CS61A – Lab Assignment 5.2	**Kurt Meinz**
University of California, Berkeley	Summer 2002

In this lab there is no actual Scheme programming; you are to devise an algorithm in English to solve

The Dining Philosophers Problem

N philosophers are sitting around a round table for dinner. There is one chopstick between each pair of philosophers, for a total of N chopsticks altogether. You need two chopsticks to eat. Philosophers spend most of their time thinking, but every so often one gets hungry and wants to eat. A hungry philosopher must obtain the two chopsticks on his or her left and right. If one or both of those chopsticks is already in use, the philosopher must wait.

How can you use synchronization to solve that problem? You must come up with a solution that's

1. correct (so you don't have two philosophers using the same chopstick at the same time),

2. efficient (so you don't restrict eating more than necessary—for example, you shouldn't have only one person allowed to eat at a time),

3. not deadlocked (don't end up with each philosopher holding one chopstick), and

4. preferably fair (so they all get an equal chance to eat).

Be sure your solution works for both even and odd values of N.