```c
/* Prueba de MUTEX */

#define _POSIX_C_SOURCE 199506L
#define _REENTRANT

#include <string.h>
#include <unistd.h>
#include <pthread.h>
#include <stdio.h>
#include <time.h>

void *hilo1(void *p);
void *hilo2(void *p);

int acabar;

#define L_CAD 50

char cadena[ L_CAD];
pthread_mutex_t muti = PTHREAD_MUTEX_INITIALIZER;

int main(void)
{
  pthread_t th1;
  pthread_t th2;
  int i;
  int res;
  void *result;

  /* Primero se deja a hilo2 que modifique cuando quiera los datos */

  printf("main: Comenzando la primera prueba, sin mutex\n");

  for(i=0; i<L_CAD; i++) cadena[ i] = '\0';
  acabar = 0;

  res = pthread_create(&th1, NULL, hilo1, NULL);
  res = pthread_create(&th2, NULL, hilo2, (void *)0);

  pthread_join(th1, &result);

  /* Cuando acaba th1, paramos a th2 */

  acabar = 1;
  pthread_join(th2, &result);

  /* Ahora se usan mutex en los dos hilos */

  printf("main: Comenzando la segunda prueba, con mutex\n");

  for(i=0; i<L_CAD; i++) cadena[ i] = '\0';

  res = pthread_create(&th1, NULL, hilo1, NULL);
  res = pthread_create(&th2, NULL, hilo2, (void *)1);

  pthread_join(th1, &result);

  /* Cuando acaba th1, paramos a th2 */

  acabar = 1;
  pthread_join(th2, &result);

  printf("Final del programa\n");

}

/* Hilo 1 */
```

```c
#define N_ITER 40

void *hilo1(void *arg)
{
  int cnt = N_ITER;
  char *msg1 = "ABCDEF";
  char aux[ L_CAD];
  int nf;

  struct timespec tim = { 0, 1000000L};

  nf = 0;
  while(cnt--)
  {
    /* En cada iteracion se compara lo que se puso con lo que
       se ha encontrado */

    pthread_mutex_lock(&muti);
    strcpy(cadena, msg1);

    /* En general no se debe esperar en la seccion critica; es solo
       para forzar el fallo */

    nanosleep(&tim, NULL);
    strcpy(aux, cadena);
    pthread_mutex_unlock(&muti);

    if(strcmp(aux, msg1) != 0)
    {
      printf("hilo1: Error; cadena encontrada: %s\n", aux);
      nf++;
    }
  }

  printf("hilo1: %d fallos de %d iteraciones.\n", nf, N_ITER);
  return NULL;
}

/* Hilo 2 */

void *hilo2(void *arg)
{
  char *msg1 = "hilo2 estuvo aqui";

  struct timespec tim = { 0, 20000000L};

  /* Cada 20 msg. modifica el valor, con o sin mutex */

  while(!acabar)
  {
    if((int)arg) pthread_mutex_lock(&muti);
    strcpy(cadena, msg1);
    if((int)arg) pthread_mutex_unlock(&muti);
    nanosleep(&tim, NULL);
  }
}
```

```
main: Comenzando la primera prueba, sin mutex
hilo1: Error; cadena encontrada: hilo2 estuvo aqui
hilo1: Error; cadena encontrada: hilo2 estuvo aqui
hilo1: Error; cadena encontrada: hilo2 estuvo aqui
hilo1: Error; cadena encontrada: hilo2 estuvo aqui
hilo1: Error; cadena encontrada: hilo2 estuvo aqui
hilo1: Error; cadena encontrada: hilo2 estuvo aqui
hilo1: Error; cadena encontrada: hilo2 estuvo aqui
hilo1: Error; cadena encontrada: hilo2 estuvo aqui
hilo1: Error; cadena encontrada: hilo2 estuvo aqui
hilo1: Error; cadena encontrada: hilo2 estuvo aqui
hilo1: Error; cadena encontrada: hilo2 estuvo aqui
hilo1: Error; cadena encontrada: hilo2 estuvo aqui
hilo1: Error; cadena encontrada: hilo2 estuvo aqui
hilo1: Error; cadena encontrada: hilo2 estuvo aqui
hilo1: Error; cadena encontrada: hilo2 estuvo aqui
hilo1: Error; cadena encontrada: hilo2 estuvo aqui
hilo1: 16 fallos de 40 iteraciones.
main: Comenzando la segunda prueba, con mutex
hilo1: 0 fallos de 40 iteraciones.
Final del programa
```