Ex: The philosophers dining problem

```
public Philosopher(String philosopher,
  Object left, Object right) {
  this.name = philosopher;
  this.leftStick  = left;
  this.rightStick = right;

  this.thread = new Thread() {
    public void run() {
      try {
        synchronized(leftStick) {
          sleep(1);
          synchronized(rightStick) {
            sleep(1);
            System.out.println(
              "Philosopher " + name +
              ": Chew, chew, chew …" +
              "*burp*!");
    }}} catch (Exception ignore) {}
  }};
}
```

1) Discuss whether the provided solution to the philosophers dining problem can be fixed, using a lock ordering strategy, i.e. order by hash value or an explicit id.
2) Make a solution to the problem, using explicit locks (`java.util.concurrent.locks`).
3) (Optional) Describe a scenario in which the lock ordering algorithm in Listing 10.3 does not prevent a deadlock, if there is any.