15 Gennaio 2016

5) Si completi il seguente codice affinché i thread dell'esercizio seguente attraversino la barriera secondo l'ordine imposto dal numero progressivo id.

```
/* Include */
#include <pthread.h>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <unistd.h>
#define NUM_THREADS     5

// Variabili globali
int next_thread;

pthread_mutex_t condition_mutex;
pthread_cond_t condition_variable;
void attendi_turno(int id){

  pthread_mutex_lock(&condition_mutex);
  while(next_thread != id)
       pthread_cond_wait(&condition_variable);


   printf("Thread %d ha attraversato la barriera\n", id);


    ++next_thread;
    pthread_cond_broadcast(&condition_variable);
    pthread_mutex_unlock(&condition_mutex);

}

void * tbody (void *num) {
    //Aspetto un po'
    srand(time(NULL));
    int r = rand()%10;
    sleep(r);
    printf("Thread %d ha iniziato\n", (int) num);
    attendi_turno((int) num);
    pthread_exit(NULL);
}

int main() {
//Inizializzazioni


   condition_mutex = PTHREAD_MUTEX_INITIALIZER;
   contition_variable = PTHREAD_COND_INITIALIZER;

   next_thread=0;

   pthread_t threads[NUM_THREADS];
   int rc, t;
   for(t=0; t<NUM_THREADS; t++){
       printf("Creating thread %d\n", t);
       rc = pthread_create(&threads[t], NULL, tbody, (void *) t);
       if (rc){
           printf("ERROR; return code from pthread_create() is %d\n", rc);
           exit(-1);
       }
   }
   pthread_exit(NULL);
}
```