What is Advanced Programming?

- Programs have two purposes:
  - To instruct a computer
    - So the program must be executable
  - To communicate with people
    - So the program must be an object of study
      - for tools, and
      - for programmers

- Many students never learn the second purpose!
  - So we are going to focus on it
Key Ideas in Programming

• Whole Object
  • The reason to use high-level data structures is to manipulate them at a high level

• Composition
  • Build complex objects and processes from simpler parts; apply this idea recursively

• Recursive Data Structures and their operations

• First class functions

• First class continuations

• Parameterization and Inheritance
Key Tools

- Test-driven Development (TDD)
  - SUnit, HUnit and Quickcheck
- Refactoring
  - Embrace change!
  - Don’t plan for generality; you ain’t gona need it
- Profiling
  - First get it right, and then make it fast
Strive for Simplicity

What Is Simplest?

So, the definition of the best design is the simplest design that runs all the test cases. The effectiveness of this definition turns on: what do we mean by simplest?

Is the simplest design the one with the fewest classes? This would lead to objects that were too big to be effective. Is the simplest design the one with the fewest methods? This would lead to big methods and duplication. Is the simplest design the one with the fewest lines of code? This would lead to compression for compression’s sake and a loss of communication.

Here is what I mean by simplest—four constraints, in priority order.

1. The system (code and tests together) must communicate everything you want to communicate.
2. The system must contain no duplicate code. (1 and 2 together constitute the Once and Only Once rule).
3. The system should have the fewest possible classes.
4. The system should have the fewest possible methods.
What about the Programming Language?

- We believe that unnecessary complexity in the language gets in the way of writing elegant, simple, programs

- We will be using two different languages, both rather small and elegant:
  - Haskell — pure, functional
  - Smalltalk — effectful, object-oriented

- Most programming techniques will be applicable to both languages!
Course Organization

- This is an experimental course (eXtreme Teaching)
  - *Don’t* expect to find everything well-prepared far in advance
  - *Do* expect us to respond to feedback as the course evolves

- Rapid Feedback is important
  - Small assignments every week, sometimes every class meeting

- We emphasize code that “speaks” to us
  - Come to class prepared to talk about your code
  - Be open to criticism: almost all code can be improved upon
Your Instructors

- The youthful Professor Sheard
  - email sheard, ‘phone 503 725 2410

- The aging Professor Black
  - email black, ‘phone 503 725 2411