2. Let's now generalize this a bit by investigating the rendezvous problem The problem is as follows you have two threads each of which are about to enter the rendezvous point in the code Neither should exit this part of the code before the other enters it Consider using two semaphores for this task and see rendezvous c for details

- The semaphores need to be as close to the critical part as possible and in this case the critical part is the second printing line in the functions *child* and *child* the interesting thing is that the threads won't change anything inside the other thread's function instead they will affect the order in which these printing lines are delivered out of order Two calls to sleep also helped by making a thread wait and making the other one run ensuring that the order of lines being delivered is always in order