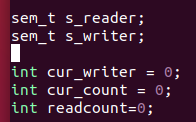
**2016025469 컴퓨터공학과 서건식**

**운영 체제 HW#6**

**제출 일자 : 2020/05/08**

1. **과제 A**
2. **자료구조 설명**

**이 과제에서 사용한 자료구조는 다음과 같습니다.**

****

****

**Cur\_writer와 cur\_count는 각각 접근한 writer의 ID와 접근한 횟수를 알려주며 readcount는 First Reader – writers problem을 해결하기 위한 변수입니다.**

**S\_reader / s\_writer라는 세마포어를 각각 생성해줍니다.**

1. **함수 설명**

****

**Reader / writer 함수는 각각 thread\_reader / thread\_writer가 사용할 함수 들입니다. 자세한 설명은 4번에서 하겠습니다.**

1. **프로그램 구조 설명**

**Main 함수를 기준으로 설명하면, 처음에 semaphore를 read와 writer 각각 한 개씩 생성해 준 후 쓰레드를 각각 만들어 줍니다. Reader는 2개, Writer 쓰레드는 5개 생성한 후에 pthread\_join으로 쓰레드의 종료를 기다린 후에 다 종료되면 semaphore를 삭제합니다.**

1. **프로그램이 어떻게 First Reader-Writers Problem을 해결하는지 설명**

**교수님 ppt의 sudo code를 응용하였습니다. read할때는 readcount 값이 1이되면 writer는 접근하지 못하게 하고 reader는 들어올 수 있도록 설정한 후에 critical section에 들어갑니다. Critical section이 끝나면 writer도 접근할 수 있도록 readcount를 0으로 설정 후 sem\_post로 unlock합니다.**

**write할때는 reader, writer 모두 Lock한 후에 critical section을 시행합니다.**

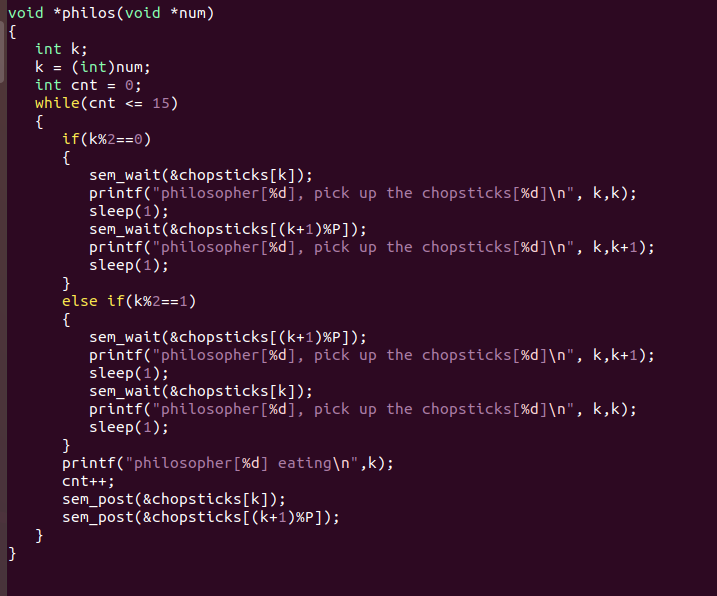
1. **과제 B**
2. **자료구조 설명**

****

**젓가락을 각각의 세마포어로, 철학자들을 쓰레드로 선언합니다.**

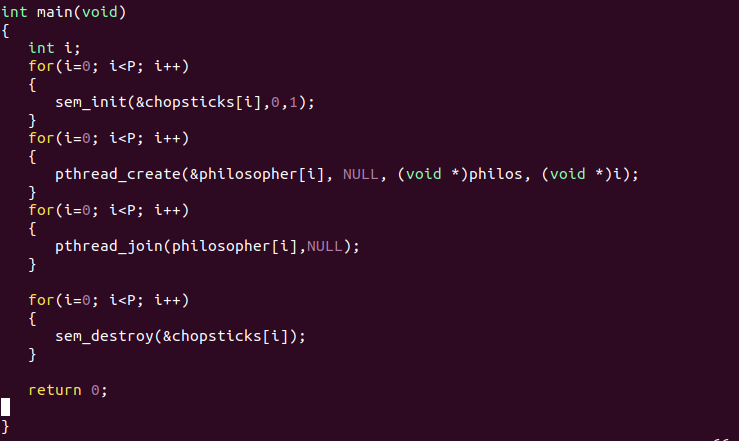
**P는 최대 철학자의 수이며 6입니다.**

1. **함수 설명**

****

**이 함수는 철학자들이 식사를 시작할 때의 함수입니다. 여기서 cnt는 큰 의미 없이 적당히 while문을 돌도록 설정해주었습니다. 자세한 내용은 4번에서 설명하겠습니다.**

1. **프로그램 구조 설명**

****

**Sem\_init으로 젓가락들을 초기화 해주고, pthread\_create로 철학자들의 쓰레드를 생성하고 join함수로 모두 끝날 때 까지 기다립니다. 이후 모든 semaphore를 삭제합니다.**

1. **프로그램이 어떻게 Dining-Philosophers Problem을 해결하는지 설명**

**교수님의 ppt를 응용하였습니다. Asymmetric solution으로 문제를 해결하였습니다.**

**K%2==0, 즉 Even number of philosophers의 경우 자신의 오른쪽 젓가락을 먼저 들고 왼쪽 젓가락을 들게 하고,**

**K%2 == 1, 즉 Odd number of philosophers의 경우 자신의 왼쪽 젓가락을 먼저 들고 오른쪽 젓가락을 들게 한다면 deadlock – free한 solution으로 문제를 해결할 수 있고, starvation의 가능성을 제거할 수 있습니다.**