Program #4 - Lucky Charms

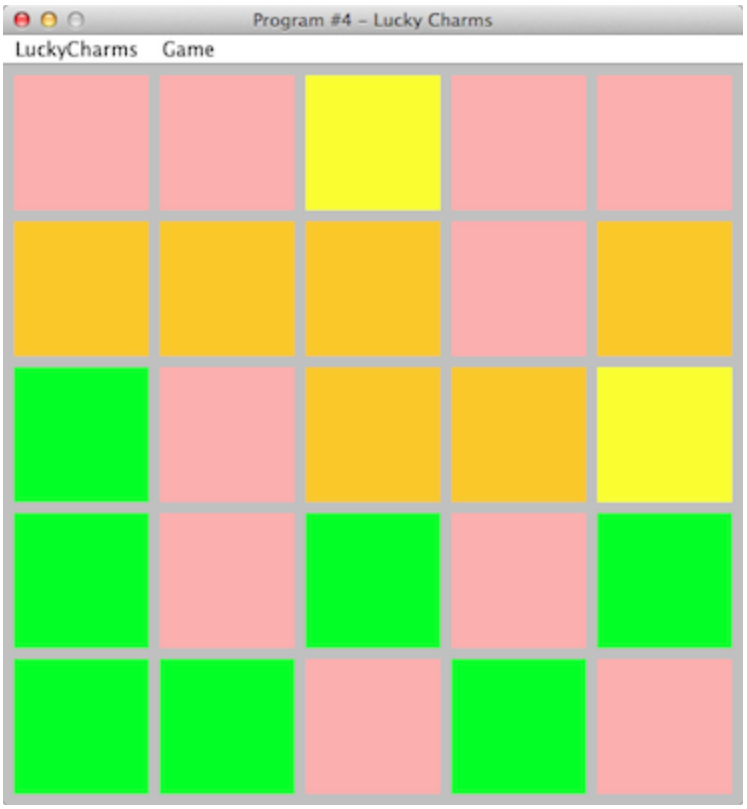
You're always after me Lucky Charms!

Logistics:

- Due: **Fri Mar 13, 2015**
- Worth: **10 points** (10% of your class grade)

1. Description

Program #4 is a GUI game where you gobble 3 (or more) connected Lucky Charms to make them disappear. The remaining charms drop to the bottom of the bowl, I mean, screen. Here's an initial screenshot:

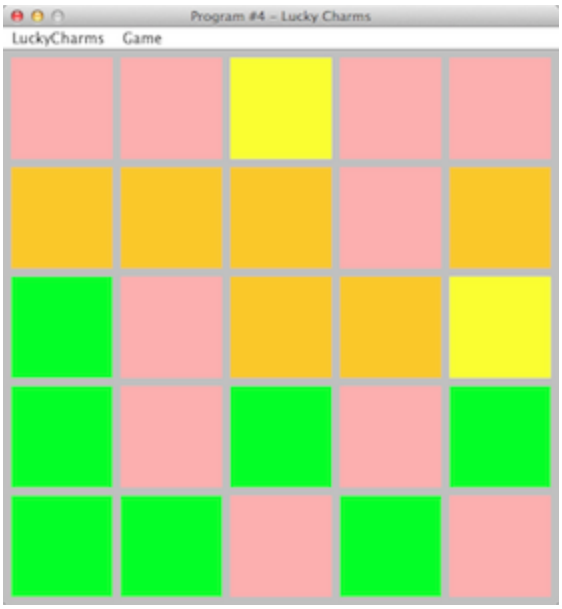


Your grid is 5x5. The color of each charm is randomly chosen from the original 4 Lucky Charms colors: green clovers, pink hearts, orange stars and yellow moons. (Source: mentalfloss.com/article/55538/50-year-history-lucky-charms-65-marbits)

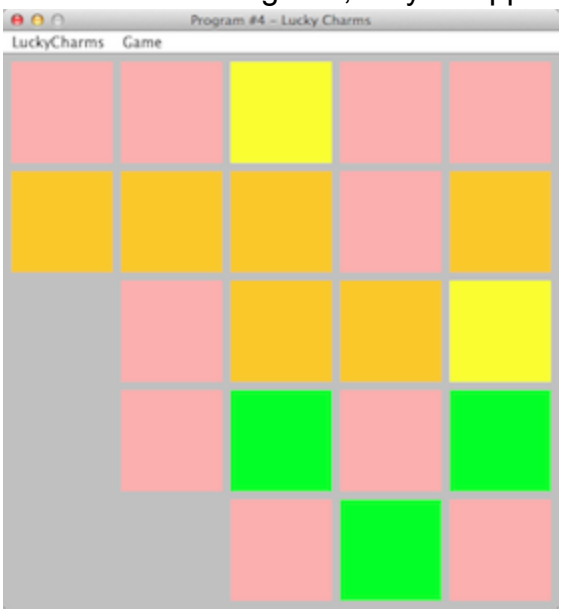
Playing - You select a square with the mouse. If 3 of the same color are connected, they are gobbled and disappear. Hit the spacebar to make gravity inch charms to the bottom.

