Slide 1

Administrivia

- Reminder: Homework 3 due today. (Accepted without penalty through 11:59pm tomorrow.)
- Quiz 2 Thursday. For topics refer to Administrivia slide from previous lecture.

Slide 2

Minute Essay From Last Lecture

- What was interesting/difficult — several people mentioned that the ideas were simple but turning them into (working!) code was not. Yes. Practice helps!

  One person remarked that for temperature conversion it's easier to use Google. True. But how do you think they do it?
- Reported times ranged from 15–20 minutes to three hours.

Slide 3

Functions in Scala — Review/Recap

- Functions in Scala are similar in spirit to functions in math, except that they can have side effects (such as printing something).
- Several reasons for using functions (reduce code duplication, allow code reuse, “manage complexity”).
- Define functions with syntax starting with `def`. Use by giving name, parameters.

Slide 4

Function Literals and Higher-Order Functions

- Scala lets you define “literals” for types such as `Int` and `String`. It also lets you define literals for functions. That may seem like a strange thing to do, but . . .
- It also supports “higher-order functions” — functions whose parameters are themselves functions. An example from math is function composition. We will see uses for this in programming later.

Slide 5

Example — Change-Counting Program Revisited

- As another example of using functions to improve(?) a program, look again at the program to count change and try to make it print its output more nicely.

Slide 6

Example — Calculating Bounding Boxes

- Something that's of interest in graphics programs is finding the smallest rectangle that encloses one or more other shapes — a "bounding box".
- Let's try writing a function that computes a bounding box for two rectangles. We'll represent rectangles as tuples of four `Ints` — the x and y coordinates of the top left corner and the width and height.
- (Tuples were discussed in chapter 3. Briefly, tuples in Scala are similar to tuples in math — ordered lists of elements, of fixed size. The syntax for making one in Scala is to enclose one or more values in parentheses.)

Repetition and Recursion — Preview

Slide 7

- Having if/else allows us to do a lot of things we couldn't do before, but there are still things we can't do easily, mostly involving some sort of repetition. Simple example — adding something to the grade program that would prompt for six quiz scores. Another example might be trying to use our bounding-box function to find a bounding box to enclose more than two rectangles, with the choice of how many up to the user.
- Scala provides many ways to do this. We will look at recursion first. (To be continued next time.)

Minute Essay

Slide 8

- Advice I heard a while back for learning new programming languages (not very tactful but true): You tend to make the same mistakes over and over, so it can help to mentally collect them into a "personal bozo list". What would be on your list at this point?