**Announcements**

HW1 due Fri. Sept.

**HW1 Questions?**

**HW0 Stats**
- Hours Spent: 6.8 avg, 3.6 stddev, 6 median
- Hardness: 4.5 avg, 0.9 stddev, 2-6 range
  - Lots of use of Piazza, study groups, videos, old notes - great!
  - Q2 hard

**Lesson**

1. To find bugs, we need specifications
2. Models, we need predicates
Big Lesson

To reason about a program

(= is it doing the right thing? wrong thing?)

we must understand what the program means!

Semantics = "Meaning"

Plan: Study IMP = simple imperative programming language

Today: abstract syntax,

operational semantics

Later: other styles of semantics
Syntax: First step in designing PL

- how we write down programs

Concrete syntax: strings that a compiler/interpreter accepts

Well understood principles for turning concrete syntax to abstract syntax

Parsing

L-ordered representation - trees

IMP's abstract syntax

IMP's syntactic entities

- variable
- sets = syntactic class
- integers: \( n \in \mathbb{Z} \)
- booleans: \( t \in T = \{ \text{true}, \text{false} \} \)
- locations (variables): \( x, y \in \text{Loc} \)
- arithmetic expressions: \( a \in \text{Aexp} \)
- boolean expressions: \( b \in \text{Beq} \)
- commands: \( c \in \text{Com} \)
\[ a := n \quad (n \in \mathbb{Z}) \]
\[ x \quad (x \in \mathbb{R}) \]
\[ a_1 + a_2 \]
\[ a_1 - a_2 \]
\[ a_1 \cdot a_2 \]

\[ 3 + \text{myvar} \]
\[ (3 + 4) + 7 \]
\[ 3 + (4 + 7) \]

\[ b := B \]
\[ a_1 = a_2 \]
\[ a_1 \leq a_2 \]
\[ \neg b \]
\[ b_1 \land b_2 \]
\[ b_1 \lor b_2 \]

\[ (b_1 \in B_{\text{exp}}, \ b_2 \in B_{\text{exp}}) \]
\[ c ::= \quad \text{skip} \]
\[ \quad \text{x := a} \]
\[ \quad c_1 ; c_2 \]
\[ \quad \text{if } b \text{ then } c_1 \text{ else } c_2 \]
\[ \quad \text{while } b \text{ do } c \]

Ouch!

Evaluation Judgment

\[ \langle a, \sigma \rangle \Downarrow n \]

\[ \langle n, \sigma \rangle \Downarrow n \]