CS3005D Compiler Design

Winter 2024

Lecture #34

Processing Declarations

Types and Relative Addresses for names

Saleena N
CSED NIT Calicut

April 2024

Declarations

int x; *float* y;

Attributes in the Symbol Table entries for x and y?

Declarations

int x; *float* y;

Attributes in the Symbol Table entries for x and y

Name, Type, Relative Address

Relative Address

Local declaration in $f()$: *int* x; *float* y;

Address relative to the base of the data area for $f()$

x at relative address 0, y at address?

Relative Address

Local declaration in $f()$: *int* x; *float* y;

x at relative address 0

y at $0 + size(int)$

Relative Address

int $x[5]$; *float* y;

Relative address of y?

Storage layout for local names

- The *width* of a type - number of storage units needed for objects of that type
- The amount of storage required at run time for a name can be determined based on its type
- Compiler assigns each name a relative address, stores type and relative address in the Symbol Table

Syntax-Directed Translation Scheme (SDT)

A notation for specifying a translation:

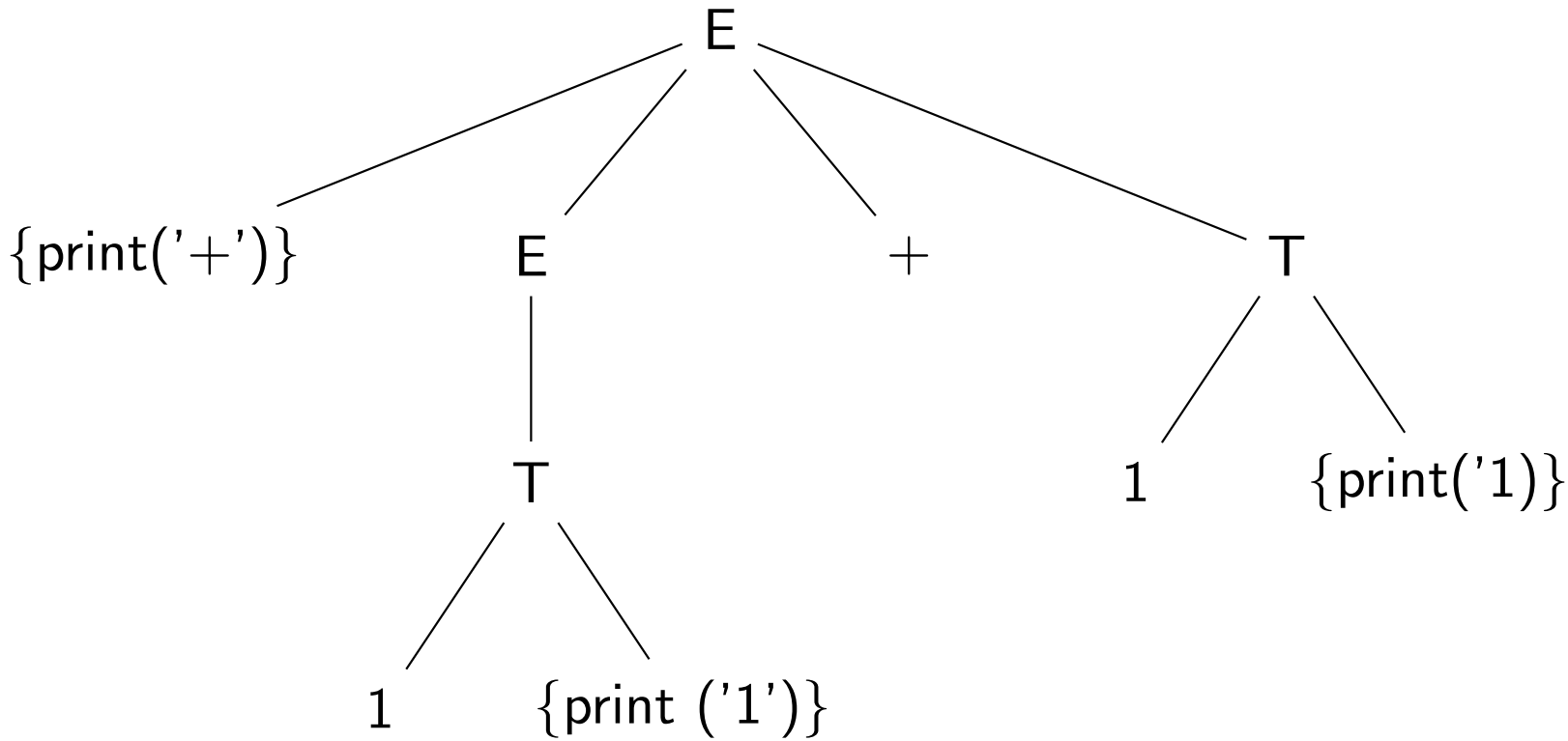
- a Context Free Grammar with *semantic actions* embedded within production bodies
- the order of evaluation of semantic rules is explicitly specified

