CSCD 467 Concurrent Systems Lab 6
Implement Our Own CyclicBarrier Class

Rules: Your code must use Java Language. If your program shows a compilation error, you get a zero credit for this lab assignment. To avoid compatibility issues, I encourage you to upgrade your JRE to latest version of SE 1.7 or 1.8.

Grading: Before you leave the laboratory, please show the instructor or TA how your program works. A score will be given to you before you leave.

Submission: Wrap up all your java files into a single zip file. Name your zip file as firstInitialYourLastNameCSCD467Lab6.zip. For example, if your legal name is Will Smith, you should name your zip file as wSmithCSCD467Lab6.zip.

For archive purpose, please also submit your single zip file on EWU Canvas by following CSCD467-01 Course → Assignments →Lab6 → Submit Assignment to upload your single zip file.

What you should do?

In the class demo posted on the course website under DemoCode section as UseCyclicBarrierDemo.zip, we used the Java built-in CyclicBarrier class to synchronize N threads. A barrier suspends threads that reach a barrier point earlier until all threads have reached the same barrier point. Then this group of threads can proceed beyond that point.

The documentation of the CyclicBarrier Class can be found here:
http://docs.oracle.com/javase/7/docs/api/java/util/concurrent/CyclicBarrier.html

In this lab, you have to write your own cyclic barrier class using the techniques we have learned. Your program has to implement the following features.

1, Implement our own class MyCyclicBarrier, which can do the same things as built-in CyclicBarrier class does.

2, You have to implement the constructor,

public MyCyclicBarrier(int parties, Runnable barrierAction)
http://docs.oracle.com/javase/7/docs/api/java/util/concurrent/CyclicBarrier.html#CyclicBarrier(int, java.lang.Runnable)

3, You have to implement the await method,
public int await() throws InterruptedException, BrokenBarrierException
http://docs.oracle.com/javase/7/docs/api/java/util/concurrent/CyclicBarrier.html#await%28%29

In the await() method, you do not need to consider the reset() operation. But you are required to implement all Exceptions that are thrown.

Test Cases

Apply the cyclic barrier you implemented to the **problem solved in the demo**

UseCyclicBarrierDemo.zip. You should get the same barrier behavior. The output should look like the following.

```
Thread-0-->message output 0
Thread-1-->message output 0
Thread-2-->message output 0
All Thread Reached this Barrier Point!
Thread-2--> WAKE UP WITH ARRIVE RANK 0
Thread-0--> WAKE UP WITH ARRIVE RANK 2
Thread-1--> WAKE UP WITH ARRIVE RANK 1
Thread-0-->message output 1
Thread-1-->message output 1
Thread-2-->message output 1
Thread-0-->message output 2
Thread-1-->message output 2
Thread-2-->message output 2
Thread-2-->message output 3
Thread-0-->message output 3
Thread-1-->message output 3
Thread-0-->message output 4
Thread-1-->message output 4
Thread-2-->message output 4
Thread-2-->message output 5
Thread-1-->message output 5
Thread-0-->message output 5
All Thread Reached this Barrier Point!
Thread-0--> WAKE UP WITH ARRIVE RANK 0
Thread-2--> WAKE UP WITH ARRIVE RANK 2
Thread-1--> WAKE UP WITH ARRIVE RANK 1
Thread-2-->message output 6
Thread-0-->message output 6
Thread-1-->message output 6
Thread-0-->message output 7
Thread-1-->message output 7
Thread-2-->message output 7
Thread-1-->message output 8
Thread-0-->message output 8
Thread-2-->message output 8
Thread-1-->message output 9
Thread-0-->message output 9
Thread-2-->message output 9
Thread-1-->message output 10
Thread-0-->message output 10
Thread-2-->message output 10
All Thread Reached this Barrier Point!
Thread-2--> WAKE UP WITH ARRIVE RANK 0
Thread-1--> WAKE UP WITH ARRIVE RANK 2
Thread-0--> WAKE UP WITH ARRIVE RANK 1
Thread-2-->message output 11
Thread-0-->message output 11
Thread-1-->message output 11
Thread-0-->message output 12
Thread-1-->message output 12
Thread-2-->message output 12
Thread-0-->message output 13
Thread-2-->message output 13
Thread-1-->message output 13
Thread-0-->message output 14
Thread-2-->message output 14
Thread-1-->message output 14
```