

# CIS 5520

# Advanced Programming

Fall 2023

Welcome!

- Sit at any table
- Sign in
- Make a name tag
- Introduce yourself to your table
- Pick a team name



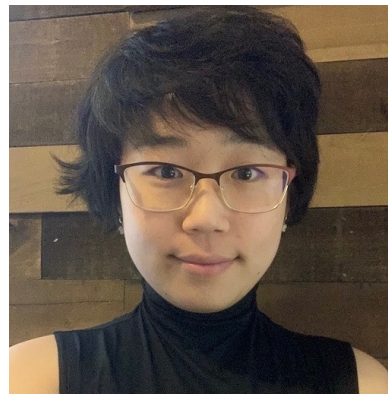
# Course Staff

Instructor: Dr. Stephanie Weirich

*[sweirich@seas.upenn.edu](mailto:sweirich@seas.upenn.edu)*

*OH: Tuesdays, 2-3:30pm, Levine 510*

TAs: Jonathan Chan, Cassia Torczon, Irene Yoon

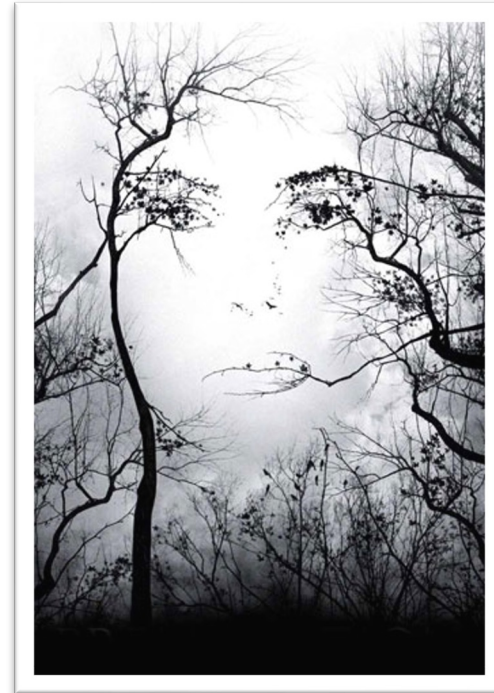


# What is Advanced Programming?

- **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
  - can explain *semantics* precisely: what their code does and why

# Simplicity through Abstraction

- Readable
  - Reusable
  - Modifiable
  - Predictable
  - Checkable
- 
- Advanced type systems:  
Multiple levels of  
abstraction available



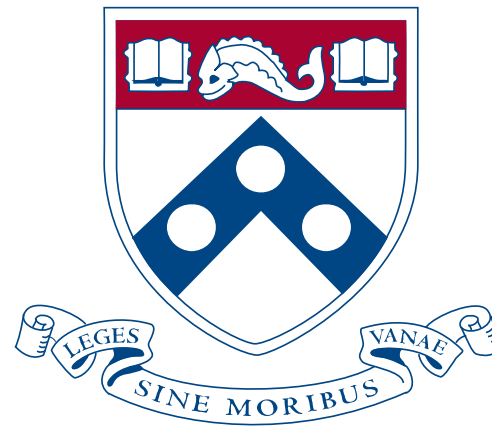
# Simplicity through Purity

- Readable
- Reusable
- Modifiable
- Predictable
- Checkable



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

# CIS 5520



# Course content

## Functional Programming

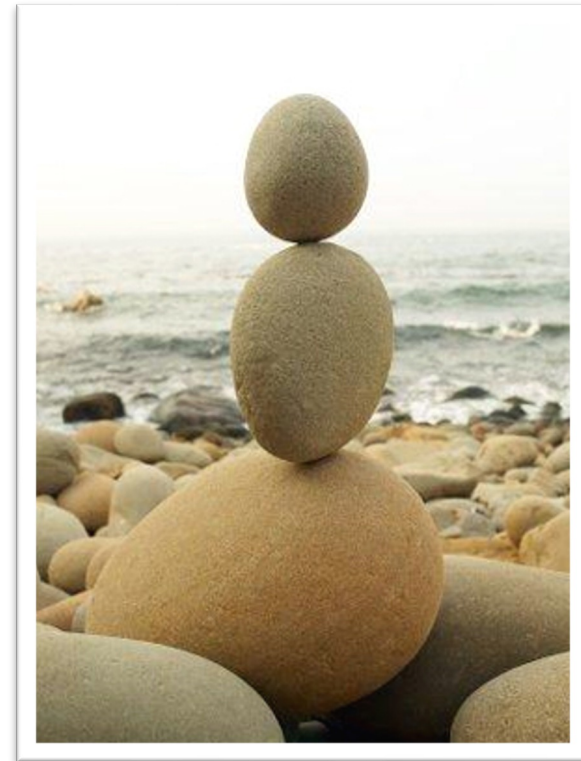
- Black-belt Haskell
- Mathematical approach to programming
- Focus on semantics and semantic types
- Many small-scale case studies

## Advanced Techniques

- Modular design and abstraction
- How to make types work for you
- Test and property 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 3500/5730, 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

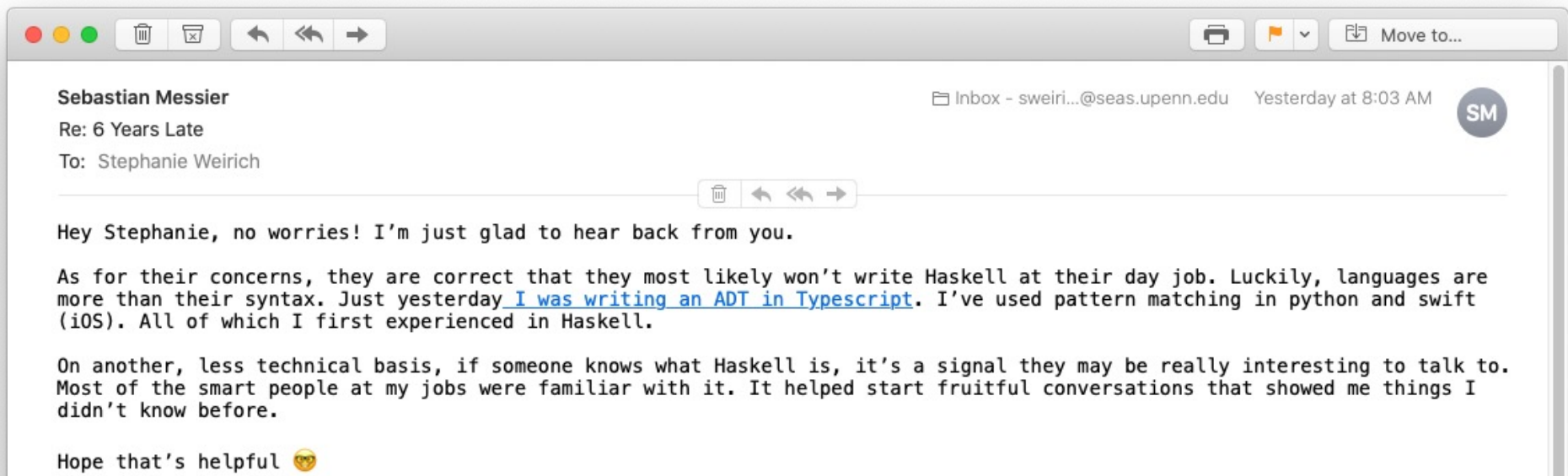


# What are you most excited about for CIS 5520?

- Functional Programming
- Haskell
- Monads and relationship to mathematics
- Learning to be a better programmer
- Learning a different way of thinking and programming
- Learn some real stuff that can be useful in my career
- Hands-on learning
- Homework and projects

# What concerns do you have about CIS 5520?

- Can I take the course with no experience in FP?
- Will my prior experience trip me up?
- Will it be too much work, especially with my other classes?
- Is it practical? Will it benefit future jobs?
- Will it be too fun?
- Will I get lost with monads?



**Sebastian Messier**

Re: 6 Years Late

To: Stephanie Weirich

Inbox - sweiri...@seas.upenn.edu Yesterday at 8:03 AM

SM

Hey Stephanie, no worries! I'm just glad to hear back from you.












As for their concerns, they are correct that they most likely won't write Haskell at their day job. Luckily, languages are more than their syntax. Just yesterday [I was writing an ADT in Typescript](#). I've used pattern matching in python and swift (iOS). All of which I first experienced in Haskell.


On another, less technical basis, if someone knows what Haskell is, it's a signal they may be really interesting to talk to. Most of the smart people at my jobs were familiar with it. It helped start fruitful conversations that showed me things I didn't know before.




Hope that's helpful 😊

Phil Eaton on X: "So 9 (but mor... x" Thinking about functional prog... x +

twitter.com/eatonphil/status/1695839314611974427

 **Post**

 **Phil Eaton**   
@eatonphil 

So 9 (but more like 11) revisions later, finally got a working (in-memory) BTree in Python that passes a few key tests (inserts decreasing, increasing, and random work correctly).

A chump, I could only get it working when I threw out all mutations and did the whole thing immutably.

However, this was a super useful way to get a correct implementation. I was banging my head for a few days while I was messing up mutation while recursing. And I think just having this correct immutable version is going to make redoing a mutable version simpler.

Alternatively, I could keep it "immutable" and make a pool of unused nodes so that I'm not actually recreating nodes all the time (only some of the time).

```
→ BTree git:(main) x ls
btree.py  btree3.py  btree5.py  btree7.py  btree9.py  test.py
btree2.py  btree4.py  btree6.py  btree8.py  inplace.py
→ BTree git:(main) x
```



Thu 7 Sep 2023 13:30 - 14:00 at A - Grand Ballroom 2 - Language design Chair(s): Peter Thiemann

## ★ The Verse Calculus: a Core Calculus for Functional Logic Programming

**DISTINGUISHED PAPER**

Functional logic languages have a rich literature, but it is tricky to give them a satisfying semantics. In this paper we describe the Verse calculus, VC, a new core calculus for functional logic programming. Our main contribution is to equip VC with a small-step rewrite semantics, so that we can reason about a VC program in the same way as one does with lambda calculus; that is, by applying successive rewrites to it. We also show that the rewrite system is confluent.



**Lennart Augustsson**  
Epic Games



**Koen Claessen**  
Epic Games



**Simon Peyton Jones**  
Epic Games

United Kingdom



**Guy L. Steele Jr.**  
Oracle Labs

United States



**Joachim Breitner**  
Unaffiliated

Germany



**Ranjit Jhala**  
Epic Games



**Olin Shivers**  
Epic Games



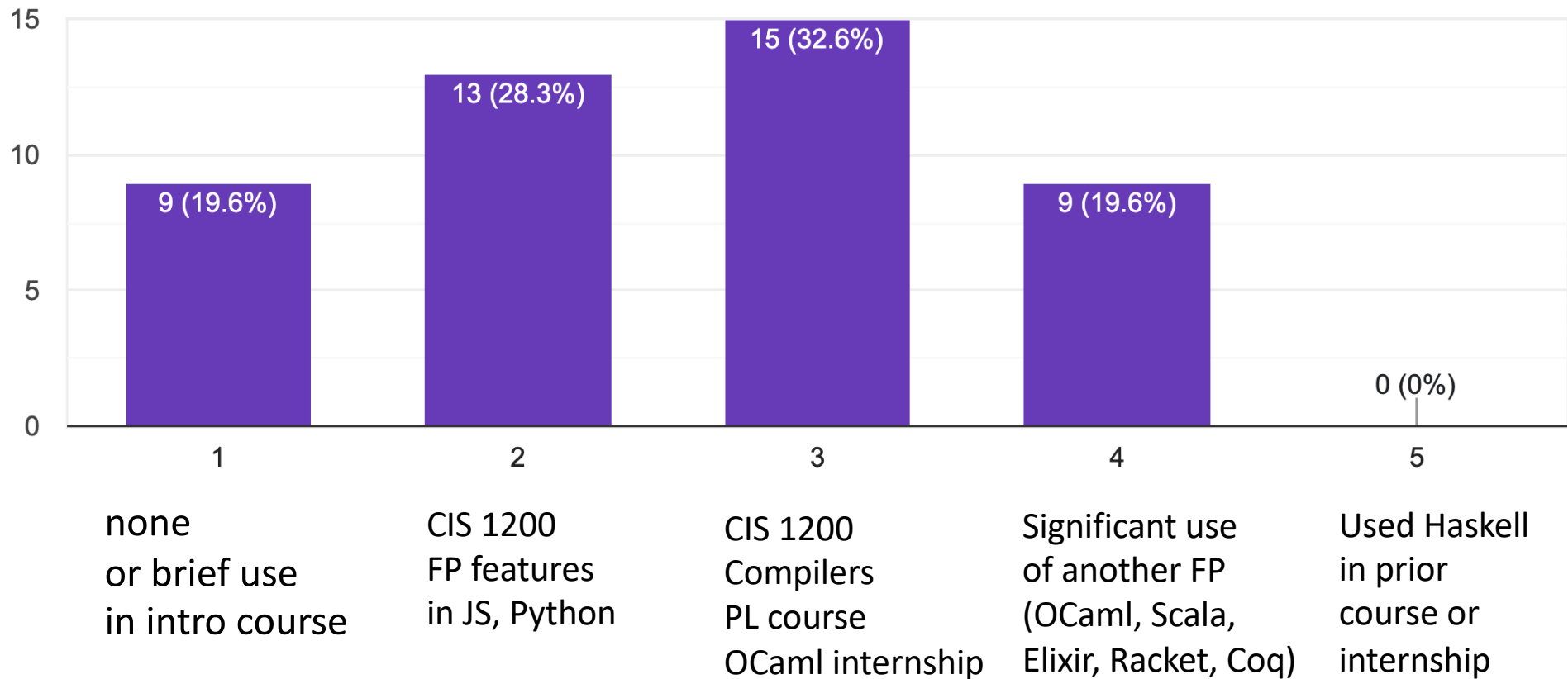
**Tim Sweeney**  
Epic Games

# Audience

- People with strong background in programming and mathematics
- No experience with FP expected, but helps
- Undergraduates, Masters, and PhD students together

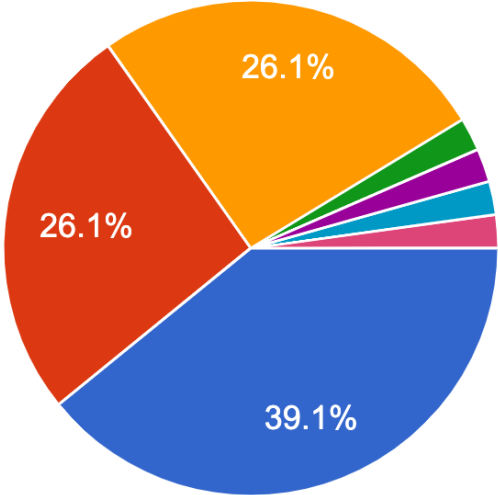
# How much experience do you have with functional programming ?

46 responses



# What is your status in Fall 2023

46 responses



- Undergraduate
- Submatriculant
- MSE or MCIT program (non submatriculant)
- PhD
- DATS Masters Student
- IGSP student, fourth year undergraduate in my home university...
- Undergrad, want to submatriculate but not yet finished requirements



*How will this all  
work?*

# General Course Structure

- Every week has a topic
  - Read module and complete quiz by beginning of class Monday
  - Interactive lecture Monday (module highlights w/live coding)
  - Active in-class exercise Wednesday
  - Homework due alternate Thursdays (midnight), covers two topics
- Some weeks may be different (Fall break, Thanksgiving)
- End-of-semester: final project

# Grading Structure

- 15 % Quizzes
  - pre-class quizzes (study "lecture" content asynchronously)
  - **first module/quiz available now**
- 15 % Active learning / Attendance
  - in class exercises
  - office hours – let's chat!
- 50 % Programming assignments
  - in pairs, most *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**
  - Quiz graded mostly on completion

# Active Learning Goals

- Goal for the semester: create a CIS 5520 *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 class time and outside
  - Small and large group discussions
  - In-class exercises with a partner or table
  - Random homework partners
  - TODAY: PL-themed icebreaker game

# Homework #1

- Based on "Basics" (available now) and "HigherOrder" modules (tba)
- Clone public repo to complete the assignment
- Work alone or with a partner (your choice), only one person should submit via Gradescope
- Must compile to get any credit
- **Due Thursday, Sept 14th 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 have an emergency, please ask for an extension*

# Academic Integrity Expectations

- CIS 5520 is a course and not a developer job
  - we will ask you to refrain from using standard libraries or referring to (easily accessible) solutions
- **Homework solutions must be yours**
  - Don't ask ChatGPT to solve your homework
  - Don't search for solutions online
  - Don't ask someone else (other than your partner) to do your homework for you
- Can make limited use of ChatGPT, but *do so with caution*
- *Ask if you are unsure!*

