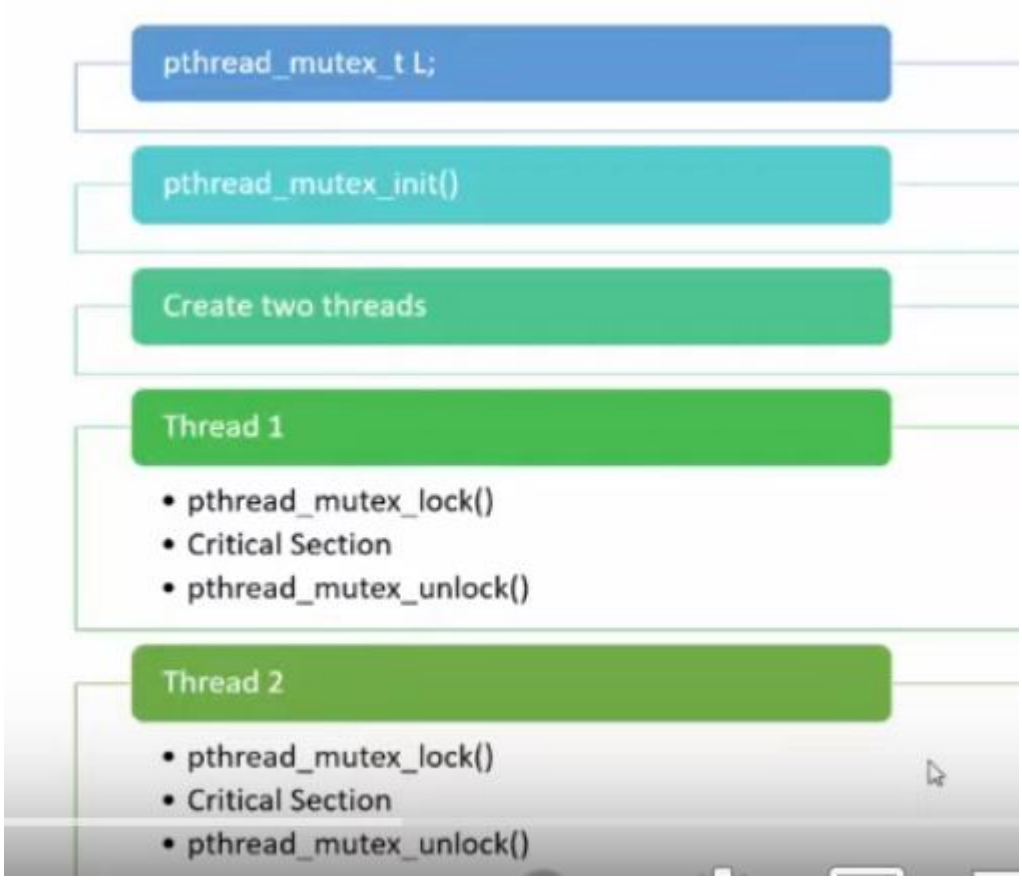
Thread Concept (Mutex):



/* Program for process synchronization using locks

Program create two threads: one to increment the value of a shared variable and second to decrement the value of shared variable. Both the threads make use of locks so that only one of the threads is executing in its critical section */

```
#include<pthread.h>
#include<stdio.h>
#include<unistd.h>
void *fun1();
void *fun2();
 int shared=1; //shared variable
 pthread_mutex_t l; //mutex lock
 int main()
 {
 pthread_mutex_init(&l, NULL); //initializing mutex locks
 pthread_t thread1, thread2;
 pthread_create(&thread1, NULL, fun1, NULL);
 pthread_create(&thread2, NULL, fun2, NULL);
 pthread_join(thread1, NULL);
 pthread_join(thread2,NULL);
 printf("Final value of shared is %d\n",shared); //prints the last updated value of shared variable
 }

// function1 executed by thread1
void *fun1()
 {
    int x;
    printf("Thread1 trying to acquire lock\n");
    pthread_mutex_lock(&l); //thread one acquires the lock. Now thread 2 will not be able to
acquire the lock //until it is unlocked by thread 1
   printf("Thread1 acquired lock\n");
    x=shared;//thread one reads value of shared variable
    printf("Thread1 reads the value of shared variable as %d\n",x);
    x++;  //thread one increments its value
    printf("Local updation by Thread1: %d\n"x,);
    sleep(1);  //thread one is preempted by thread 2
    shared=x; //thread one updates the value of shared variable
    printf("Value of shared variable updated by Thread1 is: %d\n",shared);
```

```
    pthread_mutex_unlock(&l);
    printf("Thread1 released the lock\n");
 }
void *fun2()
 {
    int y;
    printf("Thread2 trying to acquire lock\n");
    pthread_mutex_lock(&l);
    printf("Thread2 acquired lock\n");
    y=shared;//thread two reads value of shared variable
    printf("Thread2 reads the value as %d\n",y);
    y--;  //thread two increments its value
    printf("Local updation by Thread2: %d\n",y);
    sleep(1); //thread two is preempted by thread 1
    shared=y; //thread one updates the value of shared variable
    printf("Value of shared variable updated by Thread2 is: %d\n",shared);
    pthread_mutex_unlock(&l);
    printf("Thread2 released the lock\n");
 }


/*
OUTPUT-
lm@lm-VirtualBox:~$ gcc mutex.c -o mutex -lpthread
lm@lm-VirtualBox:~$ ./mutex
Thread2 trying to acquire lock
Thread2 acquired lock
Thread2 reads the value as 1
Local updation by Thread2: 0
Thread1 trying to acquire lock
Value of shared variable updated by Thread2 is: 0
Thread2 released the lock
Thread1 acquired lock
Thread1 reads the value of shared variable as 0
Local updation by Thread1: 1
Value of shared variable updated by Thread1 is: 1
Thread1 released the lock
Final value of shared is 1
*/
```