Programming Languages

Attendance:
https://tinyurl.com/997431

> Programming Languages

- Purpose:

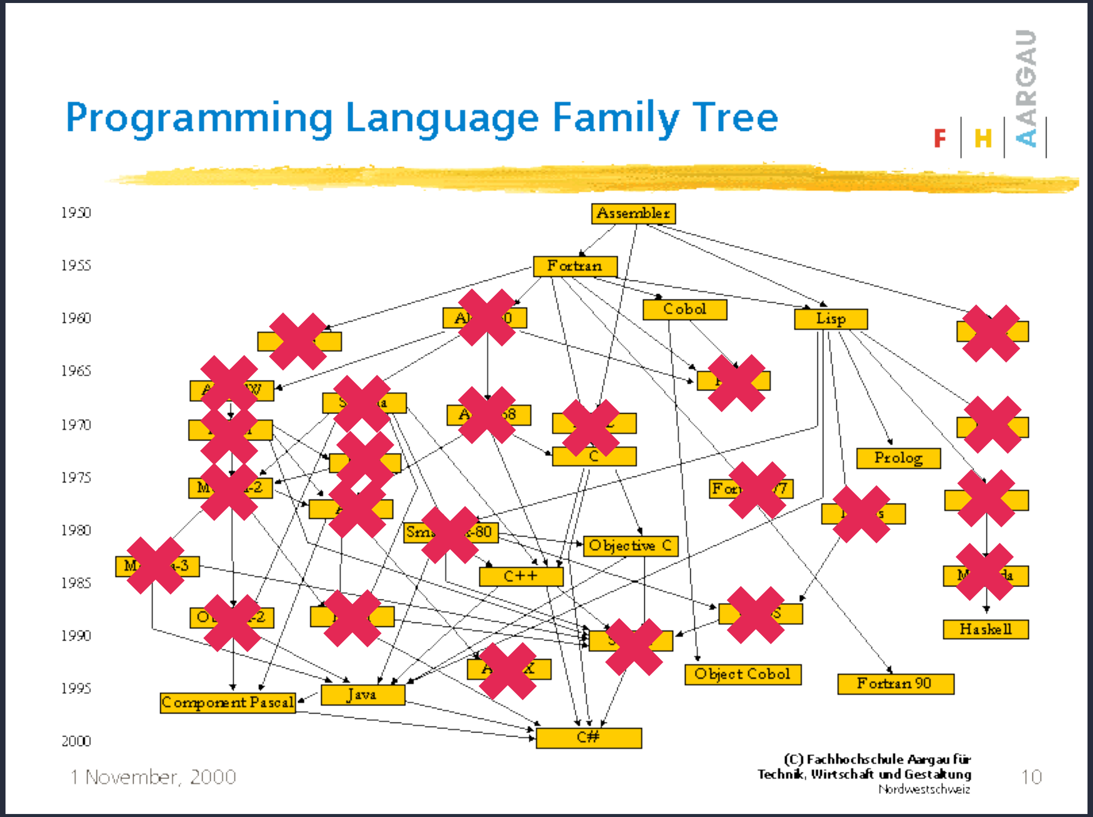
```
def main():
    for i in range(200):
        print("Hello World")
```

```
68 62 2e 2e 2e
89 e7
33 c0
...
```

- Because writing machine code is not fun
- (As you will discover in 15-213)

> Programming Languages

- There are a lot of languages
- We will focus on the ones people use



Types of programming languages

> Types of programming languages

- Imperative languages
  - Sequence of commands used to modify state
- Functional languages
  - Pure functions acting on persistent data
- Object-oriented languages
  - Broken
  - Most popular type of language

> Interpreted vs Compiled

- Compiled languages
  - Translated to machine code by a compiler
- Interpreted languages
  - An interpreted translates each instruction line-by-line during execution
- just-in-time (JIT) Compiled languages
  - Code segments are compiled at runtime as they are needed

> Static vs Dynamic typing

- Statically typed
  - Types are checked at compile-time
- Dynamically typed
  - Types are checked at runtime

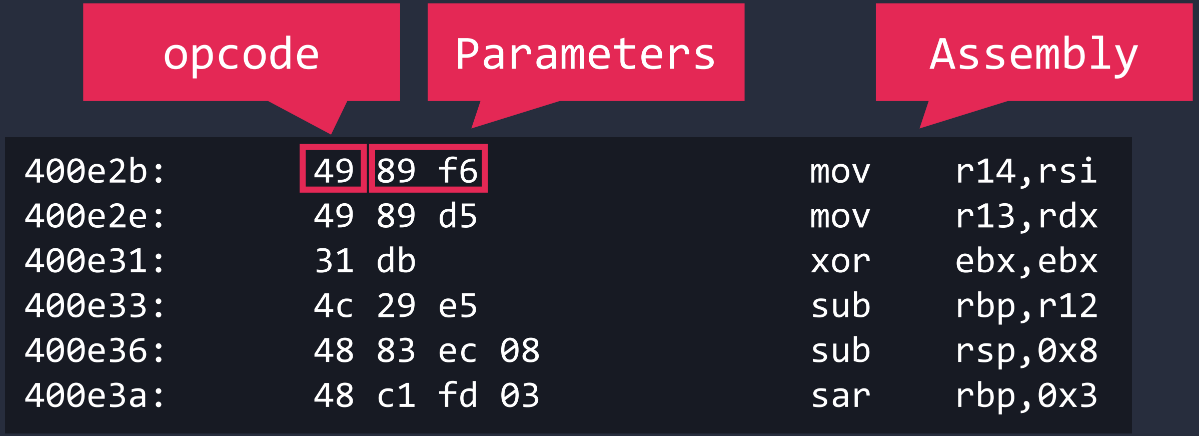
> Strong vs Weak typing

- Strong typing
  - Types must by explicitly converted
- Weakly typed
  - Program "figures out what you meant" and automatically converts types
  - In the case of JavaScript, it is almost never what you meant

Guided tour of programming languages

> Assembly

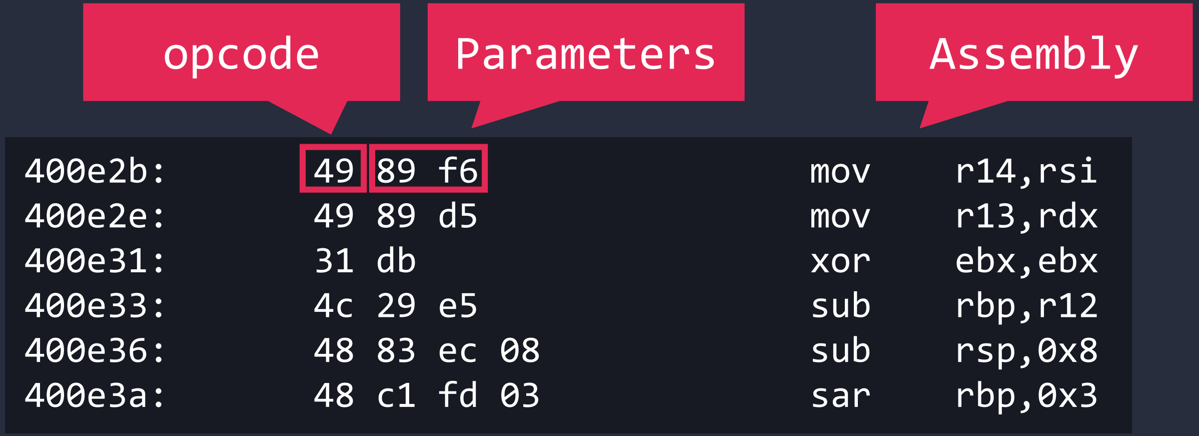
- One step away from machine code
- Turned into machine code by an assembler
- Intel vs AT&T sytax
  - Intel = Dest Source
  - AT&T = Source Dest



```
400e2b:       49 89 f6             mov    r14,rsi
400e2e:       49 89 d5             mov    r13,rdx
400e31:       31 db                xor    ebx,ebx
400e33:       4c 29 e5             sub    rbp,r12
400e36:       48 83 ec 08          sub    rsp,0x8
400e3a:       48 c1 fd 03          sar    rbp,0x3
```

> Assembly

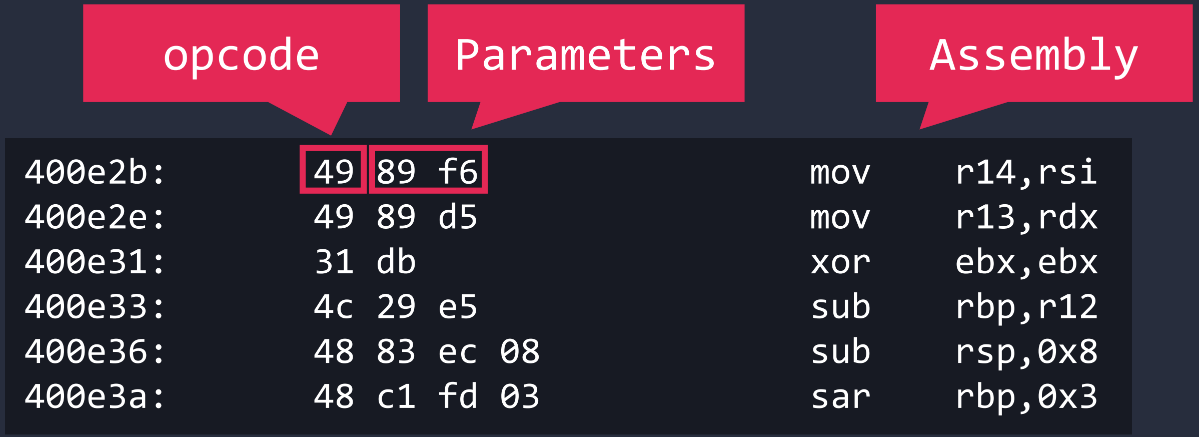
- Advantages:
  - Extremely fast
- Use case:
  - When you are writing code for devices with limited resources



```
400e2b:      49 89 f6          mov    r14,rsi
400e2e:      49 89 d5          mov    r13,rdx
400e31:      31 db             xor    ebx,ebx
400e33:      4c 29 e5          sub    rbp,r12
400e36:      48 83 ec 08       sub    rsp,0x8
400e3a:      48 c1 fd 03       sar    rbp,0x3
```

