```
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <time.h>
#include <pthread.h>
#include <semaphore.h>

#define MAX 50 // maximum number of threads

int sleepMod = 5;
int readCount = 0;
sem_t readAccess, bookAccess;

void *reader_func(void *);
void *writer_func(void *);

int main() {
        srand(time(0));
        int readers, writers;
        printf("Number of readers (max 50): ");
        scanf("%d", &readers);
        printf("Number of writers (max 50): ");
        scanf("%d", &writers);

        // set sleep mode
        if(readers > 5) sleepMod = readers;

        pthread_t readers_t[MAX], writers_t[MAX];
        sem_init(&readAccess, 0, 1);
        sem_init(&bookAccess, 0, 1);

        int i=0;

        for(i=0; i<readers; i++)
                pthread_create(&readers_t[i], NULL, reader_func, &i);

        for(i=0; i<writers; i++)
                pthread_create(&writers_t[i], NULL, writer_func, &i);

        for(i=0; i<writers; i++)
                pthread_join(writers_t[i], NULL);

        for(i=0; i<readers; i++)
                pthread_join(readers_t[i], NULL);
}

void *reader_func(void *r) {
        int rNo = *((int *)r) + 1;
        printf("\n reader %d : wanting to read", rNo);
        sleep(rand()%sleepMod);
        sem_wait(&readAccess);
                readCount+=1;
                if (readCount == 1) {
                        sem_wait(&bookAccess);
                }
                printf("\n reader %d : reading", rNo);
        sem_post(&readAccess);

        sleep(rand()%sleepMod);

        sem_wait(&readAccess);
                readCount-=1;
                printf("\n reader %d : leaving reading", rNo);
                sleep(rand()%sleepMod);
                if ( readCount == 0) {
                        sem_post(&bookAccess);
                }
        sem_post(&readAccess);
        printf("\n reader %d : finished", rNo);
        sleep(rand()%sleepMod);
        pthread_exit(0);
}

void *writer_func(void *w) {
        int wNo = *((int *)w) + 1;
        printf("\n writer %d : wanting to write", wNo);
        sleep(rand()%sleepMod);
        sem_wait(&bookAccess);
        printf("\n writer %d : writing", wNo);
        sleep(rand()%sleepMod);
        printf("\n writer %d : leaving writing", wNo);
        sleep(rand()%sleepMod);
        sem_post(&bookAccess);
        printf("\n writer %d : finished", wNo);
        sleep(rand()%sleepMod);
        pthread_exit(0);
}
```