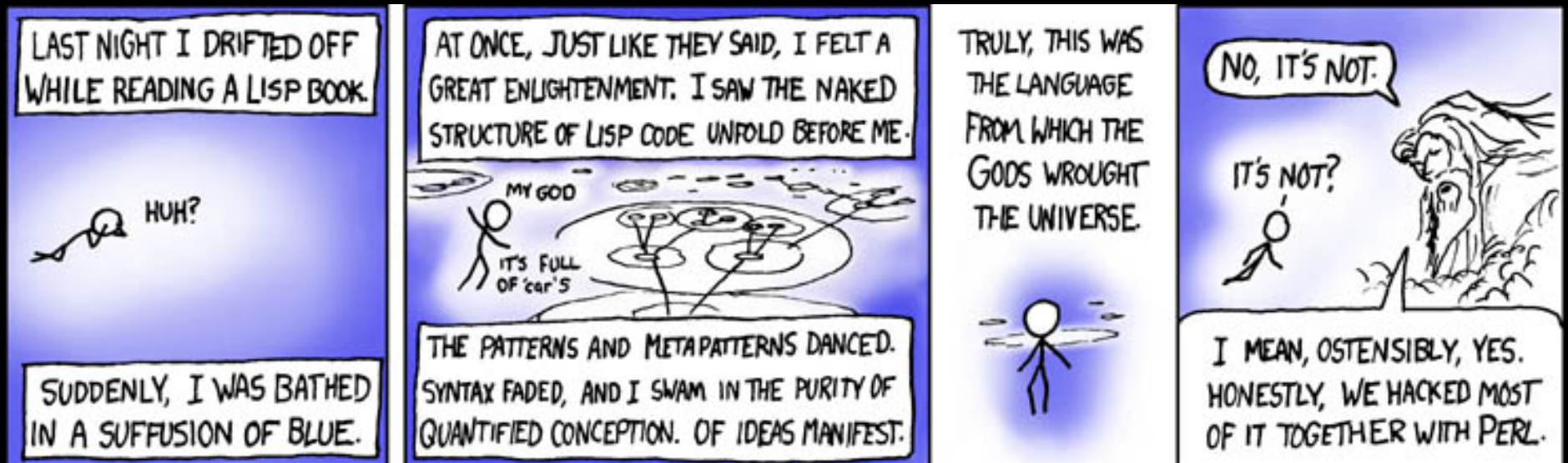


<http://xkcd.com/224/>



CS 152: *Programming Language Paradigms*



Prof. Tom Austin

San José State University

What are some
programming languages?

A word cloud of programming languages on a black background. The languages are represented by text in various colors and sizes. 'Java' is the largest word in blue. 'Python' is the second largest in green. Other languages include 'C++' in yellow, 'C#' in yellow, 'PHP' in blue, 'lua' in green, 'VBA' in green, 'R' in green, 'Objective-C' in green, 'Scala' in orange, 'Matlab' in orange, 'Perl' in blue, 'Swift' in blue, 'Ruby' in purple, 'Visual Basic' in cyan, and 'Javascript' in orange.

lua C++ C# VBA R
Objective-C
PHP Java
Scala Matlab
cPython
Perl Swift
Ruby Visual Basic Javascript

Taken from <http://pypl.github.io/PYPL.html>
January 2016



Taken from <http://pypl.github.io/PYPL.html>

August 2019

Why are there so many?

Different domains



Different design choices

- Flexibility
- Type safety
- Performance
- Build time
- Concurrency

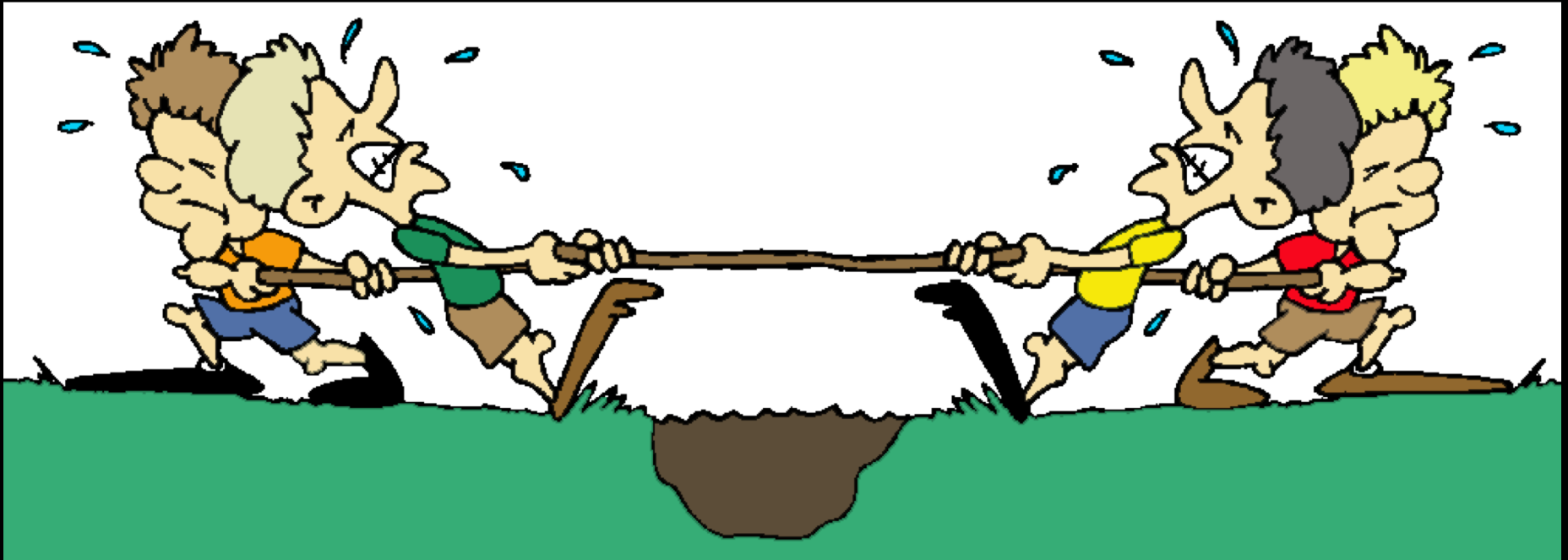


Which language is better?

Good language features

- Simplicity
- Readability
- Learn-ability
- Safety
- Machine independence
- Efficiency

These goals almost always conflict



Conflict: Type Systems

Stop "bad" programs

... but ...

restrict the programmer

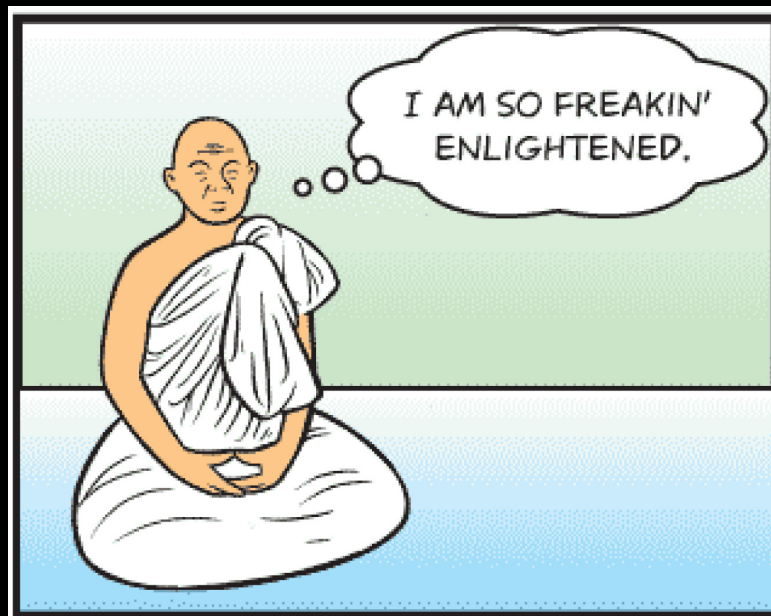
Why do we make you take a programming languages course?

- You **might use one** of these languages.
- Perhaps one of these languages is the *language of the future* (whatever that means).
- You might see **similar languages** in your job.
- Somebody made us take one, so now we want to make you suffer too.
- But most of all...

We want to warp your minds.

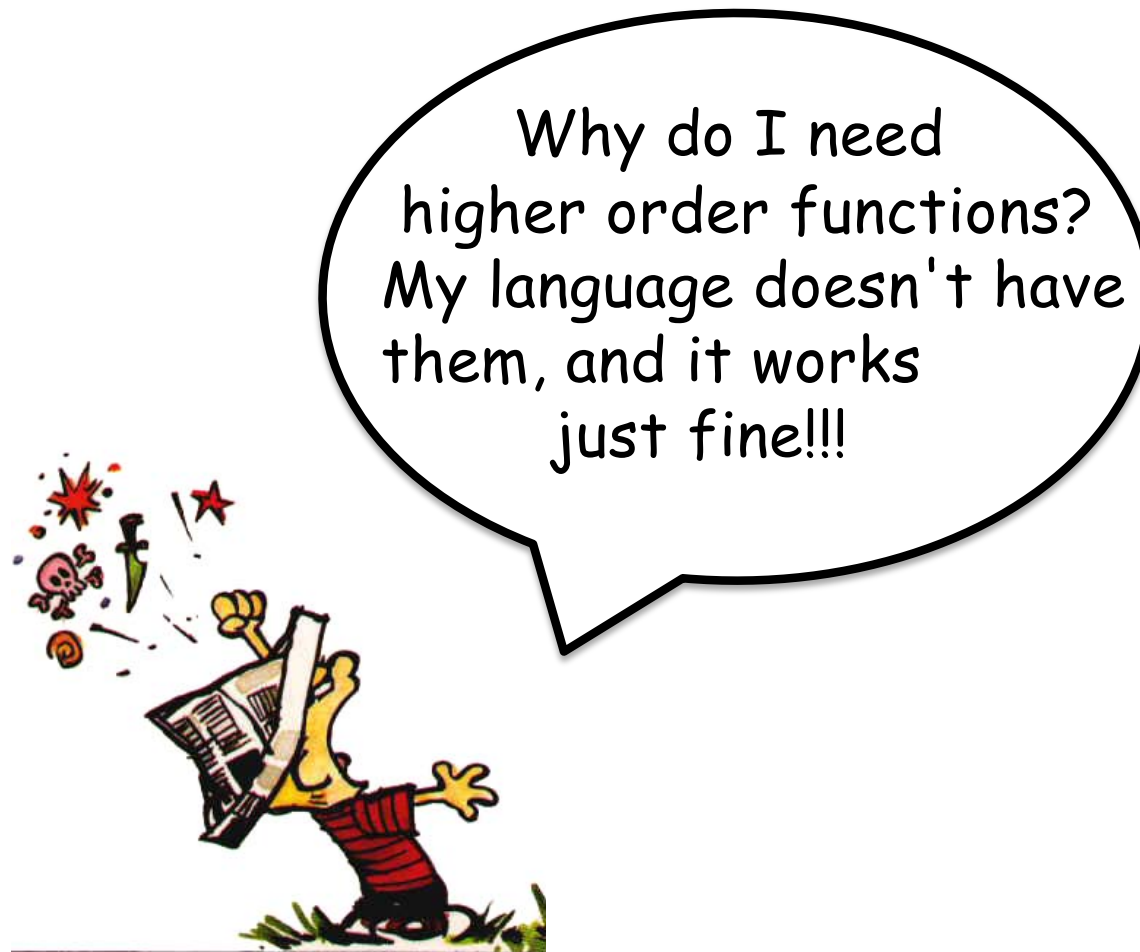


Course goal: change the way that you think about programming.



That will make you a better *Java* programmer.

The "Blub" paradox



"As long as our hypothetical Blub programmer is looking down the power continuum, he knows he's looking down..."

[Blub programmers are] satisfied with whatever language they happen to use, because it dictates the way they think about programs."

