조금이라도 `turn`변수가 늦게 수행된 프로세스가 내부의 `while`문에서 기회를 양보합니다.

임계 영역에서 나오는 프로세스는 `flag[i]`를 `i`로 함으로써, 다른 프로세스가 임계 영역에 들어가도록 허용합니다.

이렇게 하게 되면 두개의 프로세스는 겹쳐서 실행이 되지 않아 문제에 의도한 대로 `cnt`를 증가시킬 수 있습니다.

## 실행결과:

```
hwangmyeong-won@hwangmyeong-won-ui-MacBookAir desktop % vi week7_1.c
hwangmyeong-won@hwangmyeong-won-ui-MacBookAir desktop % gcc -o week7_1 week7_1.c

week7_1.c:20:1: warning: non-void function does not return a value [-Wreturn-type]
}
^
week7_1.c:32:1: warning: non-void function does not return a value [-Wreturn-type]
}
^
2 warnings generated.
```

파일 생성

```
hwangmyeong-won@hwangmyeong-won-ui-MacBookAir desktop % ./week7_1
cnt1 :: 1
cnt2 :: 2
cnt1 :: 3
cnt2 :: 4
cnt1 :: 5
cnt2 :: 6
cnt1 :: 7
cnt2 :: 8
cnt1 :: 9
cnt2 :: 10
cnt1 :: 11
cnt2 :: 12
cnt1 :: 13
cnt2 :: 14
cnt1 :: 15
cnt2 :: 16
```

실행(1)

## (중략)

```
cnt1 :: 199979
cnt2 :: 199980
cnt1 :: 199981
cnt2 :: 199982
cnt1 :: 199983
cnt2 :: 199984
cnt1 :: 199985
cnt2 :: 199986
cnt1 :: 199987
cnt2 :: 199988
cnt1 :: 199989
cnt2 :: 199990
cnt1 :: 199991
cnt2 :: 199992
cnt1 :: 199993
cnt2 :: 199994
cnt1 :: 199995
cnt2 :: 199996
cnt1 :: 199997
cnt2 :: 199998
cnt1 :: 199999
cnt2 :: 200000
Thread error: 0
hwangmyeong-won@hwangmyeong-won-ui-MacBookAir desktop %
```

실행(2)

정상적으로 실행 됨을 확인 할수 있습니다.