Lecture 1
Course Overview
JavaScript

```
[] + []
{} + []
[] + {}
[] + {} 
{} + {}
```

From Wat:
https://www.destroyallsoftware.com/talks/wat
class A {
    static int a = B.b + 1;
}

class B {
    static int b = A.a + 1;
}
Python

```python
a = [1], 2
a[0] += 3
```
Java and Scala

The Unsound Playground

We, Nada Amin and Ross Tate, broke the Java and Scala type systems!
Try it out for yourself by running the examples, which throw cast exceptions even though they contain no casts ↓
Read our paper Java and Scala's Type Systems are Unsound to learn how these examples work →
Come up with your own examples and use the save icon to update the URL to a permalink to your code ⬆

Which language would you like to break?
Java / Scala

- Unsound.java in Java 8 (the first program we broke Java with, derived from legacy.scala)
- Unsound.java in Java 9 (sometimes compilers incorrectly reject valid code)
- Unsound.java in Java 9 (we even broke Java in the future)
- UnsoundSpec.java in Java 6 (incorrectly rejected unsound program that has been valid since Java 5 in 2004)
- Nullable.java in Java 8 (this program has no occurrence of null)
- SingleParameters.java in Java 8 (every class and method in this program has exactly one parameter)

class Unsound {
  static class Constraint<A, B extends A> {}
  static class Bind<A> {
    <B extends A> B
      A upcast(Constraint<A, B> constrain, B b) {
        return b;
      }
    }
    T coerce(T t) {
      Constraint[U, ? super T> constrain = null;
      Bind<U> bind = new Bind<U>();
      return bind.upcast(constrain, t);
    }
  }
}
Question: What makes a good programming language?
Design Desiderata

**Question:** What makes a good programming language?

**One answer:** A good language is one people use.
Question: What makes a good programming language?

One answer: A good language is one people use.

Wrong! Is JavaScript bad? What’s the best language?
Question: What makes a good programming language?

One answer: A good language is one people use.

Wrong! Is JavaScript bad? What’s the best language?

Some good features:

- Simplicity (clean, orthogonal constructs)
- Readability (elegant syntax)
- Safety (guarantees that programs won’t “go wrong”)
- Modularity (support for collaboration)
- Efficiency (it’s possible to write a good compiler)
Unfortunately these goals almost always conflict.

- Types provide strong guarantees but restrict expressiveness.
- Safety checks eliminate errors but have a cost—either at compile time or run time.
- A language that’s good for quick prototyping might not be the best for long-term development.
Design Challenges

Unfortunately these goals almost always conflict.

- Types provide strong guarantees but restrict expressiveness.
- Safety checks eliminate errors but have a cost—either at compile time or run time.
- A language that’s good for quick prototyping might not be the best for long-term development.

A lot of research in programming languages is about discovering ways to gain without (too much) pain.
Course Staff

Instructor
Adrian Sampson

Teaching Assistants
Pedro Amorim
Devin Lehmacher
Alexa VanHattum
Irene Yoon
Prerequisites

Mathematical Maturity

- Much of this class will involve formal reasoning
- Set theory, formal proofs, induction

Programming Experience

- Comfortable using a functional language
- For undergrads: CS 3110 or equivalent

Interest (having fun is a goal!)

If you don’t meet these prerequisites, please get in touch.
Programming Languages and Logics
Fall 2018

Monday, Wednesday, and Friday
at 9:05–9:55am
in Gates G01

Instructor

Adrian Sampson
office hours: Monday 10–11am and Friday 11am–noon
in Gates 411A

http://www.cs.cornell.edu/courses/cs4110/2018fa/
Course Work

Homework
- 9 assignments, roughly one per week
- Always due on Wednesday night at 11:59pm
- Can work with one partner
- Three slip days and lowest score discarded
- Otherwise, no late submissions

Preliminary Exams (in class)
- October 3
- November 14

Final Exam
- Date TBD

Participation (5% of your grade)
- Introduction survey (out now!)
- Mid-semester feedback
- Course evaluation
Academic Integrity

Some simple requests:

1. You are here as members of an academic community. Conduct yourself with integrity.

2. Problem sets must be completed with your partner, and only your partner. You must *not* consult other students, alums, friends, Google, GitHub, StackExchange, Course Hero, etc.!

3. If you aren’t sure what is allowed and what isn’t, please ask.
Respect in Class

We hold all communication (in class & online) to a high standard for inclusiveness. It may not target anyone for harassment, and it may not exclude specific groups.

Examples:

- Do not talk over other people.
- Do not use male pronouns when you mean to refer to people of all genders.
- Avoid language that has a good chance of seeming inappropriate to others.

If anything doesn’t meet these standards, contact the instructor.