**1.스레드, 동기화 빠르게 복습후(**[**mutex & condition variable**](https://www.joinc.co.kr/w/Site/Thread/Beginning/ConditionVariable)**)**

**-pthread\_메소드들은 ok**

**-mutex사용방법도 ㅇㅋ** 그냥 mutex 변수 선언 후 초기화  
 critical section 접근하기 전에 lock(mutex),  
 critical section 떠날때 unlock(mutex)

**-spin-lock은 cpu를 잡아먹으므로 좋지않음.** sleep 시키기 위해서 condition variable 사용  
 condition variable이 true -> wakes up(signal,다른스레드)  
 condition variable이 false -> sleep(waiting,스스로) -> wait queue에 삽입

**-condition wait (\*cond, \*mutex)** wait를 호출하게되면 해당 스레드는 스스로 sleep  
 sleep전에 lock해줬던 mutex를 unlock해줘야하므로 매개변수가 cond,mutex  
 wait 다음과 같이 3단계로 이루어짐.(unlock mutex, sleep, lock mutex)

**-condition signal(\*cond)** cond에 대해서 signal을 보냄

**cf)wait,signal이 시간이 오래걸린다면 문제가생김.** 본인이 wait나 signal을 보내고 남을 기다리지 않으므로 생기는 문제.  
 그러면 기다리게해야지. 어떻게? 조건변수를 하나더 넣어서 아래와 같이 상대방이 받았나확인

Thread1 Thread2  
 signal(condition variable1) -> wait(condition variable1) // 특정조건 만족시 변수1으로 동기화  
 wait(condition variable2) <- signal(condition variable2) // 변수1의 신호를 수신했다는 답장

**2.과제 정리**

**cf)** Thread\* pTh = \_\_getThread(pthread\_self()); // \_\_getThread는 어디서?

**-TCB 구조체** pthread\_cond\_t(condition) / pthread\_mutex\_t(mutex) / bRunnable

**/-int thread\_create** 만들자마자 ready\_queue 뒤에  
 구현방법 나와있음. Wrapper function 이용

**/-thread\_t thread\_self** 그냥 자기자신 반환

**/-int thread\_join(thread\_t thread, void\*\* retval)** pthread\_mutex\_lock(&pthread\_mutex\_t);  
 // Critical Section  
 pthread\_cond\_signal(&pthread\_cond\_t);  
 pthread\_mutex\_unlock(&pthread\_mutex\_t);

**/-int thread\_exit(void\* retval)** 스레드가 종료되기전에 호출  
 pthread\_mutex\_lock(&pthread\_mutex\_t);  
 while (조건)  
 pthread\_cond\_wait(&pthread\_cond\_t, &pthread\_mutex\_t);  
 pthread\_mutex\_unlock(&pthread\_mutex\_t);

**-void \_\_thread\_wait\_handler(int signo) // sinal handler, signal은 SIGUSR1이다.** running -> waiting ? running -> ready  
 thread를 재우는 wait함수

**/-int thread\_suspend(thread\_t target)** target을 ready -> waiting  
 ready queue에 있는 thread를 sleep시키고 waiting queue로 이동  
 구현방법 나와있음.  
 target에 SIGUSR1을 보냄. target이 SIGUSR1을 받으면 \_\_thread\_wait\_handler으로 waiting으로감  
 ㄴㄴ 이게 아님. 이미 ready중인 thread를 \_\_thread\_wait\_handler를 호출할필요 ㄴㄴ  
 ready queue에 잇는 thread를 waiting에 넣어주고 상태만 바꿔주자

**/-int thread\_resume(thread\_t tid)** tid를 waiting -> ready  
 구현방법 나와있음

**-void \_\_thread\_wakeup(Thread\* pTh)**  
 ready -> running  
 resume thread 내부를 구현.  
 pthread\_mutex\_lock(&(pTh->readyMutex));  
 pTh->bRunable=TREU;  
 pthread\_cond\_signal(&(pTh->readyCond));  
 pthread\_mutex\_unlock(\*(pTh->readyMutex));  
**-pthread\_kill(tid, SIGUSR1)** running -> ready

**/-void \_\_ContextSwitch(Thread pCurThread, Thread\* pNewThread)** pCurThread는 ready로, pNewThread는 running으로

**/-void RunScheuler()** 무한반복.  
 ready queue가 비었으면 바로 sleep(일정시간)