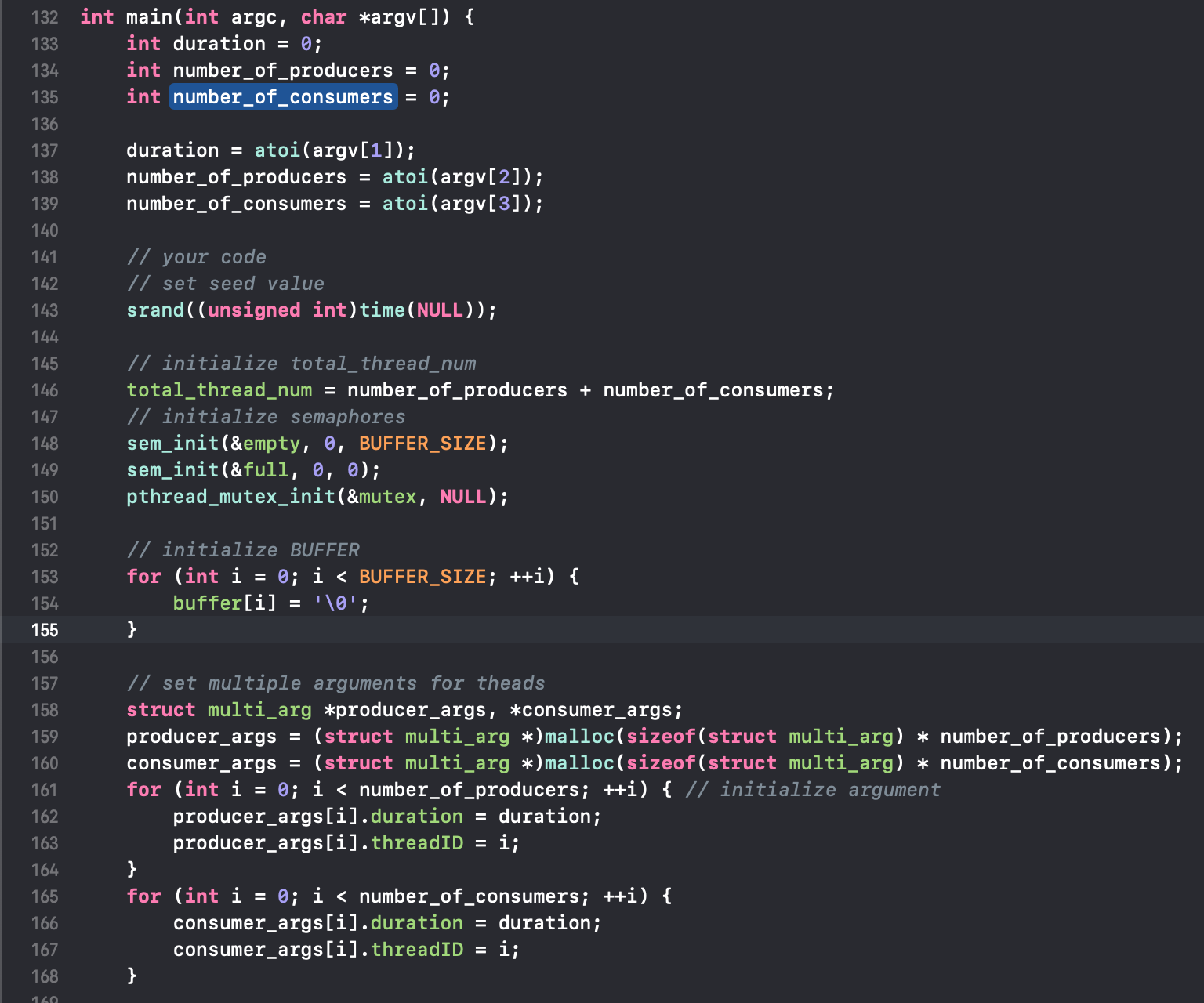
Operating System - 004

학번 12171633

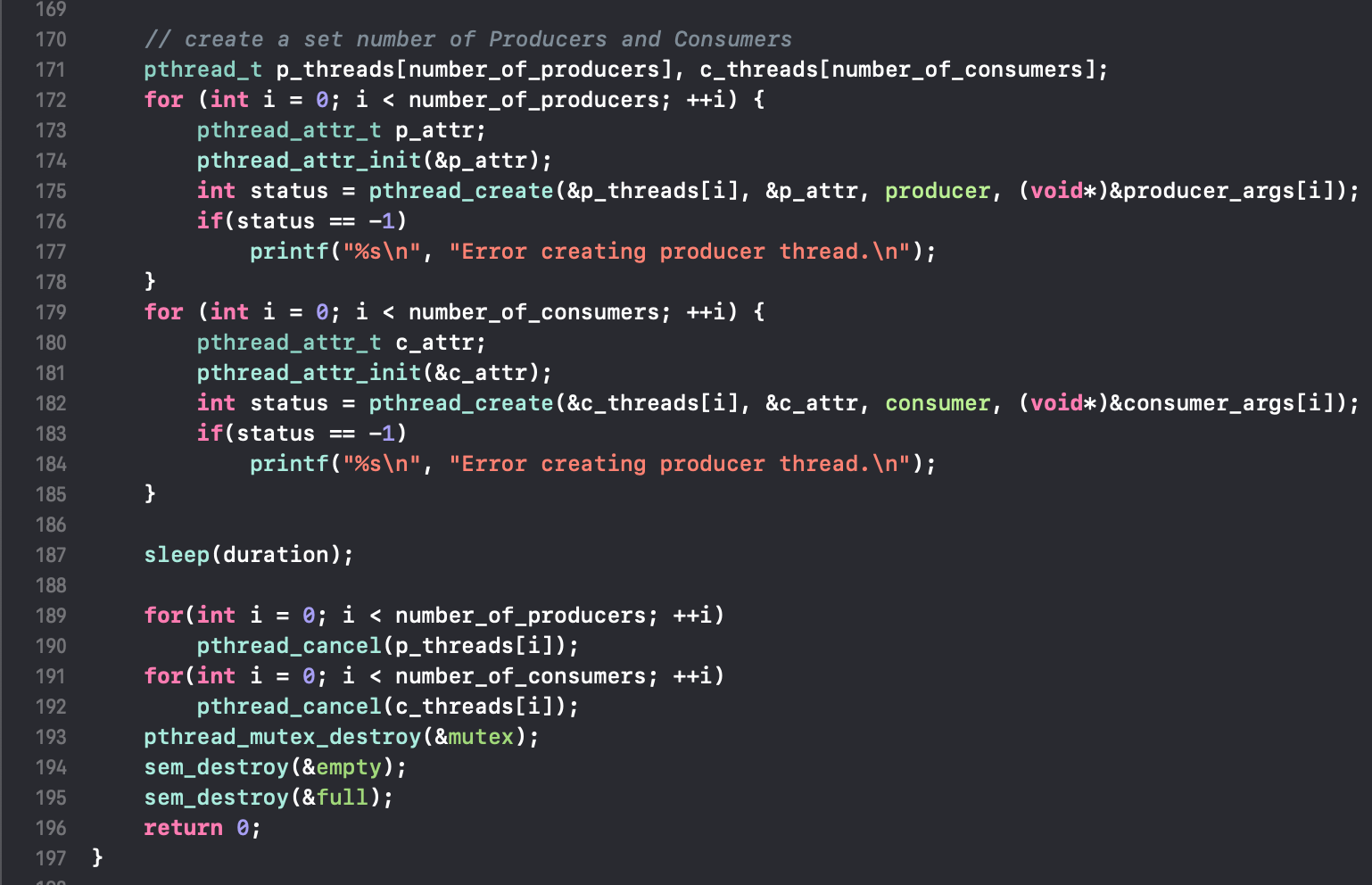
이름 서형중

2019년 5월 2일

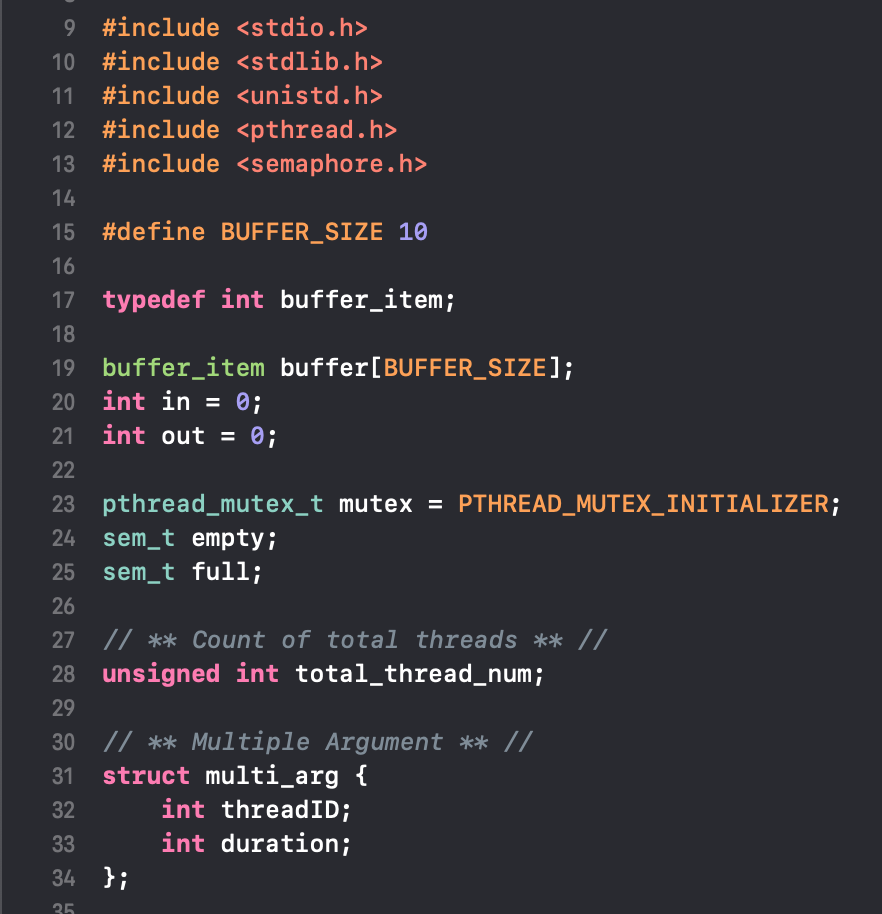
Report : 생산자 소비자 문제

먼저, main 함수에서의 코드를 보자. //your code 이전의 코드는 처음에 주어졌던 코드이므로 생략하기로 한다. srand(…)를 이용하여 시간에 따라 seed를 랜덤하게 바꾸도록 하여 이후 나오는 produce\_item()의 item 값이 랜덤하게 생성되도록 했다. 그 다음으로 total\_thread\_num은 생성하는 thread의 개수를 저장하는 변수로 이후 thread 안에서 sleep 시간을 조정할 때 사용한다. 이후 mutex와 semaphore 값을 init 함수를 이용해서 초기화 해주고, buffer의 값들도 초기화 해주었다.

Thread 생성시 thread id와 sleep 시간을 조정해줄 인자를 한꺼번에 넘겨주기 위해 multi\_arg라는 구조체를 생성하여 각 변수를 알맞는 값으로 초기화 해주었다.

line 117 to 186 : 생성자 쓰레드 배열과 소비자 쓰레드 배열을 선언해준 후 pthrad\_create를 통해 각 쓰레드의 performance함수와 인자를 넘겨줌으로써 thread를 생성해준다.

