CS 4414 Operating System
Project 4
10/27/15
Aki Gao
xag9bb

**Design:**

The barrier was implemented with mutexs and using pthread condition functions.
The steps are as follows:

1. Lock mutex
2. Decrement global variable tracking the number of threads
3. If global variable tracking the number of threads is greater than zero, call wait on the condition variable to suspend thread.
4. Else, invoke the function that broadcasts the condition variable to release all suspended threads.
5. Unlock mutex

The barrier was tested manually printing all threads in execution and checking that the appropriate number of threads are waiting until the last thread executes. For example, for a list of four numbers, one thread should wait until the last thread executes in the first round of sorting. In the next round, no thread will wait as there is only one thread.

The main thread does not participate in the comparisons. However, it does lock the for loop which generates the rounds using a while loop that runs while a global variable that tracks the number of completed threads is greater than zero.

Lastly, the list is stored in memory as a single array. Because threads never manipulate the same indexes of the list, memory is manipulated in-place. Therefore, the same array as the original array that contains the unsorted list is used to store intermediate results.