--Paul Graham

<http://www.paulgraham.com/avg.html>

Languages we will cover

(subject to change)



Administrative Details

- Green sheet:
<http://www.cs.sjsu.edu/~austin/cs152-spring20/greensheet.html>.
- Homework submitted through Canvas:
<https://sjsu.instructure.com/>
- Academic integrity policy:
<http://info.sjsu.edu/static/catalog/integrity.html>

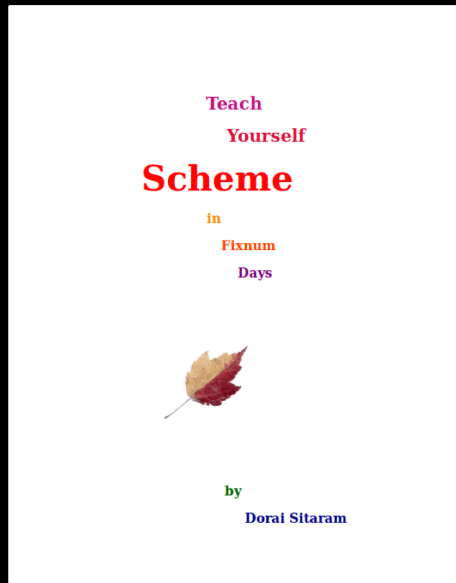
Schedule

- The class schedule is available through Canvas.
- Late homeworks will not be accepted.
- CHECK THE SCHEDULE BEFORE EVERY CLASS.

Prerequisites

- **CS 151 or CMPE 135,**
grade C- or better
- Show me proof
 - If you don't, I will drop you.

Resources



Dorai Sitaram
"Teach Yourself Scheme
in Fixnum Days".

<http://ds26gte.github.io/tyscheme/>

Other references TBD.

Grading

- 30% -- Homework assignments (individual work)
- 20% -- Class project (team work)
- 20% -- Midterm
- 20% -- Final
- 10% -- Participation (labs and drills)

Participation: Labs

- No feedback given (usually)
- I will look at them
- If you have questions, ask me

Homework

- Must be done individually
- If your assignment is too close to another student's,

**YOU BOTH GET A
ZERO.**

- Academic integrity policy:
<http://info.sjsu.edu/static/policies/integrity.html>

Project

- Work alone or with ONE partner.
- Goal: Build an interpreter.
- Use Java and ANTLR

Office hours

- MacQuarrie Hall room 216
- Mondays and Thursdays,
noon-1pm
- Also by appointment



Racket/
Scheme

What is Scheme?

- A functional language
 - Describe what things are, not how to do them.
 - More mathematical compared to imperative langs.
- A dialect of Lisp (**L**ist **P**rocessing)
- (Famously) minimal language
- Racket is a dialect of Scheme

Symbolic Expressions (s-expressions)

The single datatype in Scheme. Includes:

- **Primitive types:** booleans, numbers, characters, and *symbols*.
- **Compound data types:** strings, vectors, pairs, and of course...

LISTS!!!

Scheme lists

- Sample list:
`(list 1 2 3 4)`
- Alternate form:
`'(1 2 3 4)`
- Important functions:
 - `car`: gets the first element of the list.
 - `cdr`: gets the tail of the list.
 - `cons`: combines an element and a list.
 - `append`: appends multiple lists together.

Calling functions in Scheme

- First argument assumed to be a function
- Rest of the list are its arguments

// Java

`foo(x, y, z);`

; Scheme

`(foo x y z)`

```
$ racket
```

```
Welcome to Racket v6.0.1.
```

```
> '(1 2 3 4)
```

```
'(1 2 3 4)
```

Quote indicates list
is data

```
> (car '(1 2 3 4))
```

```
1
```

First element is
assumed to be a
function

```
> (cdr '(1 2 3 4))
```

```
'(2 3 4)
```

```
> (+ 1 (* 2 4) (- 5 1))
```

```
13
```

```
>
```

Before next class

- Install Racket from <http://racket-lang.org/>
- Read chapters 1-2 of *Teach Yourself Scheme*.
- Read Paul Graham's "Beating the Averages" article.
<http://www.paulgraham.com/avg.html>

First homework due February 13th

- This assignment is designed to get you up and running with Racket.
- Available in Canvas.
 - If you don't have access to Canvas, see <http://www.cs.sjsu.edu/~austin/cs152-spring20/hw/hw1/> instead.
- Get started now!

Lab 0

Familiarize yourself with scheme

Write functions to calculate the area of

- A rectangle
- A square
- A triangle