Variable	Value
counter	0
max	3
mutex	0
cond_var	0

stderr:

0: before mist()
1: before mist()
2: before mist()
2: after mist()
0: after mist()
1: after mist()

¿Qué hace mistery()?

Implementa una barrera con una variable de condicion

Thread 0		Thread 1		Thread 2	
41	void* run(void* data) {				
42	fprintf(stderr, "%zu: before mist()\n", (size_t)data);	41	void* run(void* data) {		
43	sleep((unsigned)(size_t)data);	42	fprintf(stderr, "%zu: before mist()\n", (size_t)data);	41	void* run(void* data) {
26	void mistery(mist_t* mist) {	43	sleep((unsigned)(size_t)data);	42	fprintf(stderr, "%zu: before mist()\n", (size_t)data);
27	pthread_mutex_lock(&mist->mutex);	99	---zzz---	43	sleep((unsigned)(size_t)data);
28	++mist->counter;	44	mistery(&mist);	99	---zzz---
29	if (mist->counter < mist->max) {	26	void mistery(mist_t* mist) {	99	---zzz---
31	pthread_cond_wait(&mist->cond_var, &mist->mutex);	27	pthread_mutex_lock(&mist->mutex);	44	mistery(&mist);
99	---zzz---	99	---zzz---	26	void mistery(mist_t* mist) {
99	---zzz---	28	++mist->counter;	27	pthread_mutex_lock(&mist->mutex);
99	---zzz---	29	if (mist->counter < mist->max) {	99	---zzz---
99	---zzz---	31	pthread_cond_wait(&mist->cond_var, &mist->mutex);	99	---zzz---
99	---zzz---	99	---zzz---	28	++mist->counter;
99	---zzz---	99	---zzz---	29	if (mist->counter < mist->max) {
99	---zzz---	99	---zzz---	33	mist->counter = 0;
99	---zzz---	99	---zzz---	34	pthread_cond_broadcast(&mist->cond_var);
99	---zzz---	99	---zzz---	36	pthread_mutex_unlock(&mist->mutex);
36	pthread_mutex_unlock(&mist->mutex);	99	---zzz---	45	fprintf(stderr, "%zu: after mist()\n", (size_t)data);
45	fprintf(stderr, "%zu: after mist()\n", (size_t)data);	36	pthread_mutex_unlock(&mist->mutex);	46	return NULL;
46	return NULL;	45	fprintf(stderr, "%zu: after mist()\n", (size_t)data);		
		46	return NULL;		