line 187 의 sleep(…)함수는 해당 duration 동안 thread가 동작할 수 있도록 main을 sleep 시킨다.

이후의 code는 생성한 쓰레드를 cancel 하고, 세마포어를 destroy한다.

line 9 to 13 : code 실행에 필요한 library 들을 include 해준다.

line 15 : 버퍼 크기를 10으로 정의한다.

line 17 : buffer\_item datatype을 int 형으로 정의해준다.

line 19 : buffer를 위에서 정의한 버퍼크기만큼 설정해준다.

line 20 to 21 : 버퍼에 접근할 포인터 in, out을 초기화 해준다.(buffer index 접근)

line 23 to 25 : mutex를 선언 및 정의해주고, 스레드 동기화를 위해 semaphore를 선언해준다.

이후, 전체 스레드 개수를 저장할 변수를 global로 선언해 각 쓰레드가 공통적으로 접근할 수 있도록 해주고, 스레드 생성시 여러개의 인자를 넘겨주기 위한 multi\_arg 구조체를 선언해준다.

produce\_item 은 랜덤한 값을 가지는 item을 생성해준다.

add\_item은 버퍼에 인자로 받아온 item을 추가해준 후 in 포인터 값을 조정한 후 생산한 item 값을 출력해준다.

remove\_item 은 buffer 배열을 인자로 받아오며, out포인터에 위치한 item를 return 해주기 위해 임시 변수를 선언 및 정의한 후 해당 위치의 buffer를 초기화(삭제) 해준다. 그 다음 out 포인터 값을 조정하고 제거된 item을 반환한다.

