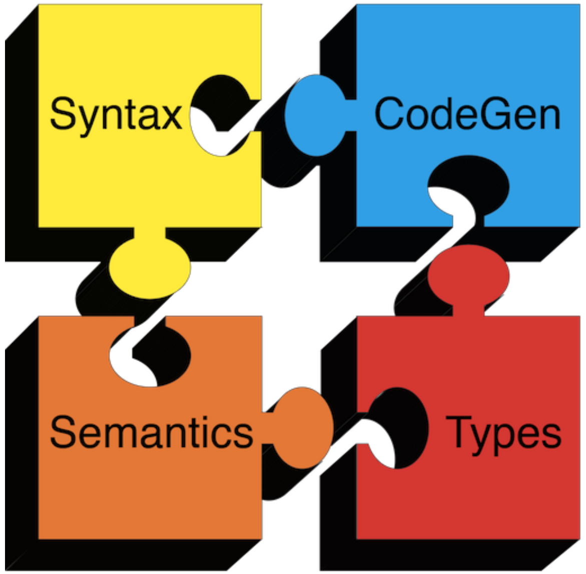


Compilers

Recursive Descent Parsing

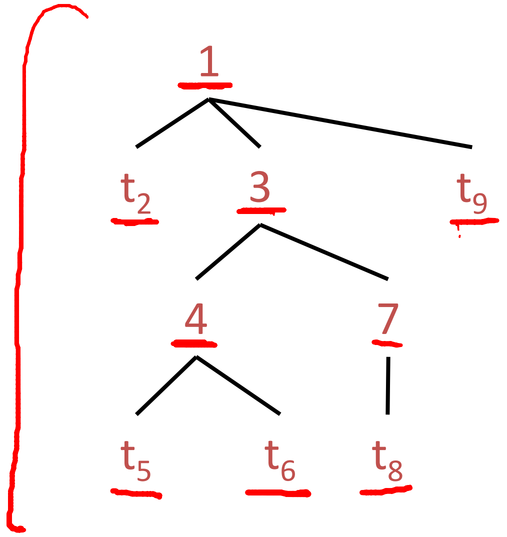
Alex Aiken

top-down

Recursive Descent

- The parse tree is constructed
  - From the top
  - From left to right
- Terminals are seen in order of appearance in the token stream:

  $t_2 \; t_5 \; t_6 \; t_8 \; t_9$



Recursive Descent

- Consider the grammar

  $E \rightarrow \underline{T} \mid \underline{T + E}$

  $T \rightarrow int \mid int * T \mid (E)$

- Token stream is: $\underline{(int_5)}$

- Start with top-level non-terminal $\underline{E}$
  - Try the rules for E in order

Recursive Descent

$E \rightarrow T \mid T + E$

$T \rightarrow \text{int} \mid \text{int} * T \mid (E)$

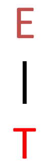
E

(int_5)

Recursive Descent

$E \rightarrow T \mid T + E$

$\underline{T \rightarrow \text{int}} \mid \text{int} * T \mid (E)$



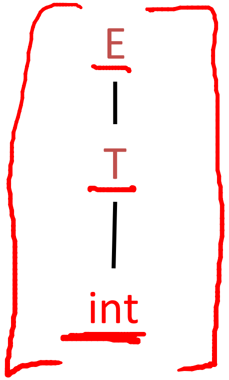
(int_5)

$\uparrow$

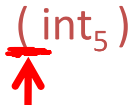
Recursive Descent

$E \rightarrow T \mid T + E$

$T \rightarrow \text{int} \mid \text{int} * T \mid (E)$



Mismatch: int does not match (
Backtrack …



Recursive Descent

$E \rightarrow T \mid T + E$

$T \rightarrow \text{int} \mid \underline{\text{int} * T} \mid (E)$

$$\begin{tikzcd} E \arrow[d, dash] \\ T \end{tikzcd}$$

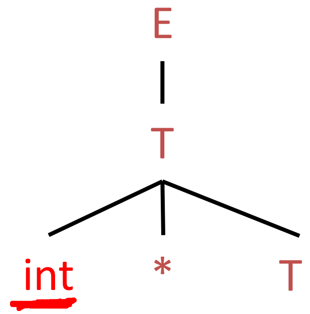
$(\ \text{int}_5 \ )$

$\uparrow$

Recursive Descent

$E \rightarrow T \mid T + E$

$T \rightarrow \text{int} \mid \text{int} * T \mid (E)$



Mismatch: int does not match (

Backtrack …

(int_5)

