**REPORT**

**| Embedded System**

**| MicroC/OS-II 스케줄러**

**| 컴퓨터정보공학과**

**| 12141540 박영창**

# 내용

3주차 실습 강의 시간에는 Microc/os II 스케줄러에 관해 배웠습니다.

* Microc/os II 스케줄러 : Ready Queue에 있는 task 중 우선순위가 가장 높은 즉 우선순위를 나타내는 수가 가장 낮은 task를 O(1)에 구하는 스케줄러
* Data structures
  + OSRdyGrp : 들어온 우선순위 값을 8로 나눈 값 (y) 위치의 bit를 1로 바꿔주며 유지하는 변수이다.
  + OSRdyTbl[] : 들어온 우선순위 값을 8로 나눈 값 (y) 의 인덱스에 우선순위 값을 8로 나눈 나머지 값 (x) 위치의 bit를 1로 바꿔주며 유지하는 변수이다.
  + OSMapTbl[] : 각 인덱스에 해당하는 위치의 bit가 1인 값을 저장해둔 배열 ( = BIT(X) )
  + OSUnMapTbl[] : OSRdyGrp, OSRdyTbl[]에 저장된 값들을 통해 우선순위에 맞는 인덱스를 찾을 수 있도록 저장해둔 배열
* 스케줄링 과정
  + Ready queue에 0~63 사이의 우선순위가 들어온다
  + OSRdyGrp 에 들어온 수를 8로 나눈 값을 or연산을 통해 bit를 1로 바꿔준다.
  + OsRdyTbl에 들어온 수를 8로 나눈 값의 인덱스에 들어온 수를 8로 나눈 나머지 값을 or연산을 통해 bit를 1로 바꿔준다.
  + OsRdyGrp이 나타내는 값을 인덱스로 OSUnMapTbl에서 값을 구한다.

y = OSUnMapTbl [ OsRdyGrp ]

* + 위에서 구한 값과 OsRdyTbl을 이용해 OSUnMapTbldptj 값을 구한다.

x = OSUnMapTbl [ OSRdyTbl [ y ] ]

* + 현재 가장 우선순위가 높은(수가 가장 낮은) 수를 찾는다.

결과 = x + y \* 8

# 과제 목표

Mandatory ( cmd == 1 )

* 1개의 태스크 생성
* 임의의 4개의 숫자 ( 0~63 ) 를 생성하고 그중 가장 작은 숫자를 스케줄러의 자료구조들을 사용해 구한다.
* final이라는 변수에 저장된 값보다 작은 숫자가 구해질 경우 색깔을 칠하고 final값을 갱신한다.
* final에 0이 대입될 때까지 반복하되 칠하는 숫자를 다음 순서대로 할 것

[ Red -> Blue -> Green -> Brown -> Red -> … ]

* 위 과정을 최소 3회 이상 반복한다

Optional ( cmd == 2 )

* Mandatory와 같지만 최소값이 아닌 최대값을 구한다
* final이라는 변수에 저장된 값보다 작은 숫자가 구해질 경우 똑같이 색깔을 칠하고 final값을 갱신한다.
* final에 63이 대입될 때까지 반복한다.

# 문제 해결

변수

* colorCount : 색깔 순서를 맞추기 위해 사용한 변수
* tmp : 임의의 난수 생성을 위해 사용한 변수
* FINAL : 과제 내용에서 설명된 값
* VAL : 들어온 4 개의 수 중에 가장 작은(또는 큰) 값
* zCount : final에 0(또는 63)이 대입이 되었을 때 증가시키는 값
* I : 반복문을 위해 사용
* pX : VAL값을 위해 구하는 x값
* pY : VAL값을 위해 구하는 y값
* disp\_x : 출력 위치의 x좌표
* disp\_y : 출력 위치의 y좌표
* colors[] : 출력할 색의 값을 저장해둔 배열
* myMapTbl[], myRdyTbl[], myRdyGrp : 스케줄링에 쓰일 자료구조
* myUnMapTbl[], myUnMapTbl\_max[] : 스케줄링에 쓰일 자료구조(각각 최소,최대를 찾음)

