# Programming Language Concepts

Madhavan Mukund, S P Suresh

Programming Language Concepts

Lecture 1, 7 January 2025

# What could this course be about?

Languages — Haskell, Python,

Java
C++
Javascript
Rust
Ruby
Perl
Go
⋮

Why so many?

Are they different?

## Why is Haskell different from Python?

| | |
|---|---|
| Functional | Object Oriented |
| Declarative | Imperative |
| What you want | How to do it |
| | Proof of correctness |
| | Control over resources |

Declarative

- λ calculus Untyped

Set of values + Allowed operations

↓

Typed λ calculus

↓

Type inference

Imperative

Storage – How / where, function calls ...

Passing values – referring to "remote" storage

Python: Mutable / Immutable

```
x = 5   ← immutable
y = x
y = y+2
x ?   5
```

```
l1 = [1,2,3]   ← mutable
l2 = l1
l2.append(4)
l1 ?   [1,2,3,4]
```

Pass values "by reference" – Side effects

Java

Rust – "ownership"

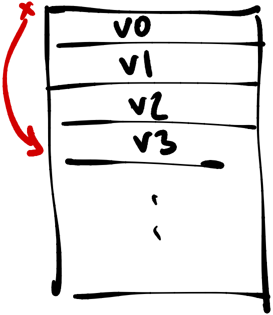
Object Oriented

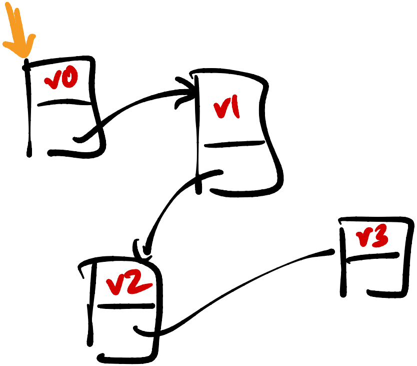
What are objects?

Data types – Scalar

Collections

Array





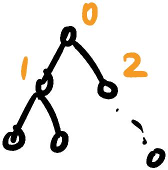
Heaps

delete_max → Implement as array

insert

$\text{children}(i) = 2i+1, 2i+2$

$\text{parent}(i) = \left\lfloor \frac{i-1}{2} \right\rfloor$



Data types ⇒ Data structures

↓

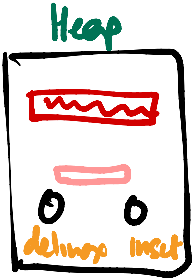
Do not allow

$a[j] = v$

Array → Heap

(executing) (delete_max, insert)

Creating a new data structure



O-O programming — disciplined use of "abstract" datatypes

Python has objects

No mechanism to hide

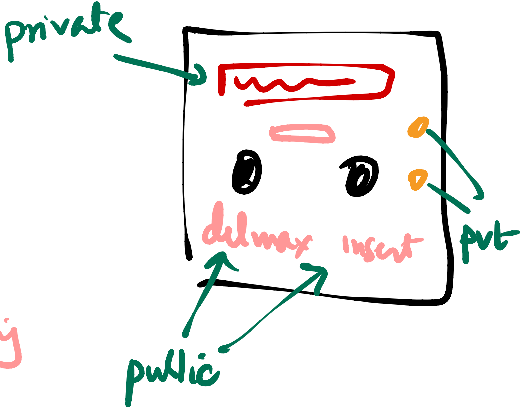
No declarations

$x = 7$ ← $x$ is int

⋮

$x =$ "abc" ← $x$ is String

"Hide" implementation



Haskell:

append :: [a] → a → [a]

Ord a

Imperative → generic functions

Generic programming

Exceptions – Exception "handling"

Concurrent programming

Shared data

Mechanisms for coordination

Next class Jan 16