CIS 552
Advanced Programming
Fall 2013
What is Advanced Programming?

• Good programmers get the job done

• Excellent programmers
  – write code that other people can understand, maintain and modify
  – rewrite/refactor code to make it clear
  – design *abstractions* to make it reusable
"There are two ways of constructing a software design:
One way is to make it so simple that there are obviously no deficiencies,
and the other way is to make it so complicated that there are no obvious deficiencies.
The first method is far more difficult."

Tony Hoare
Turing Award Lecture 1980
Goal: Obviously no deficiencies

• Want code that is so simple, it obviously works

simple code is

Readable
Reusable
Modifiable
Predictable
Checkable

• What makes code simple?
Absence of Unnecessary Elements

• No Mutation.
• No Objects.
• No Loops.
• No IO.

Focus on what code **means** instead of what it does
Functional Programming

- Readable
- Reusable
- Modifiable
- Predictable
- Checkable

- Simplicity through Purity
So, who uses FP?

• PL Educators.
So, who uses FP?

- PL Researchers.
So, who uses FP?

• Industry.
Why Haskell?

• Bleeding edge technology.
Why Haskell?

• Beautiful.
Why Haskell?

• Blows Your Mind.
Why Haskell?

• Fun.
Why Haskell?

• Types.
Why Haskell?

• Real.

This plot shows how many times slower the fastest benchmark programs for selected language implementations were, compared to the fastest programs.
Plan for the semester

Functional Programming
  – Black-belt Haskell
  – Many small-scale case studies
  – In-class design analysis

Advanced Programming Techniques
  – Modular design and abstraction
  – Black-belt Types
  – Test driven development
  – Debugging
  – Collaboration (pair programming)
What this course is not

• CIS 350/573, Software Engineering
  – Focuses on "Software in the large"
  – How to deal with code you didn’t write
  – Problems that arise in projects that are too large for one person
    • lifecycle models
    • project management
    • design modeling notations (UML)
    • formal specification

• Both courses complement each other
Audience

- People with strong background in programming and mathematics
- No experience with FP expected
- Priority to CIS undergrads/grad students
- If not registered, send email and come to my office (Levine 510) TOMORROW
Course Staff

Instructor: Dr. Stephanie Weirich
sweirich@cis.upenn.edu

TAs:
Richard Eisenberg
Antal Spector-Zabusky

Fill-in instructor: Chris Casinghino
ccasin@cis.upenn.edu
Grading

• 60% Programming assignments
  – mostly in pairs
  – graded on correctness, style and (asymptotic) efficiency

• 20 % Final Projects (your choice)

• 20 % Class participation
  – questions – say your name!
  – HW debriefings
  – Piazza participation
  – Read a good blog post about FP? post about it on Piazza!
Homework #0

• Available from course website
• Get everything set up on your machine
  – Haskell compiler and libraries
  – Version control software
  – Editor
• Download first lecture, fill in blanks and run
• Submit via course website
• Due one week from today, at midnight
Where to go for help

• Course website:
  – lecture code, homework, resources
• Textbook: Real World Haskell (free online)
  – start reading first four chapters
• Class forum: piazza.com
• Office hours
  – Open office TOMORROW (Thurs, Aug 29)
Homework #1

• Will be available next Wednesday
• Download/submit from course website
• Work with a partner, submit only once
• Must compile with –Wall –Werror to get any credit.
• Late policy
  – 10 point penalty for up to 24 hours late
  – 20 point penalty for up to 48 hours late
  – no credit for assignments submitted after 48 hours
Homework style

• Style guide on course website, read it
• Interactive HW discussions in class
  – Examples of good/bad style
• Revise, revise, revise
  – Passing all the tests is not enough
  – Code must be effective technical communication
  – How can I make this code cleaner, more general, more clear about what it is doing?
Academic Integrity

• We expect you to follow Penn's policies on Academic Integrity.

• Do not plagiarize or copy code
  – Using a library function is ok
  – Looking up the source of that function and presenting it as your own work is not

• Do not get someone else to do your work for you
  – Ok to ask for help debugging a type error
  – Not ok to ask for the answer
  – Ask high-level questions on piazza
Lecture format

• Last powerpoint slide of the semester
• Interactive code demos
• Bring your laptops and work along
• Ask questions!