Main Thread

- Declare and initialize global data/variables which require synchronization
- Declare and initialize a condition variable object
- Declare and initialize an associated mutex
- Create threads A and B to do work

Thread A

- Do work up to the point where a certain condition must occur (such as count must reach a specified value)
- Lock associated mutex and check value of a global variable
- Call pthread_cond_wait() to perform a blocking *wait for signal* from Thread B
  Note that a call to pthread_cond_wait() automatically and atomically unlocks the associated mutex variable so that it can be used by Thread B
- When signalled, wake up
  Mutex is automatically and atomically locked
- Explicitly unlock mutex
- Continue

Thread B

- Do work
- Lock associated mutex
- Change the value of the global variable that Thread A is waiting upon
- Check value of the global Thread A wait variable
- If it fulfills the desired condition, signal Thread A
- Unlock mutex
- Continue

Main Thread

- Join / Continue