$\uparrow$

Recursive Descent

$$E \rightarrow T \mid T + E$$

$$T \rightarrow int \mid int * T \mid \underline{(E)}$$

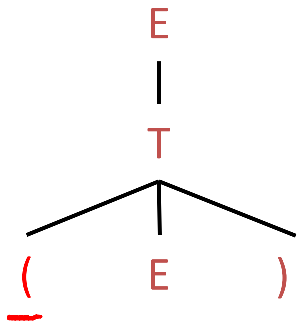
$$\begin{tikzcd} E \arrow[d, dash] \\ T \end{tikzcd}$$

$$\underset{\uparrow}{(}\ int_5\ )$$

Recursive Descent

$E \rightarrow T \mid T + E$

$T \rightarrow \text{int} \mid \text{int} * T \mid (E)$



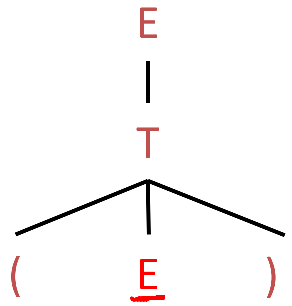
Match! Advance input.

(int_5)

Recursive Descent

$E \rightarrow \underline{T} \mid T + E$

$T \rightarrow \text{int} \mid \text{int} * T \mid (E)$

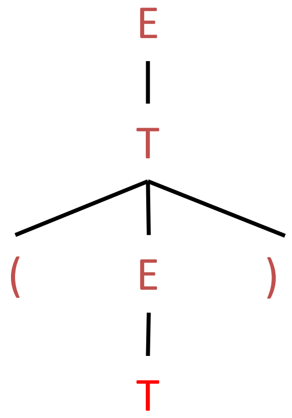


(int_5)

Recursive Descent

$E \rightarrow T \mid T + E$

$T \rightarrow \underline{\text{int}} \mid \text{int} * T \mid (E)$



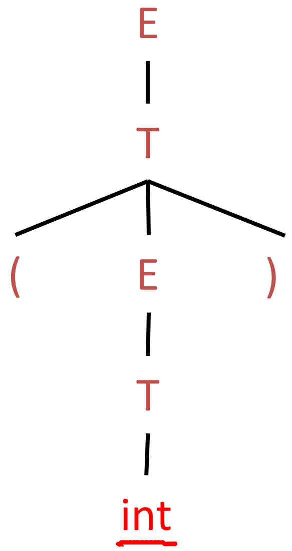
(int_5)

$\uparrow$

Recursive Descent

$E \rightarrow T \mid T + E$

$T \rightarrow \text{int} \mid \text{int} * T \mid (E)$



Match! Advance input.

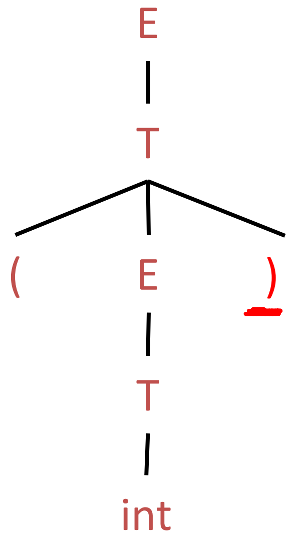
$(\underline{\text{int}}_5)$

$\uparrow$

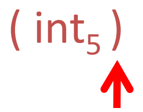
Recursive Descent

$E \rightarrow T \mid T + E$

$T \rightarrow \text{int} \mid \text{int} * T \mid (E)$



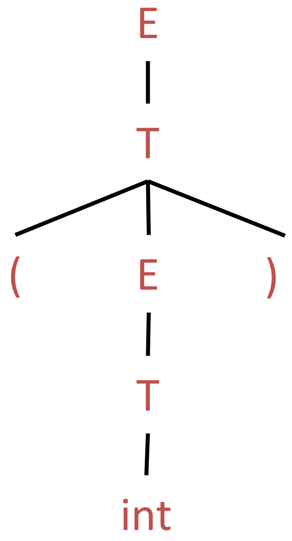
Match! Advance input.



Recursive Descent

$E \to T \mid T + E$

$T \to \text{int} \mid \text{int} * T \mid (E)$



End of input, accept.

(int_5) ↑

Recursive Descent

Choose the derivation that is a valid recursive descent parse for the string id + id in the given grammar. Moves that are followed by backtracking are given in red.

$$E \rightarrow E' \mid E' + E$$

$$E' \rightarrow -E' \mid id \mid (E)$$

○
E
E’
E’ + E
id + E
id + E’
id + id

○
E
E’
-E’
id
(E)
E’ + E
-E’ + E
id + E
id + E’
id + -E’
id + id

○
E
E’ + E
id + E
id + E’
id + id

[✓ ○]
E
E’
id
E’ + E
id + E
id + E’
id + id