Opening Discussion

- Minute essay comments:
  - In-class code and quizzes.
  - Programming competition volunteers.
  - Sort and search vs. just search.
  - Sorting other types of things.
  - Is the pie a lie?
Classification of Bugs

- We classify the errors that occur in programs in three broad groups.
  - Compile Errors – Found by the compiler. Gets a reasonable error message and line number.
  - Runtime Errors – Program crashes while running for a particular input. Gives type of error and line.
  - Logic Errors – Code runs fine, but does wrong thing. No information given to help you.

- You want to have your errors be higher up on this list because it gives you more information and makes it easier to fix.
Motivation

- We have been using “flat” text files to store things.
- Advantage: it is human readable and simple.
- Disadvantages: everything else.
  - Slow
  - Large
  - Lacks meaning
  - Hard to edit
  - Hard to debug
The eXtensible Markup Language (XML) is a standard for text encoding of data.

If you have ever done HTML, XML is similar. XHTML is HTML that follows the XML standard.

The advantage of XML is that it can encode pretty much anything and it is human readable text.

The downside is that it can be very verbose.

Composed of markup (between < and > or & and ;) or content (anything not markup).
The primary markup used in XML is the tag.
A tag begins with a `<` and ends with a `>`. 
There are three types of tags.
- Start-tag: `<student>`
- End-tag: `</student>`
- Empty-element tag: `<quiz/>`
The structure of XML documents comes primarily from elements.

An element is one of the following:

- Everything from a start-tag to the matching end-tag.
- An empty-element tag.

Elements have to be properly nested. The nesting can imply information.
Attributes

- An attribute is a name value pair.
- They can be put in start-tags or empty-element tags.
- Examples:
  - <student name="Jason" id="0123456">
  - <quiz grade="55"/>
An XML file can begin with a declaration telling information about it.

- `<xml version="1.0" encoding="UTF-8" ?>`

We won't worry about these in this class.
The Scala language supports XML at the language level.

Go to the REPL and enter some XML.

There is a scala.xml package that contains the libraries for XML.

- The NodeSeq, Node, and Elem types are particularly useful. I'll typically just use the word Node to describe something from the XML.

- So is the XML object.
The XML Object

- The loadFile method can be passed a file name and it will read in the file and return a NodeSeq that allows you to get to the contents.
- There is also a save method that takes a file name and an XML node and writes it to file.
Using \ and \\

- Use the \ operation on a node to search for the occurrences of something at the top level.
- The second argument is a string.
  - Normal string searches for tags with that label.
  - If the string starts with @ it searches for attributes.
- Use \\ to search deeply.
What questions do you have about XML?