CSC104 Project 1, Winter 2013
Due: Friday March 8th, 11:59 pm

You will complete two simulations, `ecosystem.rkt` and `ball.rkt`, described below. Your task is to download ecosystem.rkt and ball.rkt from the course website, under March 8th (right-click on them). Each of these files has seven comments indicating things you need to fix. The comments begin with three exclamation marks:

```
; !!! <some important work-needing instruction goes here>
```

Your job is to try to fix these, one-by-one, until you have a working simulation.

Below this you'll find descriptions of both simulations.

Edgy ecosystem explained

Imagine an ecosystem inhabited only by immortal mice and ravenous foxes. The mice need the foxes to prevent the mouse population from spiraling out of control, leaving the ecosystem several kilometers deep in squirming mice. The foxes need the mice to fill their bellies so that the foxes can live strong and reproduce.

This mutual dependence can be expressed (very crudely) symbolically, using the Lotka-Volterra %equation (see Wikipedia).

You create this ecosystem in `ecosystem.rkt`, and its behaviour is controlled through the values of some parameters:

MICE-START The initial number of mice.

FOXES-START The initial number of foxes.

MOUSE-BIRTH-RATE The number of mice born, per mouse, each year.

FOX-DEATH-RATE The number of foxes dying, per fox, each year.

FOX-MOUSE-EAT-BREED-RATE Rate of fox births as a proportion of the fox population times the mouse population.

FOX-MOUSE-MEAL-RATE Rate of mouse deaths as a proportion of the fox population times the mouse population.

Suppose you start the year with m mice and f foxes. The mice will give birth to MOUSE-BIRTH-RATE $\times$ m during the year and they never die of old age. If that were the only thing going on, we'd see an exponential growth in the mouse population. However the m mice and f foxes encounter each other, resulting in

FOX-MOUSE-MEAL-RATE $\times\, m \times f$ mice getting eaten each year. Putting this all together, the annual change in mouse population is:

$$m \times \text{MOUSE-BIRTH-RATE} - \text{FOX-MOUSE-MEAL-RATE} \times m \times f$$

Of course, the fox population is also affected by all this. Nobody's eating the foxes, but if we begin the year with f foxes, then FOX-DEATH-RATE $\times\, f$ foxes die each year of old age. Without any mouse snacks, this would lead to an exponential decay of the fox population, but luckily (for the foxes, at least) we can also assume that FOX-MOUSE-EAT-BREED-RATE $\times\, m \times f$ new foxes are born each year, as a result of good mouse-based nutrition. Better-fed foxes are better reproducers. Putting these together gives us the yearly change in the fox population:

$$\text{FOX-MOUSE-EAT-BREED-RATE} \times m \times f - \text{FOX-DEATH-RATE} \times f$$

You will build this idyllic fox-eat-mouse world in DrRacket. You'll get to experiment with different values for the parameters `FOX-DEATH-RATE`, `MOUSE-BIRTH-RATE`, `FOX-MOUSE-EAT-BREED-RATE` and `FOX-MOUSE-MEAL-RATE` to see what different settings mean to the long-term survival of these species.

Your formulae for updating the mouse and fox population should make sure that neither population is ever negative — nobody has ever observed fewer than zero mice or foxes. This can be done by taking the max of the calculated population and some minimum value, for example 1. Also animals occur in integer amounts (except, perhaps, in mid meal), so you'll want to round your population counts.

Bouncing octagonal ball

To represent a moving ball you need to represent its position, both the x coordinate (horizontal position) and y coordinate (vertical position), and its velocity dx (change in horizontal position) and dy change in vertical position. With these four numbers you can represent where the ball is now, and where it moves to at the next tick of the clock.

Now add bouncing off the boundary walls. To do this you need to be able to detect when the ball is beyond its left, right, top or bottom boundaries. If it's beyond the top or bottom boundaries, the change in vertical position, dy will switch — the ball switches direction from up to down, or vice-versa. If it's beyond the left or right boundary, the change in horizontal position, dx, will switch — the ball switches direction from left to right, or vice versa.

Those are all the pieces that go into representing a bouncing ball. You will implement these in DrRacket. As a possible 2% bonus, you may implement arrow key detection — `key-press` to change the rate at which position changes: faster down in response to the "down" arrow, faster up in response to the "up" arrow, etcetera.

What to hand in

You will submit the following files to MarkUs:

- `ball.rkt`
- `ecosystem.rkt`

You may work in groups of no more than 3 in preparing your project. To set up a trio or pair, one group member should log on to MarkUs and invite the other one or two. You should submit your files early and often. The first time you create a file with meaningful content, submit it. You may re-submit the same file as many times as you wish, and only the last submission is stored. A good habit is to re-submit your files each time you improve them.