EXERCISE

Assume the following program logic using (pthread) threads (ignore syntax in each statement),

```c
pthread_mutex_t mutex;
pthread_cond_t condition;

void Thread1(void* arg)
{
    /* Thread 1 repeats this twice --------------------------- */
    for (i = 0; i < 2; i++)
    {
        // generate random wait (sleep) time
        time_to_wait = rand();
        sleep(time_to_wait);

        // perform "event" and notifies it to the receiver
        pthread_mutex_lock(&mutex);
        pthread_cond_signal(&condition);
        pthread_mutex_unlock(&mutex);
    }
    return NULL;
}

void Thread2(void* arg)
{
    /* Thread2 repeats this twice -------------------------- */
    for (i = 0; i < 2; i++)
    {
        // generate random wait (sleep) time
        time_to_wait = rand();
        sleep(time_to_wait);

        // wait for the sender to perform "event"
        pthread_mutex_lock(&mutex);
        pthread_cond_wait(&condition, &mutex);
        pthread_mutex_unlock(&mutex);

        // I got it ....................
        printf("        T2 is signaled by T1!\n\n");
    }
    return NULL;
}
```

QUESTIONS: If the two threads (Thread1 and Thread2) are started roughly at the same time,

Question 1: Is it possible for the user who executes the two threads does not see any output ("T2 is signaled by T1!") from Thread 2? If yes, how? If not, why not?

Question 2: Is it possible for the user who executes the two threads does not see any output ("T2 is signaled by T1!") from Thread 2 twice? If yes, how? If not, why not?

Question 3: Is it possible for the user who executes the two threads does not see any output ("T2 is signaled by T1!") from Thread 2 once (exactly once)? If yes, how? If not, why not?