**CS 4385 - Concurrency and Distributed Systems**

**Project: Dining Philosophers Problem**

**Authors: Jecsan Blanco and Mary (Rachel) Van Pelt**

**Report**: Your project report should discuss details of your logic/functions and how your solution has met/violated the constraints and correctness properties. Any assumptions made, problems encountered, sources referred should all be included in the report  
  
**Assumptions**:  
1. A valid parseable integer will be passed to the program.

2. Readme instructions will be read for program compilation and execution.

**The main programs**

1. **Main** – the main program for the algorithm 6.10
2. **MainV2** – the main program for algorithm 6.11

The main programs are both responsible for initializing all the philosophers and the forks used by the philosophers, the number of philosophers depends on the number passed via command line by default the number is set to 5. The main programs are identical besides one is using Philosopher for algorithm 6.11

**The Philosopher class (threads):**The program represents what a philosopher based on the constraints:

 Five philosophers either eat or think - 5 by default or more and have eat and think methods.  
 They must have two forks to eat - forced to wait for left and right forks  
 Can only use forks on either side of their plate - forced to used id and id + 1 forks (left and right)  
 Cannot forcefully obtain a fork (no preemption) – the fork class insure only one philosopher can pick it up.

1. **Philosopher** - an implementation of a philosopher using algorithm 6.10
2. **Phiosopher2** - an implementation of a philosopher using algorithm 6.11

The only difference between these two classes is that **Philosopher2** if it is the nth philosopher will  
try to pick up the left fork first instead of the right fork. Since most of the code is the same, we utilized class inheritance to only change what was needed on **Philosopher2**.

**The Forks class (semaphores):**

1. **Fork** – a renamed semaphore so we could use the same terminology as in the problem.

Used for synchronization and insurance of mutual exclusion between philosophers. This is just a renamed semaphore.

Problems encountered:

Correctness properties: