CIS 552
Advanced Programming

Fall 2021

Welcome!

- Sit anywhere
- Make a name placard
- Introduce yourself to your table

Course Staff

Instructor: Dr. Stephanie Weirich

sweirich@seas.upenn.edu

OH: *Today after class*

TAs: Nicholas Rioux, Harrison Goldstein

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 and simple
  - use and create *abstractions* to capture fundamental designs

Tony Hoare
Turing Award Lecture 1980

"There are two ways of constructing a software design:

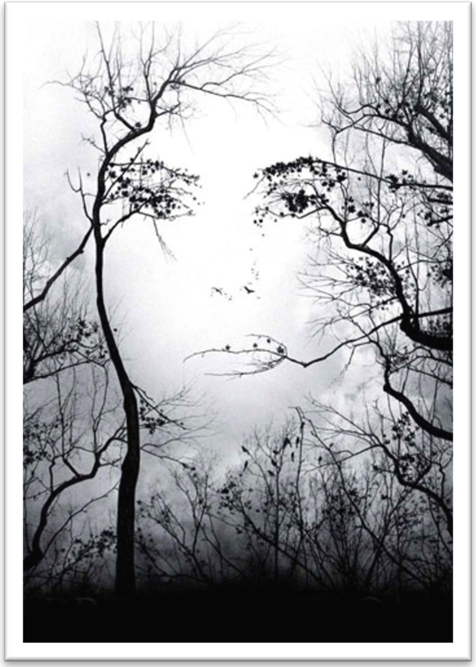
One way is to make it so simple that there are obviously no deficiencies,

and the other way is to make it so complicated that there are no obvious deficiencies.

The first method is far more difficult."

Simplicity through Abstraction

- Readable
- Reusable
- Modifiable
- Predictable
- Checkable



- Advanced type systems: Multiple levels of abstraction available

Simplicity through Purity

- Readable
- Reusable
- Modifiable
- Predictable
- Checkable

- Functional Programming: Focus on what code **means** instead of what it does
- Programming inspired by pure mathematics

CIS 552

Haskell

Why Haskell?

- Leading edge language research.

Why Haskell?

- Beautiful mathematics.

Why Haskell?

- Stretches Your Mind.

Why Haskell?

- Fun.

Course content

Functional Programming

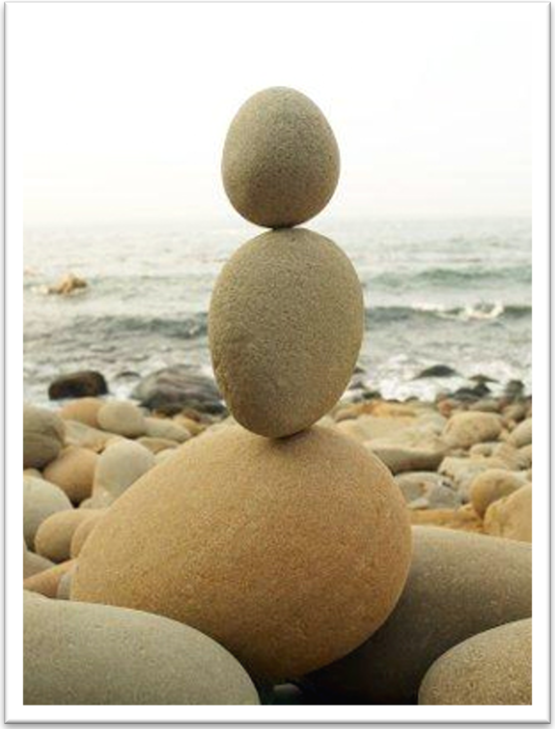
- Black-belt Haskell
- Mathematical approach to programming
- Many small-scale case studies

Advanced Programming Techniques

- Modular design and abstraction
- Black-belt types
- Test driven development
- Collaboration (pair programming)

Lots of programming!

- Small in-class exercises
- Bi-weekly homework assignments
- End of semester project



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
- The two courses complement each other

Audience

- People with strong background in programming and mathematics
- No experience with FP expected, but helps
  - We'll split the class by background for the first few weeks, but will converge quickly
- Undergraduates, Masters, and PhD students together

How will this all work?

Course Structure

- 30 % Active learning / participation
  - pre-class quizzes (study "lecture" content asynchronously!)
  - in class exercises, in class discussion
  - office hours – let's chat!
- 50 % Programming assignments
  - in pairs, mostly *randomly* assigned
  - graded on correctness, style and (asymptotic) efficiency
  - first assignment available now
- 20 % Final Projects (your choice)

Because of the active learning component, in person participation is essential!

