1. Coding [24pts]

   An evil queen would like to host a ball, but she finds her own castle too humdrum. Instead, she would like to host the ball at the fabulous estate of Pemberley. Unfortunately, there are other evil queens who would like to host a ball at Pemberley, so she'll need to be certain the estate is not already in use when she hosts her ball. Additionally, she'll need food and music. Fortunately, there are five music collections available, so she can reserve one of those. Though initially there is no food, there is one chef available in the country who continually cooks party food. The guests at each ball consume all the food.

   Your job as the ball planner is to help the evil queen host her ball by helping her acquire the necessary resources—food, music, and, of course, Pemberley. Be certain that the evil queen does not end up sharing her ball with another queen! Nothing could be worse.

   (a) Solve the problem using semaphores. You'll need two functions: one to acquire and manage resources for the ball, and one for the chef to cook food.

   (b) Solve the party problem using monitors. Again, you'll need two functions.

   Be certain to follow coding conventions discussed in class. You may indicate the ball using <party> and the food creation using <cook>. A ball only requires one food unit. Clearly indicate your solutions to parts (a) and (b).

```
Semaphore Solution [10pts]:

Semaphore pemberley = 1
Semaphore food = 0
Semaphore music = 5

partyAtPemberley(){     cookFood(){
   music->Wait()           <cook>
   food->Wait()            food->Signal()
   pemberley->Wait()    }
   <party>
   pemberley->Signal()
   music->Signal()
}
```

Monitor Solution [14pts]:

```
class Party{
   Lock lock;
   condVar musicVar;
   condVar foodVar;
   condVar locVar;
   music = 5;
   food = 0;
   location = 1;

   partyAtPemberley(){
      lock->acquire()

      while(music <= 0)
         musicVar->Wait()
      music--

      while(food <= 0)
         foodVar->Wait()
      food--

      while(location <= 0)
         locVar->Wait()
      location--

      <party>

      music++
      musicVar->Signal()

      location++
      locVar->Signal()

      lock->release()
   }
```

```
cookFood(){
   lock->acquire()
   <cook>
   food++
   food->Signal()
   lock->release()
}
```