쓰레드 수행 과제

20185309

황명원

1.문제1번

1번 소스코드 : 쓰레드를 2개 생성해서 화면에 메시지를 출력하는 코드입니다.

수행화면:

텍스트이(가) 표시된 사진

자동 생성된 설명

동작 설명:

우선 예제코드를 보기전에 알아야 할것은 pthread\_create 와 pthread\_join 코드입니다.

pthread\_create 는 쓰레드를 생성하는 코드이며 세부적인 내용을 설명하기 위해 예시코드로 보면 이렇습니다.

(ex 코드)

int pthread\_create(pthread\_t \*thread, const pthread\_attr\_t \*attr,  
void \*(\*start\_routine)(void \*), void \*arg);

첫번째 매개변수인 thread 는 쓰레드가 성공적으로 생성되었을때 생성된 쓰레드를 식별하기 위해서 사용되는 쓰레드 식별자입니다.

두번째 매개변수인 attr 은 쓰레드 특성을 지정하기 위해서 사용하며, 기본 쓰레드 특성을 이용하고자 할경우에 NULL 을 사용합니다.

세번째 매개변수인 start\_routine는 분기시켜서 실행할 쓰레드 함수이며,

네번째 매개변수인 arg는 위 start\_routine 쓰레드 함수의 매개변수로 넘겨집니다.

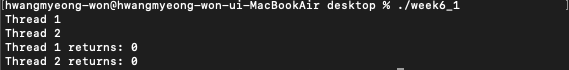
(성공적으로 리턴 할경우 0을 리턴합니다.)

그리고 pthread\_join 은 특정 쓰레드가 종료하기를 기다렸다가, 쓰레드가 종료된 이후 다음 진행하는 코드 입니다. 코드의 세부적인 내용을 설명하기 위해 예시 코드로 보면 이렇습니다.

int pthread\_join(pthread\_t th, void \*\*thread\_return);

첫번째 매개변수 **th**는 기다릴 쓰레드의 식별자 입니다.  
두번째 매개변수 **thread\_return**은 쓰레드의 리턴값이며  hread\_return 이 NULL이 아닌 경우 해당 포인터로 쓰레드의 리턴 값을 받아올수 있습니다.

즉 실행 결과를 보면



pthread\_create로 쓰레드를 생성하고 pthread\_join이 쓰레드가 종료될때까지 기다렸다가 실행되는 것을 볼수있습니다.(결과화면에 첫번째줄,두번째줄)

그러고 성공적으로 pthread\_create를 반환 했으므로 둘다 반환 값은 0이 나왔습니다.(결과화면에 세번째줄,네번째줄)

즉 이 코드를 실행할경우 쓰레드를 생성하고 종료될때까지 기다리고 실행하는 모습을 볼수 있습니다.

2.문제2번

2번 코드 : 다수의 쓰레드를 생성해서 counter 값을 증가시키는 코드입니다.

수행화면:

텍스트이(가) 표시된 사진

자동 생성된 설명

동작설명:

우선 각 코드들이 무슨 기능을 하는지(1번문제에서 설명한 코드는 설명 제외) 알기위해 하나씩 보자면 pthread\_self() 는 실행하면 현재 쓰레드의 식별자를 확인할수 있습니다.

그리고 pthread\_mutex\_t 라는 코드가 나오는데 mutex 는 여러개의 쓰레드가 공유하는 데이타를 보호하기 위해서 사용되는 도구로써, 보호하고자 하는 데이터를 다루는 코드영역을 한번에 하나의 쓰레드만실행가능 하도록 하는 방법으로 공유되는 데이타를 보호합니다. 즉 수업 시간에 배운 임계 영역의 내용입니다. 코드에서 mutex 있는 부분을 없애고 실행을 해보았더니 counter 가 예측하기 힘들게 출력이 되었습니다. pthread\_mutex\_lock,pthread\_mutex\_unlock 코드를 사용하므로써 임계영역을 지정해주고 돌리니 counter가 순서대로 잘 나오는것을 확인했습니다.

즉 여러개의 쓰레드를 생성하고 공통으로쓰는 전역변수인 counter를 1씩 더해줘도 서로 부딫히지않고 1씩 올라가는 걸 코드를 실행할때 확인할수 있습니다.

수행결과:

(counter의 증가를 계속해서 확인하기 위해 \*thread\_function안에 printf(“counter value: %d\n”,counter); 를 넣어 확인했습니다.)

(\*pthread\_mutex\_lock,pthread\_mutex\_unlock을 없앨경우)

텍스트이(가) 표시된 사진

자동 생성된 설명

(\*pthread\_mutex\_lock,pthread\_mutex\_unlock을 넣어 실행할경우)

텍스트이(가) 표시된 사진

자동 생성된 설명

3.문제3번

3번 코드 : 생산자-소비자 동작원리를 쓰레드로 구현했습니다.

수행화면

:텍스트이(가) 표시된 사진

자동 생성된 설명

동작설명:

우선 각 코드들이 무슨 기능을 하는지(1번,2번문제에서 설명한 코드는 설명 제외) 알기위해 하나씩 보자면

set\_t empty, set\_t full은 세마포어 생성을 의미합니다.

sem\_init(sem\_t \*sem, int pshared, unsigned int value); 여기서 하나씩 살펴보자면

-첫번째 인자: 세마포어 객체를 초기화 할 세마포어를 받습니다.

-두번째 인자: 여기에 0을 주지 않을경우 sem\_init는 항상 ENOSYS 에러코드를 반환합니다.(0을 쓰도록 합니다.)  
- 세번째 인자 : 세마포어를 몇으로 초기화 할지 의미합니다.

sem\_wait(sem\_t \*sem) 은세마포어의 P역할을 합니다. 즉, 세마포어를 하나 감소시키는 역할을 하고 세마포어가 0일 경우에는 1이상이 될 때까지 스레드는 대기 상태에 있고 0이 아닐 경우에는 대기상태에서 빠져나와 세마포어를 또 1 감소시킵니다.

sem\_post(sem\_t \*sem)은세마포어의 V역할을 하고 세마포어 값을 1 증가 시킵니다.

sem\_destroy(sem\_t \*sem) 은 세마포어와 관련된 리소스들을 소멸시킵니다. (객체 소멸)

usleep(값) 은 값 만큼 잠시 멈추는 코드입니다.(값/1000000초 로 계산합니다.)

즉 main 함수에서 세마포어 empty를5로 초기화 했으므로

for(int i = 0; i < 5; i++) {  
 pthread\_create(&pro[i], NULL, (void \*)producer, (void \*)&a[i]);  
}

부분에서 5번 이 수행될때

sem\_wait(&empty);

이 코드로 인해 empty라는 세마포어는 한개씩 줄어들고

sem\_post(&full);

이 코드로 인해 full이라는 세마포어는 한개씩 증가할것입니다.

그렇게 empty는 5번이 다돌면 0이 되고 쓰레드는 대기 상태로 들어갑니다.

그렇게 cousumer 함수가 5번 실행되며 이번엔 반대로 full이 한개씩 감소하고

empty가 한개씩 증가합니다. 이렇게 실행되는 과정은 sem\_wait와 sem\_post안에 있는 mutex 때문에 임계영역이 생겨 한개씩만 실행되며 usleep으로 인해 0.5초씩 쉬면서 실행되는것을 커맨드 창을 통해 확인할수 있습니다.

실행결과:

텍스트이(가) 표시된 사진

자동 생성된 설명