$$
\begin{aligned}
E &\rightarrow \{print('+')\}\ E + T \\
E &\rightarrow T \\
T &\rightarrow 1\ \{print('1')\}
\end{aligned}
$$

$E \rightarrow \{print('+')\}\ E + T$

$E \rightarrow T$

$T \rightarrow 1\ \{print('1')\}$



Input: 1+1

Do a left-to-right depth-first traversal of the tree, when a leaf node for a semantic action is visited, execute the semantic action.

Postfix SDT - all actions at the right ends of the production bodies

$$E \rightarrow E + T \quad \{print('+')\}$$
$$E \rightarrow T$$
$$T \rightarrow \quad 1 \quad \{print('1')\}$$

Input: 1+1

Computing Types and their Widths

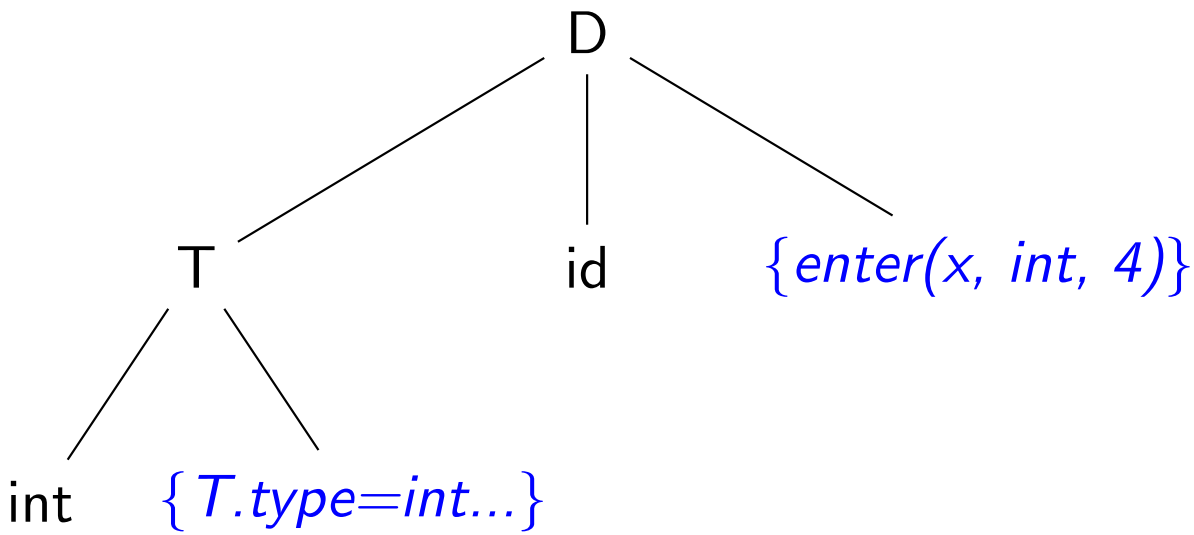
$D \rightarrow T\ id \quad \{enter(id.lexeme,\ T.type,\ T.width)\}$

