CS1760: Multiprocessor Synchronization **Due: 6 December 2018**

Professor: Maurice Herlihy *HTA: Jonathan Lister*

Programming Assignment 5 - Bitonic Counting Networks

For this programming assignment, you will need to implement a Bitonic counting network, as described in Chapter 12 of the book, that supports both traversals (used by tokens) and anti-traversals (used by anti-tokens). Note that the book does have some mistakes - errata is listed here.The stencil code for this assignment is located at the course's pub directory at `/course/cs1760/stencil/bitonic/` and can be installed with

```
cs1760_install bitonic
```

A Bitonic counting network is a recursively-defined network of Mergers and Balancers that can be used as a counter in multithreaded programs.

Once you copy the stencil code into your directory, you'll be ready to go! The files you need to modify are:

```
Bitonic.java
Merger.java
Balancer.java
TestBitonic.java
BitonicTester.java
```

Note that a portion of this assignment's total score is reserved for testing. Passing basic functionality tests, located in `BasicTestBitonic.java` and `BasicBitonicTester.java`, will reward you with some points; however, we also expect you to write any additional tests you deem necessary.
Failing basic functionality tests does not necessarily imply that you will receive no credit for the assignment. At the same time, the course staff will be unable to conduct a rigorous inspection of non-functional code to award partial credit.

To hand in your code for this assignment, run

```
cs1760_handin bitonic
```

Here are some tips to get you started on the assignment:

1. Make sure you understand the recursive structure and the functioning of the Bitonic counting network described in the textbook - this will make testing and debugging your program much easier.

2. Recall that given the same Balancer state, an anti-traversal will send its anti-token on the output wire opposite to the one that a normal traversal would send its token on.

3. The file `Consts.java` provides definitions for constants that you are free to use at your discretion. Look at the comments in this file for an explanation.