Asynchronous "Lecture" Content

- Course content available in two forms
  - Formatted reading: on the *public* course website (under "Schedule")
  - IDE experimentation (*recommended*): public repo in github
- Read module "Basics" before next class
  - Part of the "01-intro" project on github
  - Fill in the "undefined" parts in your IDE
- Canvas quiz on material due before next class (10 AM)
  - Make sure that you ask a question to guide our next in-class discussion

Active Learning Goals

- Goal for the semester: create a CIS 552 *community*
  - You should get to know me and the TAs (they're great!)
  - You should get to know each other (you are all great!)
- Forced, random interactions during synchronous class time
  - (Quizzes on asynchronous content due before start of class)
  - Small and large group discussions
  - In-class exercises with a partner or table
  - TODAY: PL-themed icebreaker game

Fall 2021 and COVID-19

- Masks are **required** for everyone.
- Don't come to class if you have red pass. Send me a CAR.
- Contingency plans
  - All classes will be recorded and posted to Canvas (my laptop screen, plus room camera). There are six microphones in the ceiling.
  - Backup recording: Zoom call (also my laptop screen) & lavalier mic. Zoom link available from Canvas, but I can't see the Zoom chat.
  - This is the tentative plan and error prone.
  - Remote participation (maybe?) PollEverywhere, VS Code live share

Homework #1

- Based on "Basics" module
- You will be provided with a *private* repo to complete the assignment
- Work alone or with a partner (your choice), only one person should submit
- Must compile to get any credit
- Due Sunday, Sept 12th at midnight
- Late policy (all homework assignments)
  - 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
  - if you are affected by COVID, please ask for an extension

Where to go for what

- Public site (http://www.seas.upenn.edu/~cis552)
  - Haskell related asynchronous lecture material, HW instructions
- Github organization (https://github.com/cis552)
  - Code repos for lecture content, in-class exercises (public) & HW (private)
- Canvas site (https://canvas.upenn.edu/courses/1606916)
  - Syllabus, Zoom links, lecture recordings, quizzes, grades
  - Link to Piazza (Announcements and questions)
  - Link to Gradescope (Homework submission)

Things to do right now

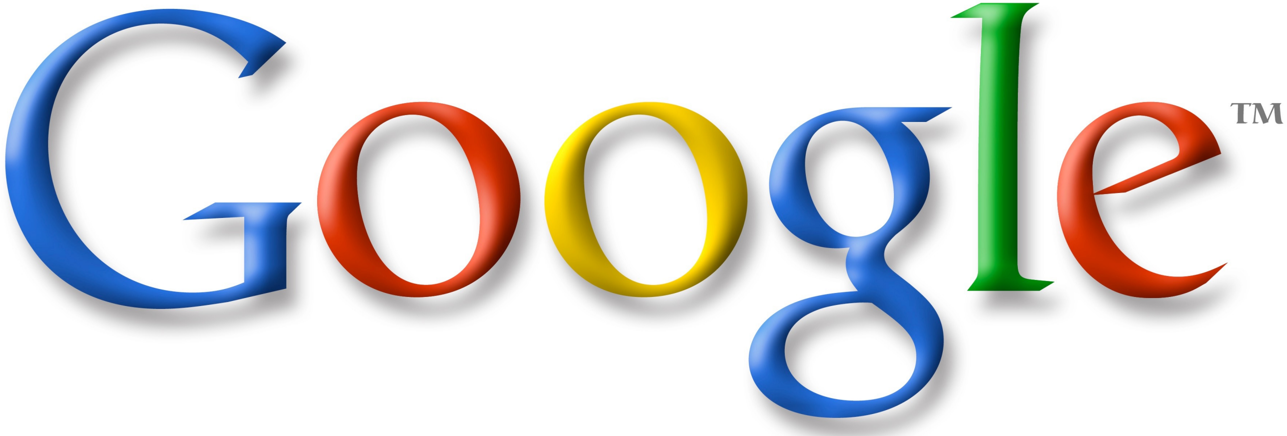
- Respond to Fall 2021 survey (if you haven't already)
- Create a github account (if you do not have one) and tell us your github id (link to form on Canvas)
- Introduce yourself to the others at your table, make a name card
- (After class) Start reading "Basics" module, install software (stack, VSCode), watch for access to your hw01 repo

https://pl-quiz.herokuapp.com/

- Each table is a team and should choose a team name
- **One** person should enter the answers for the whole team
- Don't refresh or leave the page or you will lose your score
- Winner is the team with the most points by 1:20PM