> Assembly

- disassemble a binary file with:
  - objdump -d -M intel [file name]



```
400e2b:        49 89 f6              mov     r14,rsi
400e2e:        49 89 d5              mov     r13,rdx
400e31:        31 db                 xor     ebx,ebx
400e33:        4c 29 e5              sub     rbp,r12
400e36:        48 83 ec 08           sub     rsp,0x8
400e3a:        48 c1 fd 03           sar     rbp,0x3
```

> C

- Like c0 but with segfaults
- A fast imperative compiled language
- A "small" language. Low overhead.
- Manual memory management
- Behavior defined by the ISO
- Undefined behavior
- Has a powerful macro system

> C++

- A superset of C
- Manual memory management
  - Mostly
- Object-oriented
- Supports templating
  - Pleasant error messages

> C++

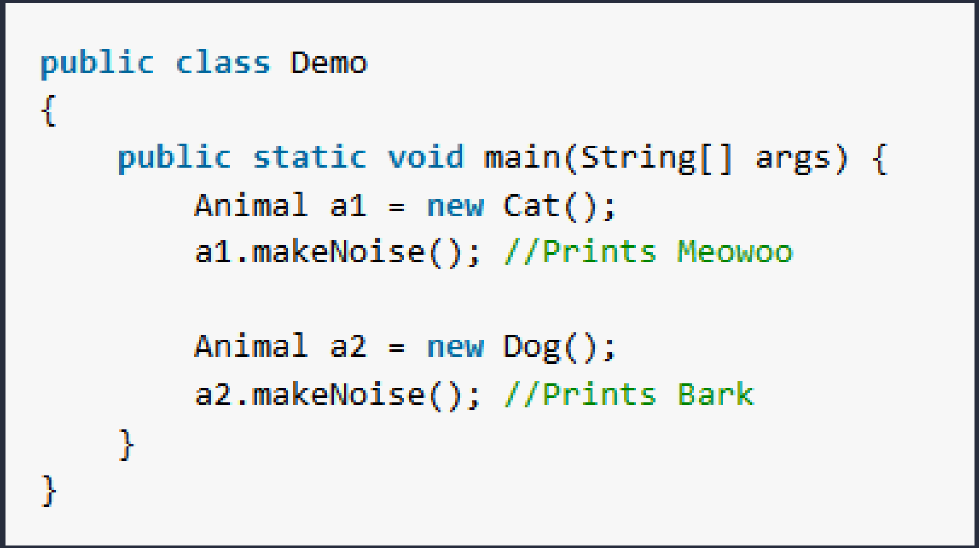
- Advantages:
  - Fast, low-level
  - More functionality than C
- Use cases:
  - Large applications that must be very performant
  - Ex: Computer Graphics

> Go

- Modern low-level language designed by Google
- Features
  - memory safety
  - garbage collection
  - good concurrency support
- No Generic types

> Java

- Object-oriented to the extreme
- Supports dynamic dispatch
- Enforces encapsulation (access level modifiers)



```java
public class Demo
{
    public static void main(String[] args) {
        Animal a1 = new Cat();
        a1.makeNoise(); //Prints Meowoo

        Animal a2 = new Dog();
        a2.makeNoise(); //Prints Bark
    }
}
```

Dispatch Matrix

	Method 1	Method 2
Subclass 1	[method]	[method]
Subclass 2	[method]	[method]

> Java

- Compiles to Bytecode for the JVM
- Slower than compiled languages like C++ or C
- Object oriented languages have some issues
  - Subtyping generic types
  - Is it vector.times(scalar) or scalar.times(vector)?