Mandatory

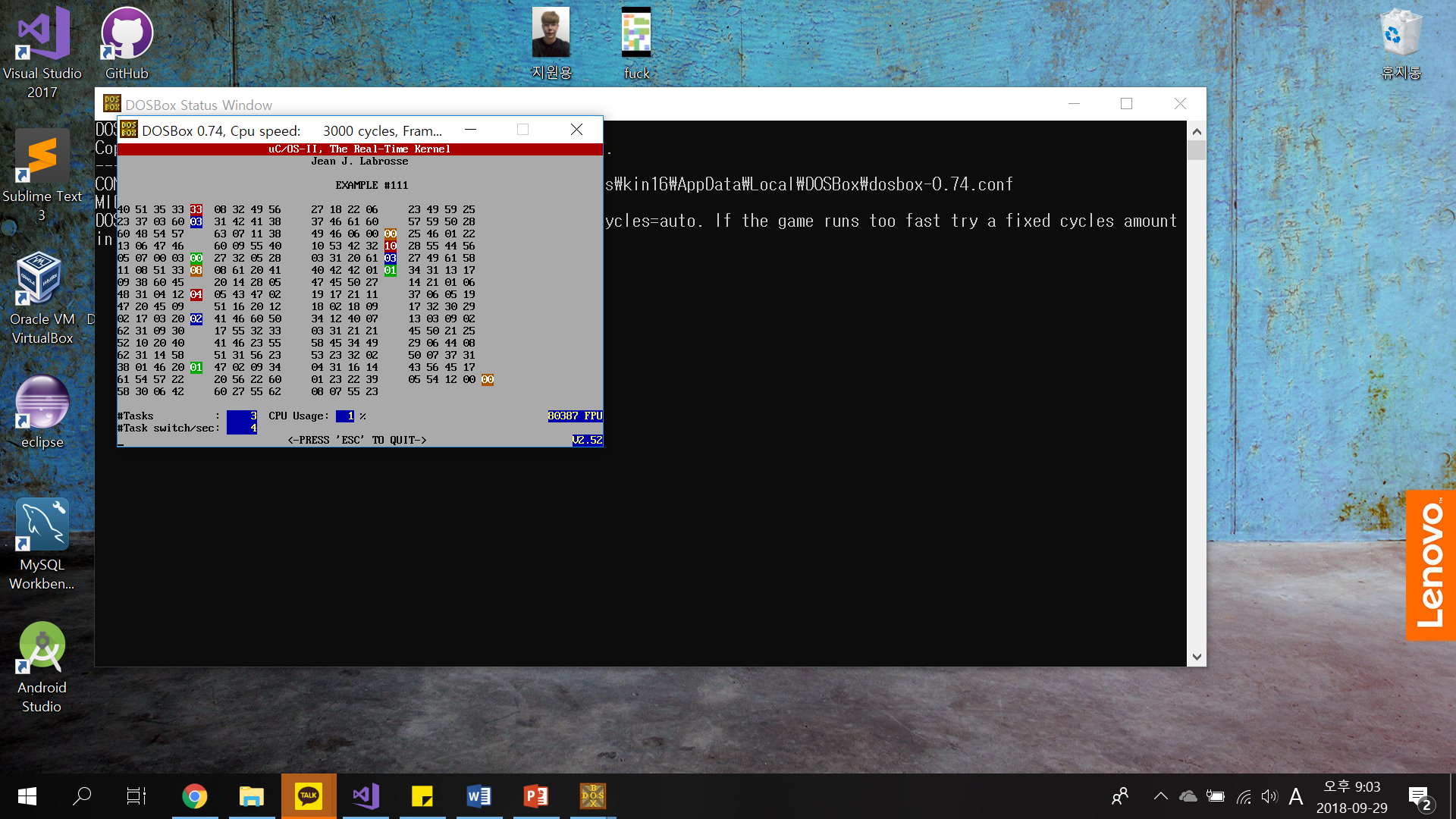
1. FINAL값을 63으로 초기화해준다.
2. myRdyGrp, myRdyTbl을 초기화한다.
3. 4개의 난수를 생성해 myRdyGrp, myRdyTbl에 값을 넣어준다.
4. 4개의 수 중의 최소값을 구해 VAL에 저장한다.
5. FINAL값보다 작은 값이 들어오면 갱신하고, 출력한다. 이때 colorCount를 증가시켜 색깔을 바꿔준다.
6. 만약 FINAL이 0으로 갱신되었다면, 출력 후에 다시 63으로 갱신하고 zCount를 증가시켜준다. 만약 zCount가 3이라면 task를 종료한다.
7. 2. ~ 6.을 반복한다.

Optional

1. FINAL 값을 0으로 초기화해준다.
2. myRdyGrp, myRdyTbl을 초기화한다.
3. 4개의 난수를 생성해 myRdyGrp, myRdyTbl에 값을 넣어준다.
4. 4개의 수 중의 최대값을 구해 VAL에 저장한다.
5. FINAL값보다 큰 값이 들어오면 갱신하고, 출력한다. 이때 colorCount를 증가시켜 색깔을 바꿔준다.
6. 만약 FINAL이 63으로 갱신되었다면, 출력 후에 다시 0으로 갱신하고 zCount를 증가시켜준다. 만약 zCount가 3이라면 task를 종료한다.
7. 2. ~ 6.을 반복한다.

# 결과

1. Mandatory



1. Optional

