Functions

9-19-2011

Opening Discussion

- Minute essay comments
  - Bitwise operators
  - When reading is going over your head.
    - Unclear sections.
    - Potential modifications.
  - When to read. Most like after class, but some before.
  - Arrays in 2 weeks.
  - Scientific notation in Scala.
  - Letters and words in binary.
  - Skating for the novice.

Functions in Math

- Let's review the concept of functions from math.
- In algebra a function would take one or more values and give you back a value. The values were generally numbers.
- In higher level math this is generalized with things like sets.
- In math functions the same input always leads to the same result.

Functions in Programming

- The concept of a function is critically important to programming.
- Functions can take one or more arguments and give us back values. (Most languages allow only one return value.)
- Let's think of some examples of functions that we could write.

Functions in Scala

- We declare functions in Scala using def. Here is the general form.
  - def *name*(*arg1*:*Type1*, *arg2*:*Type2*, ...):*Type* = e*xpression*
- The argument list can have zero or more elements. If there are zero even the parentheses can be left off.
- Function arguments must have types.
- The return type is optional, but it is recommended.

Why Functions?

- Functions are used in programs for a number of reasons.
  - Reduce code duplication. You can call the same function multiple times and only write it once.
  - Improve readability and maintainability. Good function names make it easier to read. Small functions are easier to test and debug.
  - Break problems down/problem decomposition.

Problem Decomposition

- Never solve a hard problem. If a problem is hard, break it into smaller problems that are easier. Repeat until you are only solving trivial problems.
- Top-down
  - This is the “normal” approach where you start with the full problem and break it into pieces.
- Bottom-up
  - Sometimes you realize that different trivial pieces will be useful and build up from those.

Function Literals

- Just like 5 is a literal for an Int and “hi” is a literal for a string, you can write literals of functions.
- The full syntax is an argument list followed by an equals arrow followed by the function expression.
  - (a:Int,b:Int) => 3*a+2*b
- Types don't have to be specified in many situations, only if Scala can't figure it out.

Higher-Order Functions

- These are functions that take functions as arguments or return functions.
- These are the main things we use function literals for. We will see them a lot in two weeks.

Minute Essay

- What questions do you have about functions?
- Do you feel that end of section exercises would be helpful for the book?
- IcP #2 is next class.