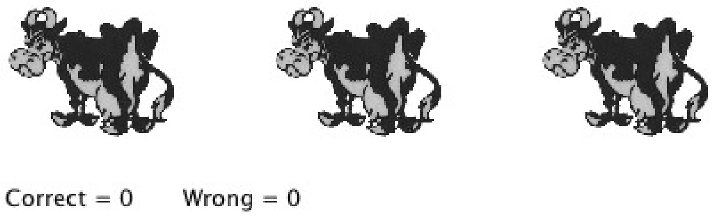
CS 134 Programming Exercise 4:
Three-cow Monty

Objective: To gain experience using loops and active objects.

The Scenario.

The goal of this program is to play a simple game called “Three-Card Monty” (sometimes also known as the Shell Game). Of course, this is Williams, so our version actually plays “Three-Cow Monty”. The program begins with three seemingly identical purple cows lined up in a row as shown below. However, one of the cows is different. If you click on the middle cow, it moos and you score a point for picking the correct cow. If you click on either other cow, they do not moo and your incorrect score increases by one. In itself, this is not much of the game, but if you click outside any cow the program shuffles their order in front of you. The goal is to follow the moving cows and click on the one that moos after they have been shuffled. The online version of this handout has a demo version of Three-cow Monty.



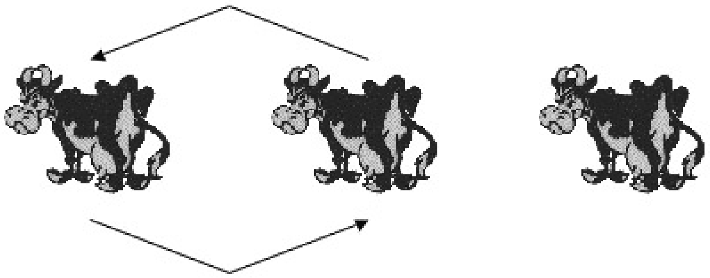
```
Correct = 0     Wrong = 0
```

Your program should be divided into two classes- the ThreeCowMonty window controller class and the Shuffler (or Moover!) active object class. The ThreeCowMonty class creates the three cows and handles the mouse interaction. The Shuffler moves the three cows, as illustrated in our demo. It selects two cows and gradually moves each of them from its original location to the opposite cow’s location. Once those two cows are repositioned, it randomly selects two new cows and moves them again, repeating this process perhaps 10 or 20 times before stopping.

The ThreeCowMonty class should prevent anything from happening if the user clicks while a Shuffler is shuffling the cows. In other words, the window controller should not test whether you click on the correct cow or create a second Shuffler if a Shuffler is currently running.

While it is more natural to describe the shuffler as randomly picking two cows to move, you may find it is easier to write the required code if you instead think of the shuffler as randomly picking one cow that it won’t move and then moving the two that were not picked. If the code for your `run` method begins to get complicated, remember that you can define `private` methods to perform common operations and then invoke these methods from `run`.

You should probably start by just moving the cows horizontally until the shuffling works. Implementing the shuffling in a way that shows up-and-down motion makes the game more entertaining, but it can be added after the rest of the program is working. The motions of the cows in the demo look like smooth curves, but they actually move in straight lines, as in the following picture:



We provide you with the cow picture and the mooing noise in the starter folder, but feel free to shuffle your choice of livestock, frogs, basketballs, or anything else.