# Where to go for more information

- Public site (<http://www.seas.upenn.edu/~cis5520>)
  - Haskell related material, HW instructions
- Github (<https://github.com/upenn-cis5520>)
  - Code repos for lecture content, in-class exercises (public)
  - HW repos
- Canvas site (<https://canvas.upenn.edu/courses/1741501>)
  - **Syllabus**
  - Quizzes
  - Link to Ed (Announcements and questions)
  - Link to Gradescope (Homework submission)



## First three weeks

- Today: Introductions
- Wed, Sep 6: **Basics** module, quiz due (SCW at ICFP)
- Mon, Sep 11: **HigherOrder** module, quiz due
- Wed, Sep 13: **Foldr** in-class exercise (SCW at VerseCon)
- Thurs, Sep 14: HW #1 due

# Waitlist and registration

- Current status: 24 on waitlist, 9 unused permits
- I will process waitlist requests until September 12<sup>th</sup>
- Priority to those who fill out intro survey, come to class, and complete the quiz
  - (If you are here today, I will add you to Canvas so you can access the quizzes)
  - Send me email if you will miss class
- Let me know if you no longer want to be on the waitlist

# Things to do right now

- Read syllabus on Canvas
- Create a github account (if you do not have one)
- Respond to Fall 2023 intro survey (if you haven't already)
- Introduce yourself to the others at your table
- Start reading "Basics" module, install software, clone hw01 repo (after class)
- Office hours this week:  
Stephanie: Today, 1:30-2:30 PM, Levine 510

# PL game!

- Each table is a team and must have a team name (we'll collect names when we start)
- Match each code listing with its **algorithm** and **programming language**
- Each algorithm / language is used only once
- No google / web searching allowed
- Bring completed sheets to me for scoring, I will process in order. Only one guess per sheet!
- Winner is the team with the most points by **1:20PM**

*fin*

So, Who Uses FP?

Google™

So, Who Uses FP?



**Microsoft®**

So, who uses FP?

The Facebook logo, consisting of the word "facebook" in white lowercase letters on a blue rounded rectangular background.

**facebook**



So, Who Uses FP?

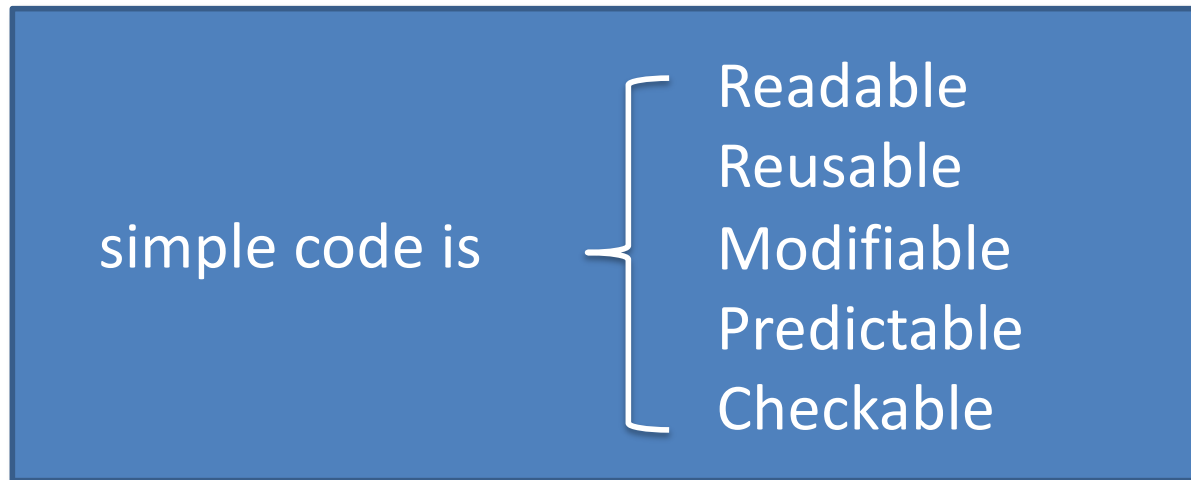


# So, Who uses FP?



# Goal: Obviously no deficiencies

- Want code that is so simple, it obviously works



- OK... so what makes code simple?