> Python

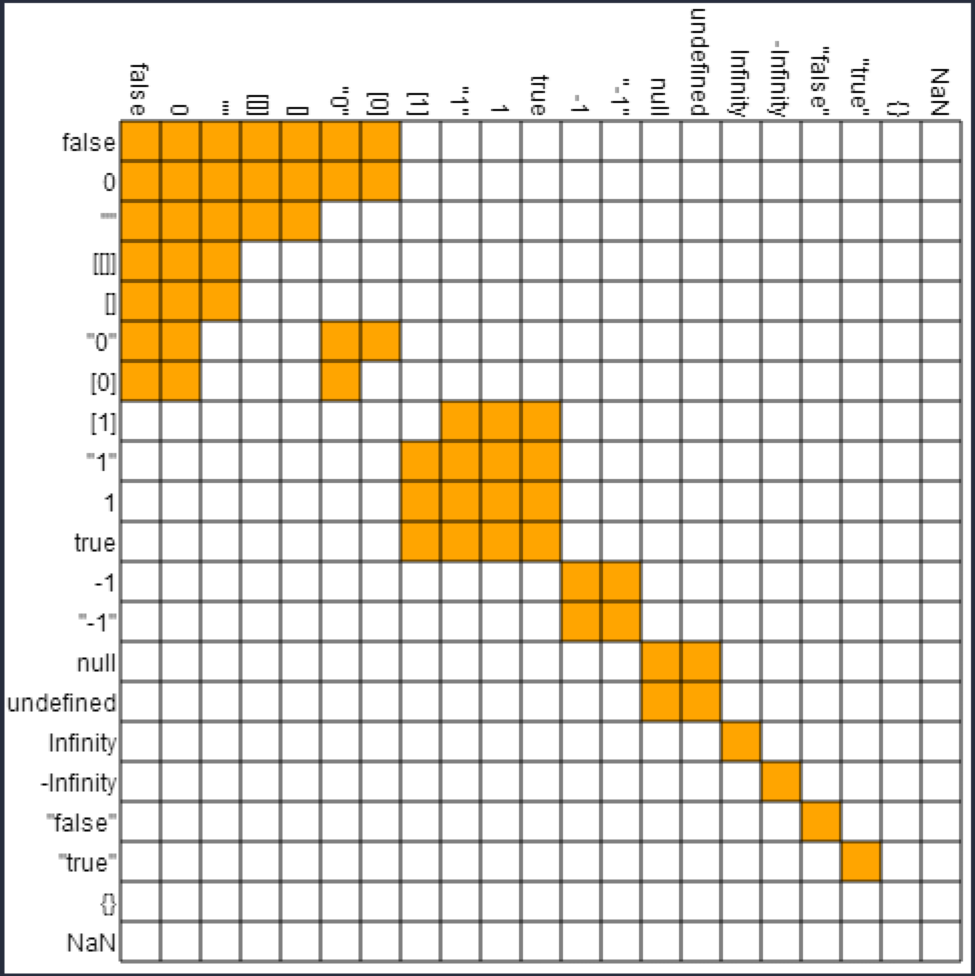
- Dynamically typed language
- Interpreted
- Sacrifices runtime speed and maintainability for development speed
  - Up to 1000 times slower than C
- Supports "Duck Typing"
- No encapsulation whatsoever
- Has really nice, concise syntax
  - List conjugations, decorators

> Python

- Advantages:
  - Very fast for throwing together a quick script
  - Lots of libraries (numpy, etc.)
  - Don't have to worry about making the types work
- If you use it for a large code base, be sure to use lots of unit tests

> JavaScript

- Weak, dynamically typed
- Runs on almost any device
- Used for cross-platform applications
- Has some "interesting" design decisions



	false	0	""	[[]]	[]	"0"	[0]	[1]	"1"	1	true	-1	"-1"	null	undefined	Infinity	-Infinity	"false"	"true"	{}	NaN
false	🟧	🟧	🟧	🟧	🟧	🟧	🟧														
0	🟧	🟧	🟧	🟧	🟧	🟧	🟧														
""	🟧	🟧	🟧	🟧	🟧																
[[]]	🟧	🟧	🟧																		
[]	🟧	🟧	🟧																		
"0"	🟧	🟧				🟧	🟧														
[0]	🟧	🟧				🟧															
[1]									🟧	🟧	🟧										
"1"								🟧	🟧	🟧	🟧										
1								🟧	🟧	🟧	🟧										
true								🟧	🟧	🟧	🟧										
-1												🟧	🟧								
"-1"												🟧	🟧								
null														🟧	🟧						
undefined														🟧	🟧						
Infinity																🟧					
-Infinity																	🟧				
"false"																		🟧			
"true"																			🟧		
{}																					
NaN																					

> JavaScript

- Advantages:
  - ...The only option for web applications

> Swift

- Announced by Apple in 2014
- "Protocol-oriented programming"
- Used for iOS development
- Uses reference counting
- optional types: Never dereference null again!

> OCaml

- A functional programming language similar to SML
- Used by Jane Street
- Static, Strongly-typed
- Eager

> Other Languages

- C#
- Haskel
- Kotlin
- TypeScript

Programming Languages

Attendance:
https://tinyurl.com/997431