Here's a play sequence.

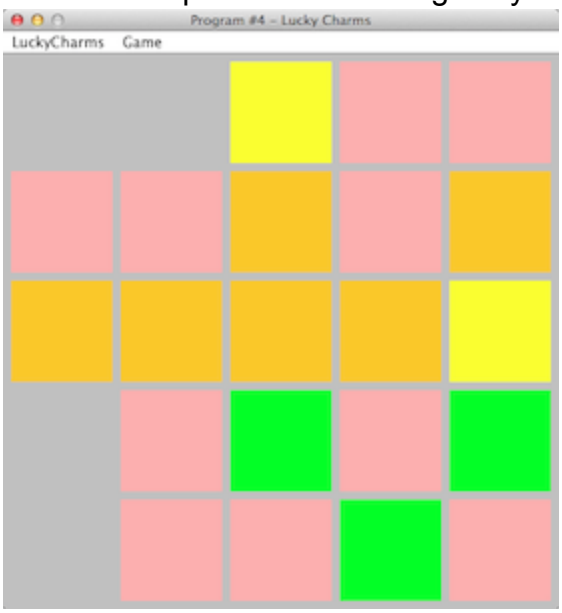
1. Start screen



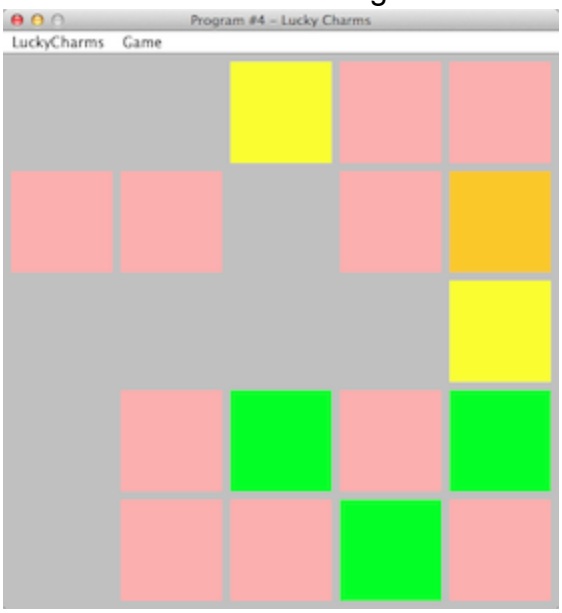
2. Select lower left green; they disappear



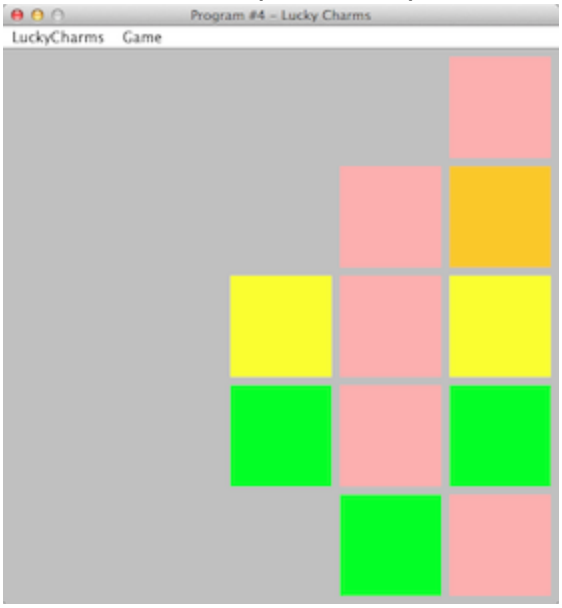
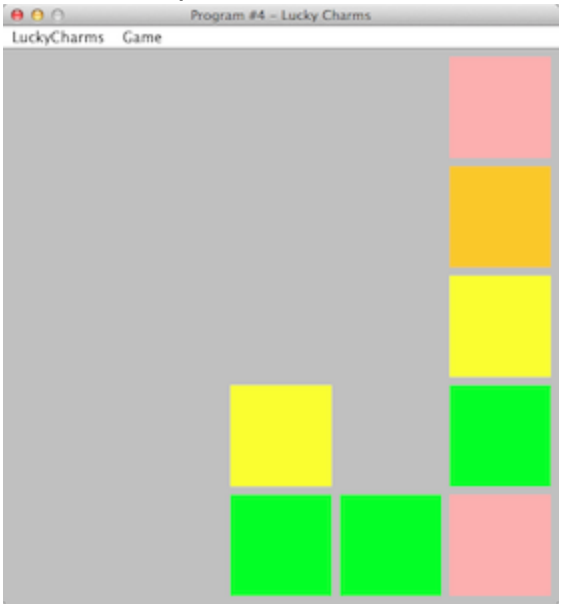
3. Hit spacebar once for gravity



4. Select a middle oranges charm



Squares are only connected up, down, left and right… not diagonally.
Eventually, you'll get to a point where there are no more color sequences left, and you're done. Your score is the number of charms gobbled.

