1. 소스코드

|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  #include <unistd.h>  #include <errno.h>  #include <pthread.h>  #include <sys/time.h>  //뮤텍스 변수 초기화  pthread\_mutex\_t lock = PTHREAD\_MUTEX\_INITIALIZER;  //조건 변수 초기화  pthread\_cond\_t cond = PTHREAD\_COND\_INITIALIZER;  //공유 변수  int glo\_val1 = 1, glo\_val2 = 2;  void \*ssu\_thread1(void \*arg);  void \*ssu\_thread2(void \*arg);  int main(void) {  pthread\_t tid1, tid2;  //스레드 생성  pthread\_create(&tid1, NULL, ssu\_thread1, NULL);  pthread\_create(&tid2, NULL, ssu\_thread2, NULL);  //스레드가 종료할 때까지 대기  pthread\_join(tid1, NULL);  pthread\_join(tid2, NULL);  //뮤텍스 변수 해제  pthread\_mutex\_destroy(&lock);  //조건 변수 해제  pthread\_cond\_destroy(&cond);  exit(0);  }  void \*ssu\_thread1(void \*arg) {  sleep(1);  glo\_val1 = 2;  glo\_val2 = 1;  if(glo\_val1 > glo\_val2) //cond를 기다리고 있는 모든 스레드를 재시작  pthread\_cond\_broadcast(&cond);  printf("ssu\_thread1 end\n");  return NULL;  }  void \*ssu\_thread2(void \*arg) {  struct timespec timeout;//nano sec  struct timeval now;//macro sec  //lock  pthread\_mutex\_lock(&lock);  //시간을 now에 저장  gettimeofday(&now, NULL);  timeout.tv\_sec = now.tv\_sec + 5;  timeout.tv\_nsec = now.tv\_usec \* 1000;  if(glo\_val1 <= glo\_val2) {  printf("ssu\_thread2 sleep\n");    //정해진 기간까지 cond를 기다림  if(pthread\_cond\_timedwait(&cond, &lock, &timeout) == ETIMEDOUT)  printf("timeout\n");//기한이 지난 경우  else//기한 내에 cond를 받은 경우  printf("glo\_val1 = %d, glo\_val2 = %d\n", glo\_val1, glo\_val2);  }  //unlock  pthread\_mutex\_unlock(&lock);  printf("ssu\_thread2 end\n");  return NULL;  } |

2. 실행결과

