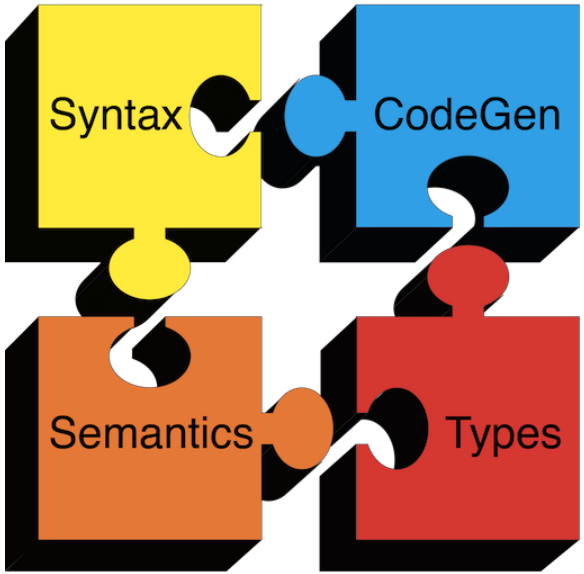


Compilers

Abstract Syntax Trees

Abstract Syntax Trees

- A parser traces the derivation of a sequence of tokens
- But the rest of the compiler needs a structural representation of the program
- Abstract syntax trees
  - Like parse trees but ignore some details
  - Abbreviated as AST

Abstract Syntax Trees

- Consider the grammar

  $E \rightarrow \text{int} \mid (\ E\ ) \mid E + E$

- And the string

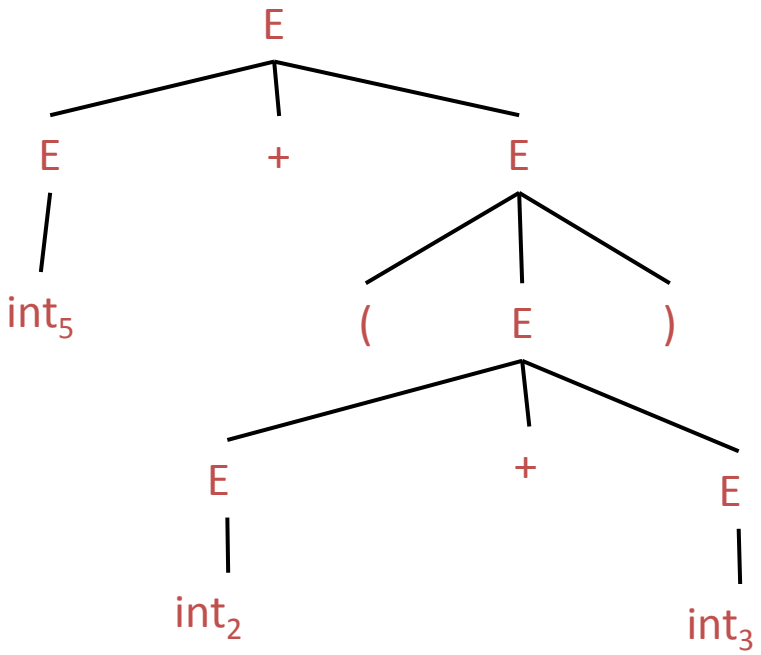
  5 + (2 + 3)

- After lexical analysis (a list of tokens)

  int_5 '+' '(' int_2 '+' int_3 ')'

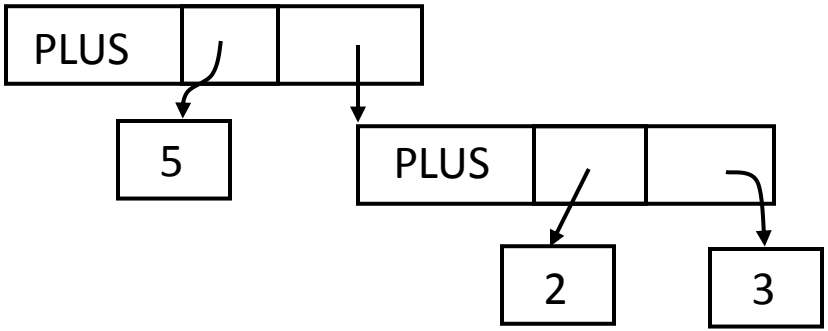
- During parsing we build a parse tree …

Abstract Syntax Trees



- A parse tree:
- Traces the operation of the parser
- Captures nesting structure
- But too much information
  - Parentheses
  - Single-successor nodes

Abstract Syntax Trees



- Also captures the nesting structure
- But abstracts from the concrete syntax

  => more compact and easier to use
- An important data structure in a compiler