5. Click, click, spacebar, spacebar	6. Click, spacebar and we're done
	

Our score in this example is 17 charms out of 25. Pretty good.
My menus are:

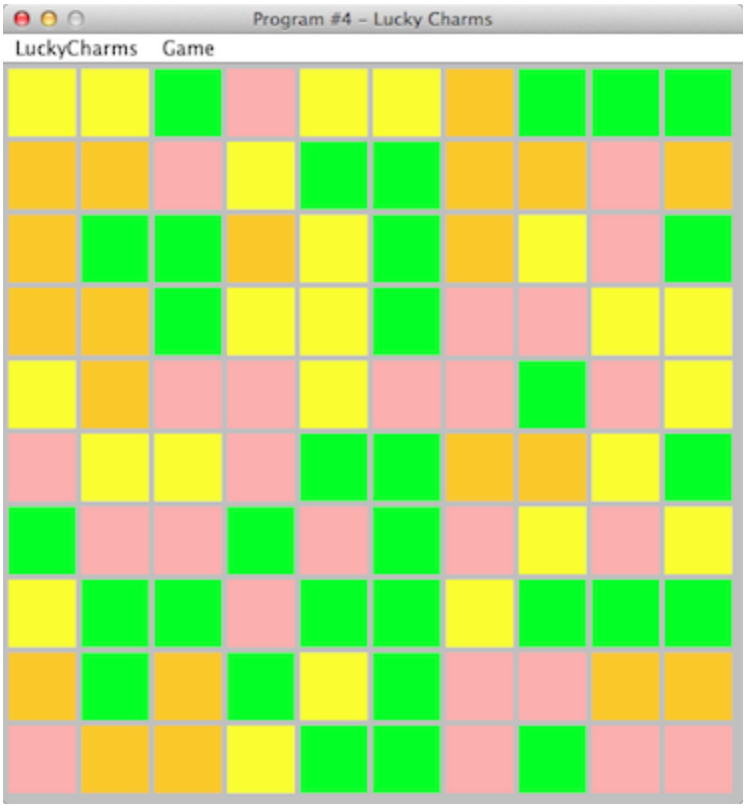
- LuckyCharm/About - standard author deal
- LuckyCharms/Exit - exit the program and dispose of your frame
- Game/Score - show the score
- Game/Reset - reset the lucky charm colors, creating a new game
- Game/2x - replace the 5x5 game with an incredible 10x10 game… 2X the fun!

Check out the (totally cool) 2X game pic on the next page. Yow!

2. Implementation

Here are some requirements of your Lucky Charms game:

- Well, first the bad news: You are prohibited from using JLabels or JButtons as charms. So, your `LuckyCharm` objects must draw themselves as filled rectangles on the `Graphics` object of a `JPanel`. This is a new paradigm for us, so look at page 880 in our textbook to get you going.
- Without fancy `Swing` objects, you have to implement selection yourself. Argh!
- Use a 2D array to store your `LuckyCharm` objects in a grid (a `LuckyGrid`)
- Use a `KeyListener` to do gravity (any key) and quit your game (escape key)
- Use an `ActionListener` to select charms with the mouse



A 2X game screen shot

2.1. Classes

I have 5 classes (in top-down order):

- `Program4` - main(); create the game and start it
- `LuckyCharmsGame` - has-a `JFrame` and has-a `LuckyPanel`; build menu bar
- `LuckyPanel` - is-a `JPanel` and has-a `LuckyGrid`; a `KeyListener` to catch gravity commands; an `ActionListener` to catch mouse selection of charms; override `paint()` method to draw the grid; call `repaint()` to display changes
- `LuckyGrid` - has-a `LuckyCharm`; this is where the 2D array of charms is built and maintained; `draw( Graphics)` method draws all the charms in the grid
- `LuckyCharm` - class vars are: (x, y) location, color, size, selected status and gobbled status; `draw( Graphics)` to draw one charm

We'll start up some design notes and talk (a lot) more about this over the next 2 weeks. Please note: Unlike Program #3, Lucky Charms will require a significant amount of your love and attention and cannot be completed in a day or two before the deadline.

3. Grading

Create a `program4` folder in your k: drive.
Place these files in that folder:

- A `README` file describing the state of your program.
- All the Java files that comprise your Program #4 solution

All your code must follow our 161 Coding Guidelines. Your code must be **beautiful**! Ugly code will be penalized with a 0-100% reduction in points. A program that doesn't even compile is worth 0 points.

Good luck with *your* Lucky Charms!
thanks… yow, bill