

# LIS-3353

## Programming



Everyone needs to learn to code!!

?



~~Everyone needs to learn to code!!~~

Everyone needs to learn  
about coding.



“Lots of very simple instructions  
can add up to complex  
computations.”

“Turing Completeness / Turing Machine”  
(an infinite tape w/ simple instructions)

“Lambda Calculus”  
(this is literally all you have to know)



# Understanding Power

```
10 PRINT "John is AWESOME";  
20 GOTO 10
```



# WARNING:

Discussions about coding are very passionate  
and controversial..

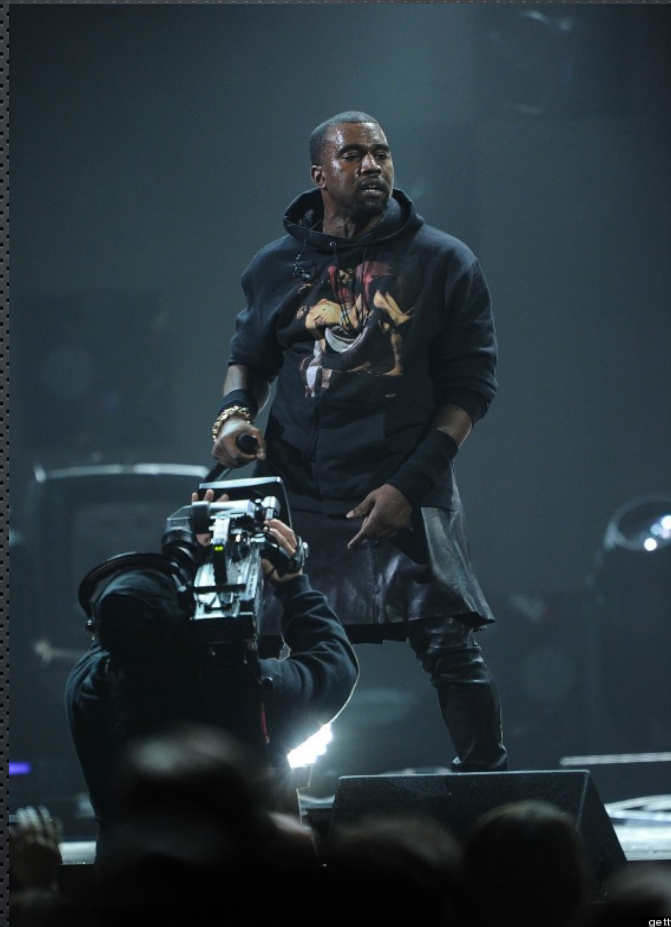
Everyone thinks that their way is the best, and  
that anything not their way sucks.

(someone **might** be right...)



# “Popularity” can be misleading..

Languages are like fashion.. maybe the new hot thing will stick around, maybe not...(and vice versa)





# Know about BOTH..

What you hear about

v.

What people actually use



# A bit on “coding”

What is taught as coding is usually:  
“IN THE BEGINNING”

ie:



# A bit on “coding”

```
print "Hello world";
```

and/or

```
for i = 1; i < 10; i = i + 1{  
  print " the count is $i"  
}
```

(as if you were starting from scratch)



But perhaps we should be doing;  
“Unix/Linux way scripting”

- how to find and download little linux programs
- how they all work on the command line
- how they all talk in/with text
- how to string them together to do **useful things**



# Human Readability ("Matrix" vs. "English")

Old school - Human readability is not important

- Short abstractions are concise and thus quicker
- Forced whitespace is limiting
- There should be *MANY* ways to do a thing.



# Human Readability

## ("Matrix" vs. "English")

New school – Hey, looks like human readability is at least a little important:

- Multiple people working on projects
- Older code needs to be understandable
- There is value in forcing people to do things only one way

*"Always code as if the guy who ends up maintaining your code will be a violent psychopath who knows where you live."*



# “Closeness” to machine

**Compiled** (older method) – Before running, you have to “convert” it. Usually Makes for faster/more efficient code.

**Interpreted** – “Conversion” happens on the fly.

**Scripted** – No “conversion” necessary. Usually only good for shorter/smaller/ scripted things.



# Compiled Languages

- Older
- “Normal”
- “Close to machine”



# “COMPILED” – Source Code





# Compiled Code (binaries)





# Why is “scripting” NOT considered “real programming?”

- Quick and dirty
- Can be slow (especially as compared to compiled)
- Lacks “libraries” or “frameworks”
- Few tools/structures designed for reuse or collaboration



