

CSC151.02 2014S, Class 11: A Design Perspective

Overview

- Preliminaries.
 - Admin.
 - Upcoming work.
 - Overview of the exam. (Policies and Problems).
 - A few notes on testing.
 - Questions.
- A bit about design.
- A bit about color theory.
- Exploring some images and design spaces.

Preliminaries

Admin

- Confidence survey: <http://bit.ly/csc151confidence>
- Reminder: Lots of resources to help you
 - Evening tutors
 - Weekly mentor sessions
 - Sam (email or office hours)
 - Individual or small group tutors
 - Let me know if you think you need one
 - But try the other resources, first
- Tomorrow's review session: 9:00 a.m.
 - Sorry for the inconvenience.
 - I'll also try to be around for questions at other times.
- Extra credit:
 - Convocation, Wednesday, 4:15, JRC 101 (Putin and Pussy Riot, Rescheduled)
 - CS Extras, Wednesday, 4:30 in Science 3821: AppDev
 - CS Table, Friday at noon in the Day PDR: NP Completeness

Upcoming work

- Study for Friday's quiz: Documentation, Testing, Side Effects.
- Writeup from yesterday's class due Friday.
- No writeup for today! (No lab, either.)
- Friday's readings: Raster Graphics, RGB Colors.
- Exam 1 prologue due Friday night. (optional)

- Exam 1 due Tuesday night (electronic).
- Exam 1 due Wednesday (printed).

About Exam 1

- Policies.
- Problems.

Some other notes.

Here's one group's set of tests, just for isosceles triangles

```
(test-case "unit side length isosceles triangle"
  (check-equal? (classify-triangle 1 2 2) "isosceles")
  (check-equal? (classify-triangle 2 1 2) "isosceles")
  (check-equal? (classify-triangle 2 2 1) "isosceles")
  (check-equal? (classify-triangle 1 3 3) "isosceles")
  (check-equal? (classify-triangle 3 1 3) "isosceles")
  (check-equal? (classify-triangle 3 3 1) "isosceles")
  (check-equal? (classify-triangle 1 4 4) "isosceles")
  (check-equal? (classify-triangle 4 1 4) "isosceles")
  (check-equal? (classify-triangle 4 4 1) "isosceles")
  (check-equal? (classify-triangle 1 5 5) "isosceles")
  (check-equal? (classify-triangle 5 1 5) "isosceles")
  (check-equal? (classify-triangle 5 5 1) "isosceles")
  (check-equal? (classify-triangle 1 6 6) "isosceles")
  (check-equal? (classify-triangle 6 1 6) "isosceles")
  (check-equal? (classify-triangle 6 6 1) "isosceles")
  (check-equal? (classify-triangle 1 7 7) "isosceles")
  (check-equal? (classify-triangle 7 1 7) "isosceles")
  (check-equal? (classify-triangle 7 7 1) "isosceles")
  (check-equal? (classify-triangle 1 8 8) "isosceles")
  (check-equal? (classify-triangle 8 1 8) "isosceles")
  (check-equal? (classify-triangle 8 8 1) "isosceles")
  (check-equal? (classify-triangle 1 9 9) "isosceles")
  (check-equal? (classify-triangle 9 1 9) "isosceles")
  (check-equal? (classify-triangle 9 9 1) "isosceles")
  (check-equal? (classify-triangle 1 10 10) "isosceles")
  (check-equal? (classify-triangle 10 1 10) "isosceles")
  (check-equal? (classify-triangle 10 10 1) "isosceles")
  (check-equal? (classify-triangle 2 3 3) "isosceles")
  (check-equal? (classify-triangle 3 2 3) "isosceles")
  (check-equal? (classify-triangle 3 3 2) "isosceles")
  (check-equal? (classify-triangle 2 4 4) "isosceles")
  (check-equal? (classify-triangle 4 2 4) "isosceles")
  (check-equal? (classify-triangle 4 4 2) "isosceles")
  (check-equal? (classify-triangle 2 5 5) "isosceles")
  (check-equal? (classify-triangle 5 2 5) "isosceles")
  (check-equal? (classify-triangle 5 5 2) "isosceles")
  (check-equal? (classify-triangle 2 6 6) "isosceles")
  (check-equal? (classify-triangle 6 2 6) "isosceles")
  (check-equal? (classify-triangle 6 6 2) "isosceles")
  (check-equal? (classify-triangle 2 7 7) "isosceles")
  (check-equal? (classify-triangle 7 2 7) "isosceles"))
```


