Myriad Aspects of Computer Science

- Component-based problem solving
  Building solutions to complex problems by “glueing” together solutions to smaller problems
- Data structures and operations on them
  stacks, queues, lists, trees, . . .
- Recognition and reuse of useful computational patterns
- Capabilities and limitations of computation
- Languages, environments, and tools
- Analysis of different approaches and understanding tradeoffs

We’ll be exploring many of these items using Haskell.

Why Haskell?

- Strong support for working with patterns
  - patterns of data
  - computational patterns
- Well suited for logical analysis and assurance arguments
- Useful for rapid prototyping of ideas
- Future of computing (e.g., concurrency and multi-cores) depends on higher-levels of abstraction
- Weight lifting for your mind!
  You have no choice but to grapple with types, pattern matching, and recursion.

Don’t be Fooled: This is Not a Standard Intro Course

- The only way to learn programming is to practice.
  - Start early, and ask for help if needed.
- You need to keep up to date with the material.
  - Later topics build on previous topics.
  - If you wait to work on assignments, the intervening lectures likely won’t make much sense.
- We may seem to start slow, but we’ll ramp up within a couple weeks.
  Develop good habits early!
Labs start tomorrow!

Office hours: Mon (3-5pm), Fri (1:30-3pm), appointment (email!)

Course web site:
http://www.cis.syr.edu/courses/cis252

I generally use Blackboard only for grades.

Grades are based on:
- Labs (12%): every week (except exam weeks)
- Homeworks (24%): every week (except exam weeks)
- Exams (58%): three in-class exams, one optional final
- Pop quizzes and in-class activities (6%)

It’s essential to keep up to date: later topics build on previous topics.

Start assignments early: late assignments are not accepted.

Don’t be shy in asking for help: that’s why we’re here!