7주차 실습 보고서

실습 주제

- Event flag, mailbox종합 실습

성공 실패 여부

- 성공

원인 분석

1. mailbox 생성 및 event flag생성

```
    for(i = 0; i < 3; i++) {
        numberMbox[i] = OSMboxCreate((void *)0);
    }
    colorMbox[0] = OSMboxCreate((void *)0);

    grp = OSFlagCreate(0x00, &err);
```

2. 각 random task에서 생성한 랜덤 숫자를 mailbox에 넣고, event flag를 통해 wait상태로 대기한다.

```
OSMboxPost(numberMbox[task_num], (void *)&rand_num);
OSFlagPend(grp, flags[task_num], OS_FLAG_WAIT_SET_ALL + OS_FL
```

3. calculate task에서 각각의 mailbox를 받는다.
   사실 random task 들의 우선순위가 calculate task보다 더 높기 때문에 이 부분에 메일박스로 굳이 동기화 할 필욘 없다.

```
for (i = 0; i < 3; i++) {
    num = *(int *)OSMboxPend(numberMbox[i], 0, &err);
    ...
```

4. 이후 calculate task에서 display task로 계산한 색깔 정보(char)을 mailbox를 통해 보내고, event flag를 통해 wait상태로 대기한다.

```
OSMboxPost(colorMbox[0], (void *)&color);
OSFlagPend(grp, 0x08, OS_FLAG_WAIT_SET_ALL + OS_FLAG_CONSUME,
```

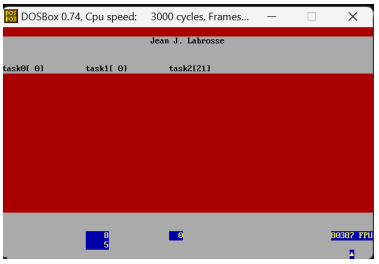
5. 이후 display task가 ready상태로 변하고, 나머지 task들이 모두 wait상태이기 때문에 mailbox에 적힌 색깔 정보를 얻는다.

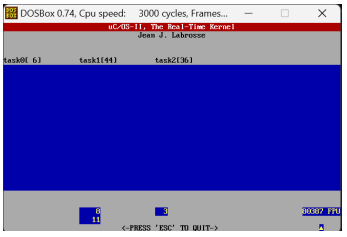
```
color = *(char *)OSMboxPend(colorMbox[0], 0, &err);
```

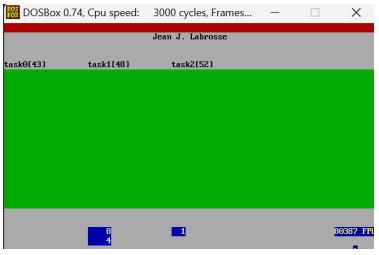
6. 특정 문자에 대해 배경화면을 색칠한 display task는, 이후 event flag의 값을 0x0F를 변경함으로써 나머지 task들이 깨어날 수 있게 한다.

```
OSTimeDlyHMSM(0, 0, 3, 0);
// post to all task (your code)
OSFlagPost(grp, 0x0F, OS_FLAG_SET, &err);
```

결과







5. 결과 분석

정상적으로 빨강 파랑 초록이 번갈아 수행되는 것을 확인할 수 있다.