```c
//inclusion of headers
#include "_threadsCore.h"
#include "_kernelCore.h"
#include "osDefs.h"

//inclusion of libraries
#include <LPC17xx.h>
#include "stdio.h"
#include "uart.h"
#include "LED.h"

//global variables
int thread1Call = 0;
int thread2Call = 0;
int thread3Call = 0;
int x = 0;
extern threadStruct threadCollection[MAX_THREADS];
extern int threadCurr;

//thread functions
void thread1(){
  while(1){
    //CASE 1
    osAcquireMutex(UART);
    thread1Call++;
    printf("Running thread 1. Call count: %d\n", thread1Call);
    osReleaseMutex(UART);

    //CASE 2
    /*
    osAcquireMutex(GLOBAL_X);
    x++;
    printf("Running thread 1. x = %d\n", x);
    osReleaseMutex(GLOBAL_X);
    */

    osYield();
  }
}

void thread2(){
  while(1){
    //CASE 1
    osAcquireMutex(UART);
    thread2Call++;
    printf("Running thread 2. Call count: %d\n", thread2Call);
    osReleaseMutex(UART);
    osYield();

    //CASE 2
    /*
    osAcquireMutex(GLOBAL_X);
    osAcquireMutex(LEDS);

    LED_display(x%47);
    printf("Running thread 2\n");

    osReleaseMutex(LEDS);
    osReleaseMutex(GLOBAL_X);
    */
  }
}

void thread3(){
  while(1){

    //CASE 1
    osAcquireMutex(UART);
    thread3Call++;
    printf("Running thread 3. Call count: %d\n", thread3Call);
    osReleaseMutex(UART);

```

```
73        osYield();
74
75        //CASE 2
76        /*
77        osAcquireMutex(LEDS);
78        LED_display(0x71);
79        printf("Running thread 3.\n");
80        osReleaseMutex(LEDS);
81        */
82      }
83    }
84
85    int main( void )
86    {
87      SystemInit();
88      printf("\nRunning L-OS-S...\r\n");
89
90      //Clear all LEDs
91      LED_setup();
92      LED_clear(0);
93      LED_clear(1);
94      LED_clear(2);
95      LED_clear(3);
96      LED_clear(4);
97      LED_clear(5);
98      LED_clear(6);
99      LED_clear(7);
100
101     //Initialize the kernel
102     kernelInit();
103
104     //Initialize each thread
105     osThreadNew(thread1, TIMESLICE_DEFAULT, 0);
106     osThreadNew(thread2, TIMESLICE_DEFAULT, 0);
107     osThreadNew(thread3, TIMESLICE_DEFAULT, 0);
108
109     osThreadNew(idleThread, TIMESLICE_IDLE, 0); //always initialize last
110
111     //Create each mutex
112     osCreateMutex(); //UART
113     osCreateMutex(); //GLOBAL_X
114     osCreateMutex(); //LEDS
115
116     //Initialize frequency of SysTick_Handler
117     SysTick_Config(SystemCoreClock/1000);
118
119
120     //Start running the threads
121     osKernelStart();
122
123     printf("L-OS-S is lost (done)");
124     while(1);
125   }
126
```