consume\_item 은 remove\_item에서 제거된 item을 인자로 받아와 소비한 item 값을 출력해준다.

producer는 먼저 void\*로 받아온 인자를 multi\_arg로 캐스팅 해주어 받아온 여러 인자를 사용할 수 있도록 한다. 그 다음 main에서 sleep함수가 지정한 시간동안 Entry Section을 통과한 후 Critical Section에 진입하여 produce\_item(…)과 add\_item(…)을 실행해준다. 또한 어떤 스레드가 item을 생성했는지 알 수 있도록 threadID를 출력해준다. (printf 는 IO\_interrupt를 발생시키는 system\_call를 호출하기 때문에 critical section에 위치하도록 한다.) critical section에서의 task를 모두 마친 후 mutex lock 을 해제해 다른 thread가 critical section에 진입할 수 있도록 하고, 또한 signal을 sleep 하고 있는 thread에 보내 해당 thread를 깨워준다.

consumer도 producer와 같이 받아온 인자를 캐스팅 해주고, Entry Section, Critical Section, Exit Section을 나누어 스레드 동기화가 이루어지도록 한다. consumer는 critical section에 진입하면 remove\_item(…)을 통해 버퍼에서 item을 삭제하고 consume\_item을 통해 삭제된 item을 소비해준다.