Motivation

- For language designers
  - Accurate, unambiguous goals; exploration of alternative paradigms; definition; standardization; PL research
- For language implementors
  - Specification of mechanisms and structures; guide for methods

Motivation

- For language learners
  - Provides understanding of model of computation (normative programming styles)
- For language users
  - Reference for “pushing the envelope”

Abstract Syntax

- Concrete syntax is about parsing a character string into a program
- Abstract syntax is about representing semantically meaningful structures
- The link between the two are parse trees

Example:

```
x := y + 2
```

The concrete tokens are: ident, assign-op, ident, plus, integer
Its parse tree could be:

```
x +
  /
 y 2
```
Abstract syntax

- Semantically we can remove all the concrete elements
- The semantics are clear even though all the terminal symbols from the concrete string have disappeared
- Thus the goal of abstract syntax is representation, not parsing

Example of an Imperative Language

- Abstract syntax (i.e., semantically meaningful)
  categories:
  - Programs
  - Commands
  - Expressions
  - Operators
  - Identifiers
  - Numerals
  - Booleans

  The grammar:
  - P ::= C
  - C ::= nop | C1;C2 | I := E | if B then C1 else C2 end | while B do C end
  - E ::= N | I | E1 OA E2
  - OA ::= plus | minus | mult | divide
  - B ::= true | false | I OR E
  - OR ::= equal | greater | less

  Example program:
  ```
  x := three;
  y := two;
  while x greater zero do
    if y equal x then
      nop
    else
      y := x minus one
    end
  end
  ```

Example of a Functional Language

- Syntactic categories:
  - Programs
  - Functions
  - Expressions
  - Identifiers
  - Numerals

  The grammar:
  - P ::= F1 F2...Fn E
  - F ::= I (I1,...,In) = E
  - E ::= N | I | E1 O E2 | if E1 then E2 else E3 |
  - O ::= plus | minus | mult | divide | equal | greater | less

  Example program:
  ```
  even(x) = if x then odd(x minus 1) else 1
  odd(x) = if x then even(x minus 1) else 0
  even(four)
  ```