[illegible]

```

(check-equal? (classify-triangle 5 5 8) "isosceles")
(check-equal? (classify-triangle 8 6 6) "isosceles")
(check-equal? (classify-triangle 6 8 6) "isosceles")
(check-equal? (classify-triangle 6 6 8) "isosceles")
(check-equal? (classify-triangle 8 7 7) "isosceles")
(check-equal? (classify-triangle 7 8 7) "isosceles")
(check-equal? (classify-triangle 7 7 8) "isosceles")
(check-equal? (classify-triangle 8 9 9) "isosceles")
(check-equal? (classify-triangle 9 8 9) "isosceles")
(check-equal? (classify-triangle 9 9 8) "isosceles")
(check-equal? (classify-triangle 8 10 10) "isosceles")
(check-equal? (classify-triangle 10 8 10) "isosceles")
(check-equal? (classify-triangle 10 10 8) "isosceles")
(check-equal? (classify-triangle 9 5 5) "isosceles")
(check-equal? (classify-triangle 5 9 5) "isosceles")
(check-equal? (classify-triangle 5 5 9) "isosceles")
(check-equal? (classify-triangle 9 6 6) "isosceles")
(check-equal? (classify-triangle 6 9 6) "isosceles")
(check-equal? (classify-triangle 6 6 9) "isosceles")
(check-equal? (classify-triangle 9 7 7) "isosceles")
(check-equal? (classify-triangle 7 9 7) "isosceles")
(check-equal? (classify-triangle 7 7 9) "isosceles")
(check-equal? (classify-triangle 9 8 8) "isosceles")
(check-equal? (classify-triangle 8 9 8) "isosceles")
(check-equal? (classify-triangle 8 8 9) "isosceles")
(check-equal? (classify-triangle 9 10 10) "isosceles")
(check-equal? (classify-triangle 10 9 10) "isosceles")
(check-equal? (classify-triangle 10 10 9) "isosceles")
(check-equal? (classify-triangle 10 6 6) "isosceles")
(check-equal? (classify-triangle 6 10 6) "isosceles")
(check-equal? (classify-triangle 6 6 10) "isosceles")
(check-equal? (classify-triangle 10 7 7) "isosceles")
(check-equal? (classify-triangle 7 10 7) "isosceles")
(check-equal? (classify-triangle 7 7 10) "isosceles")
(check-equal? (classify-triangle 10 8 8) "isosceles")
(check-equal? (classify-triangle 8 10 8) "isosceles")
(check-equal? (classify-triangle 8 8 10) "isosceles")
(check-equal? (classify-triangle 10 9 9) "isosceles")
(check-equal? (classify-triangle 9 10 9) "isosceles")
(check-equal? (classify-triangle 9 9 10) "isosceles")

```

What could that group have done better?

That group could have tried at least one non-integer side length.

That group could have noted that their code is insanely repetitious and written a helper like the following.

```

(define check-issoc
  (lambda (same diff)
    (check-equal? (classify-triangle same same diff) "isosceles")
    (check-equal? (classify-triangle same diff same) "isosceles")
    (check-equal? (classify-triangle diff same same) "isosceles")))

```

On a separate note, over the long term, you want your tests to not only tell you that there is an error, but to help you identify the error. So coming up with useful names for your tests is always a good idea.

Questions

A bit about design

- Our main goal is to learn CS.
- But we're going to try to make "interesting"/successful images
- Big goals:
 - Images that people want to look at for more than five seconds
 - Think about guiding the viewer's eye through the image
- First approach: Use what we know about color (or at least learn about color)

A bit about color theory

- Two basic models of colors
 - Additive - Color adds to white
 - Subtractive - Color "adds" to black
- Learn to think additively
- Two additive models
 - RGB - red, green, and blue components
 - Lots of each - bright/white
 - Little of each - dark/black
 - HSV - hue, saturation, and value
- See the pretty pictures linked from the outline.
- For us, the most important aspect of color theory is the sets of colors we'll put together
- What natural ways to group colors did you read about?
 - Complimentary - Across from each other on the color wheel - makes them more vibrant - can draw the attention of the viewer, make the colors stand out, etc.
 - Analogous - Three next to each other on the color wheel
 - Monochromatic - Multiple things on the same basic color,
- Other effects of colors
 - E.g., red affects perceived sexual attractiveness or success
 - E.g., blue is reputed to be calming
 - E.g., green pen makes you creative

Exploring some images and design spaces

Sam attempted to discuss images from a deck by Matt Kluber.

Samuel A. Rebelsky, rebelsky@grinnell.edu

Copyright (c) 2007-2014 Janet Davis, Samuel A. Rebelsky, and Jerod Weinman. (Selected materials are copyright by John David Stone or Henry Walker and are used with permission.)



This work is licensed under a Creative Commons Attribution 3.0 Unported License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nc/3.0/> or send a letter to Creative Commons, 543 Howard Street, 5th Floor, San Francisco, California, 94105, USA.