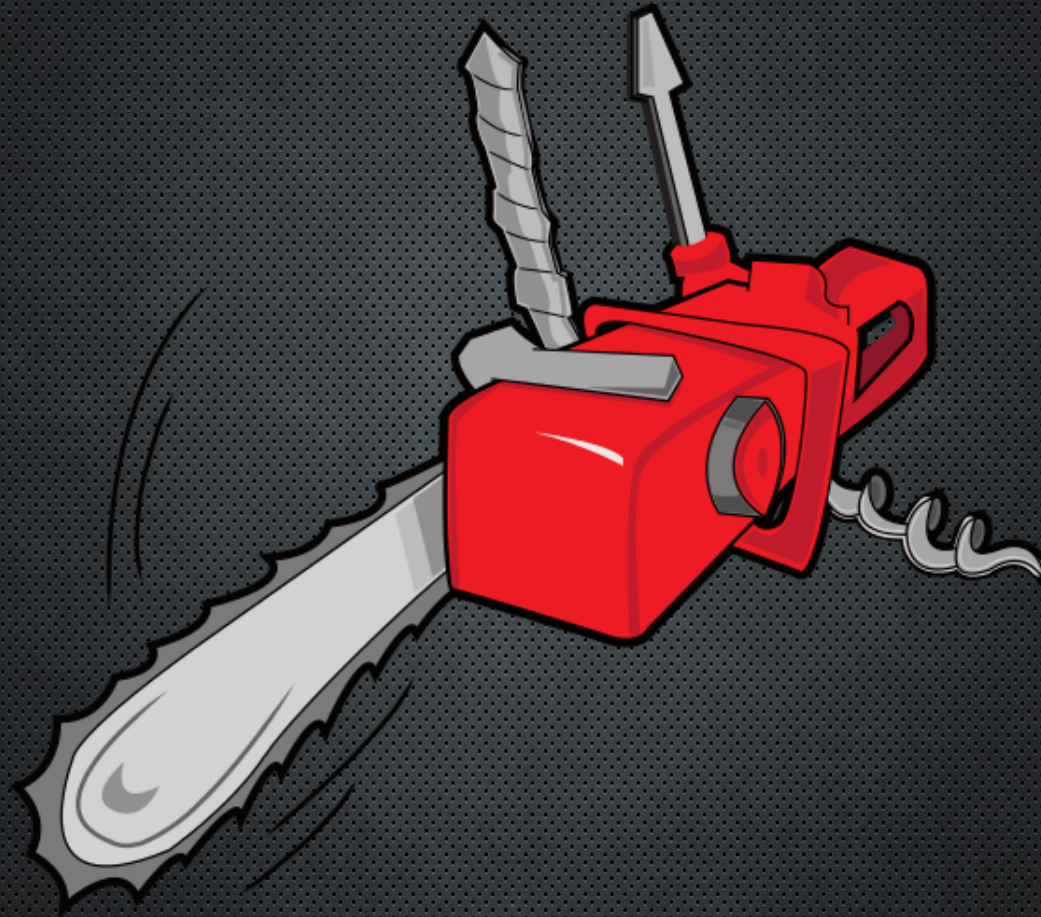
# “Real” Programming Languages

(C, Python, Ruby, Java etc.)





# Bash/Shell scripting





For example...

```
sudo rm -rf /
```

- seriously, don't do this.



# INTERPRETED LANGUAGES:

- Basically, they split the difference; interpreted on the fly. MOST WEB LANGUAGES are this:

PHP and Javascript!



# Important “real life” Categories. (these are fuzzy)

- **Procedural (default)** – Do things step by step
- **Functional** – Turn it into straight up math. No variables, no “procedures”
- **Object Oriented** – Everything uses an “object” metaphor. (“Black Box Approach”)



## More on “age...”

Older usually equals "more ways than one"

Older usually equals less "human readable"

Older usually means more versatile

Older usually means more support/libraries



# Languages and the Web

"Normal" Languages – web glue required

Lisp, C, Java\*, Perl, Ruby\*, (Bash), Python

"Web required" Languages

PHP, Javascript, Flash

\*still VERY web oriented



# Lisp (and other functionals)

(Haskell, Clojure)

- MATHY
- VERY, perhaps TOO “versatile”
- Stallman and Emacs
- Functional is seeing a resurgence, because of its mathematical “purity.” There are NO VARIABLES, WHOA.