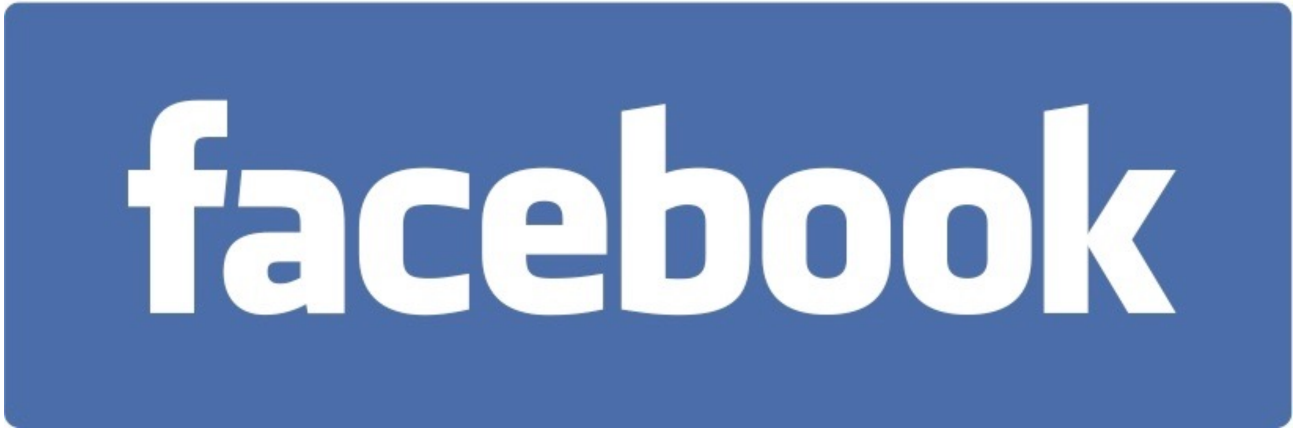
fin

So, Who Uses FP?



So, Who Uses FP?

So, who uses FP?



So, Who Uses FP?

So, Who uses FP?

JANE STREET

BARCLAYS

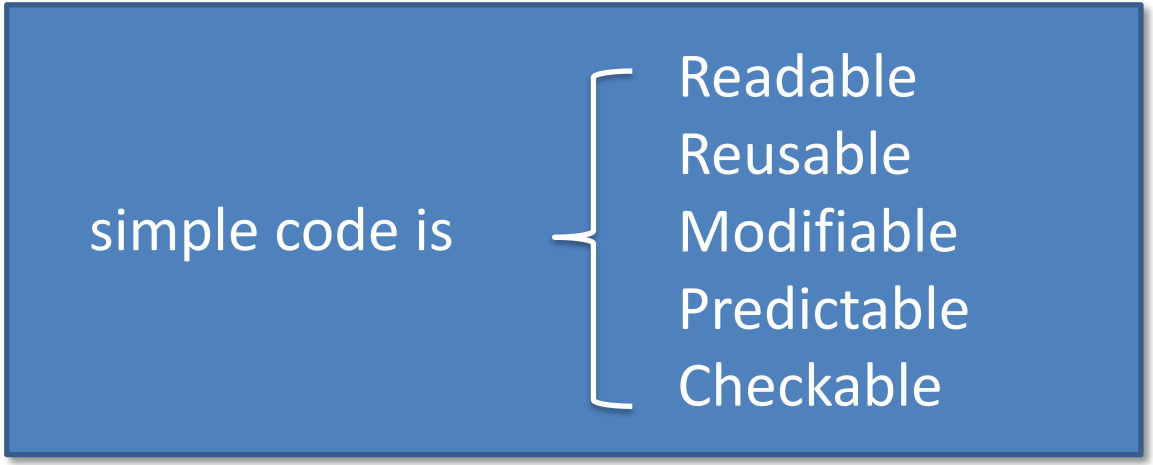
Standard Chartered

CREDIT SUISSE

Bank of America Merrill Lynch

Goal: Obviously no deficiencies

- Want code that is so simple, it obviously works



simple code is

- Readable
- Reusable
- Modifiable
- Predictable
- Checkable

- OK… so what makes code simple?