# C (C++, C#)

- Practical granddaddy of (modern) everything
- Still a good choice for speed and control, at the cost of ease
- "Why C sucks" -- TOO MUCH control, too much “hardcore stuff” to worry about (e.g. “*malloc*”)



# Python

- Forced whitespace
- Usually "one way to do a thing"
- In the "middle" on just about all parameters.
- "Why Python Sucks" – oddly, not super popular – perhaps not SPECIALIZED enough



# Ruby

Designed for elegance, ease of use, but still powerful

- Highly OO
- Very easy to get started
- Very easy to read, yet still concise.
- Poignant's guide to ruby
- Arguably not quite as fast as slightly older languages
- "Why Ruby Sucks" – Turns out to not be fast enough, probably getting eaten by Javascript?



# Java

- First attempt at dethroning C, looks a lot like it
- Originally company driven, as a result got popular/useful quickly
- interpreted, not compiled (Virtual Machine required)
- "Why Java Sucks" - slow, verbose, too many options/fragmented



# SERVER-SIDE

Code resides on server

Code is executed by server

Dynamic content is produced

(Ruby on Rails, Django, PHP\*, Wt)

Note why PHP is a little weird. Instead of independent code, PHP code is EMBEDDED IN HTML, but run by the server -- where you'd usually expect it to be the other way around, i.e., have your code PRODUCE html. This also might be why it's so popular.



# PHP

- Designed to handle the web and HTML
- Features were added as needed, not from ground up
- "Why PHP Sucks" Purists HATE IT; probably the most duct-tapiest language of all (still, facebook and wordpress)  
(ASP is Microsoft's slightly different “PHP”)



# CLIENT SIDE

## CLIENT SIDE APPROACHES

### Java/Flash

Download their little programs, and let them run in your browser

(yes, this can be as dangerous as it sounds)

### Javascript

CODE is EMBEDDED in the html and RUN IN/BY YOUR BROWSER.

(yes, this is even more dangerous than the above. Also very useful. Get NoScript.) -

(greasemonkey is cool, though)

### Silverlight- M\$'s flash

**HTML5** - will hopefully save us all. Reimplementing all the good stuff in flash, openly.



# HTML/CSS

- Not really languages. ( Can't really do anything besides dress up and move around text and pictures and other things)
- Traditionally, for dressing up text because fonts and colors (and [HYPERLINKS](#) WHOA) used to be a pretty big deal.
- Today, not even a great choice if you need a website, BUT you should know it because it's now the **FRAMEWORK for the web.**



# Flash

- Designed for "multimedia" - animations, video etc.
- Can be hacked into a general purpose language (but didnt start as one)
- Very "closed"
- "Why Flash sucks" -- ah, where to begin



# Many more languages out there

Scheme, Scala, Haskell, Lua

LOLCODE, Shakespeare

Brainf\*\*k, Whitespace



# THE SIMPLE WEB

- XML – text, but with bracketed up metadata

Markdown, Zim for other examples of this concept in action

(note, JSON is similar to XML and can do all the same things, but is designed for Javascripty stuff)

- HTML – (very forgiving) “XML” for the web  
(tags for bold/headlines, etc)

- CSS – Hey, if you have a website, you'll probably want to have everything the same font/color, why not specify that ONCE and be done with it.



# CGI-BIN

Common Gateway Interface.

- That is, just let the website "reach in" to your computer and run stuff.
- Thus, any language can be used here. Perl is common
- Old school, not common anymore



# Newer, popular “glues”

Other popular "glues" – from language to web

- Django for Python
- Catalyst for Perl
- Ruby on Rails
- (Bash on Balls?)



# Javascript

- Though looks like c, VERY DIFFERENT FROM JAVA. Confusing, huh?
- CLIENT SIDE (mostly).
- JAVASCRIPT IS EATING THE WEB RIGHT NOW. LOOKING VERY DOMINANT.



# But what about personally?

First Principles:

The useful information/communication is  
TEXT.



# Text tools:

Built in:

Notepad/Leafpad.

Word? (just say no?)



# Text tools: Programming

Classic:

Vim / Emacs / Nano

Newer:

Comodo / jedit / gedit / Notepad++

Even newer:

Sublime/Atom



# Text tools: Taking notes/notebooks:

Zim?

Anything with outlining?

Emacs Org-Mode.



# Text tools: Communication

Email (very open)

v.

SMS (very closed)

v.

Everything else? (Slack / Facebook / Signal /etc)



# Text tools: News

Facebook/Social Media (push)

vs.

RSS/ Simple list of links (pull)