$T \rightarrow int \quad \{T.type = int;\ T.width = 4\}$

$T \rightarrow float \quad \{T.type = float;\ T.width = 8\}$

Parse tree with semantic actions for int x ?

int x



Sequence of Declarations

$P \rightarrow D$

$D \rightarrow D; D$

$D \rightarrow T\ id$

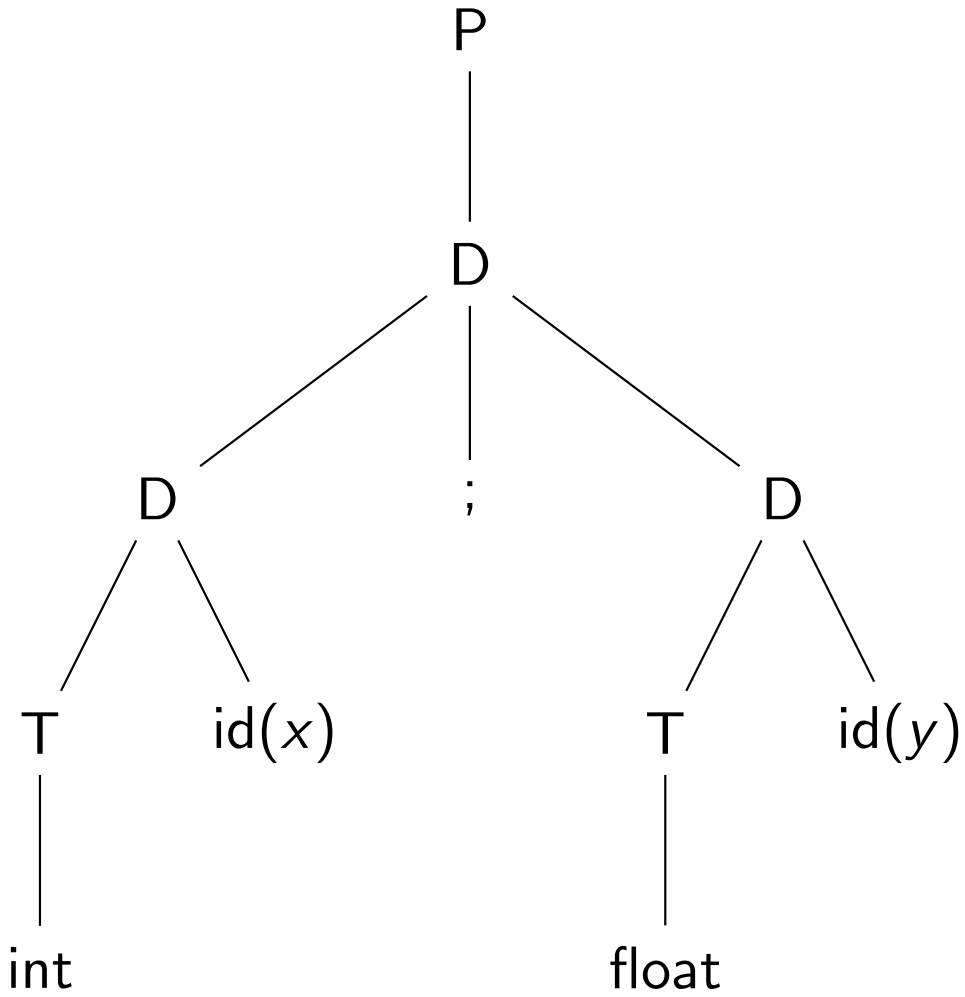
$T \rightarrow int$

$T \rightarrow float$

Draw parse tree for

int x; float y

int x; float y



Assigning Relative Addresses to Names

$P \rightarrow \{offset = 0\}\ D$

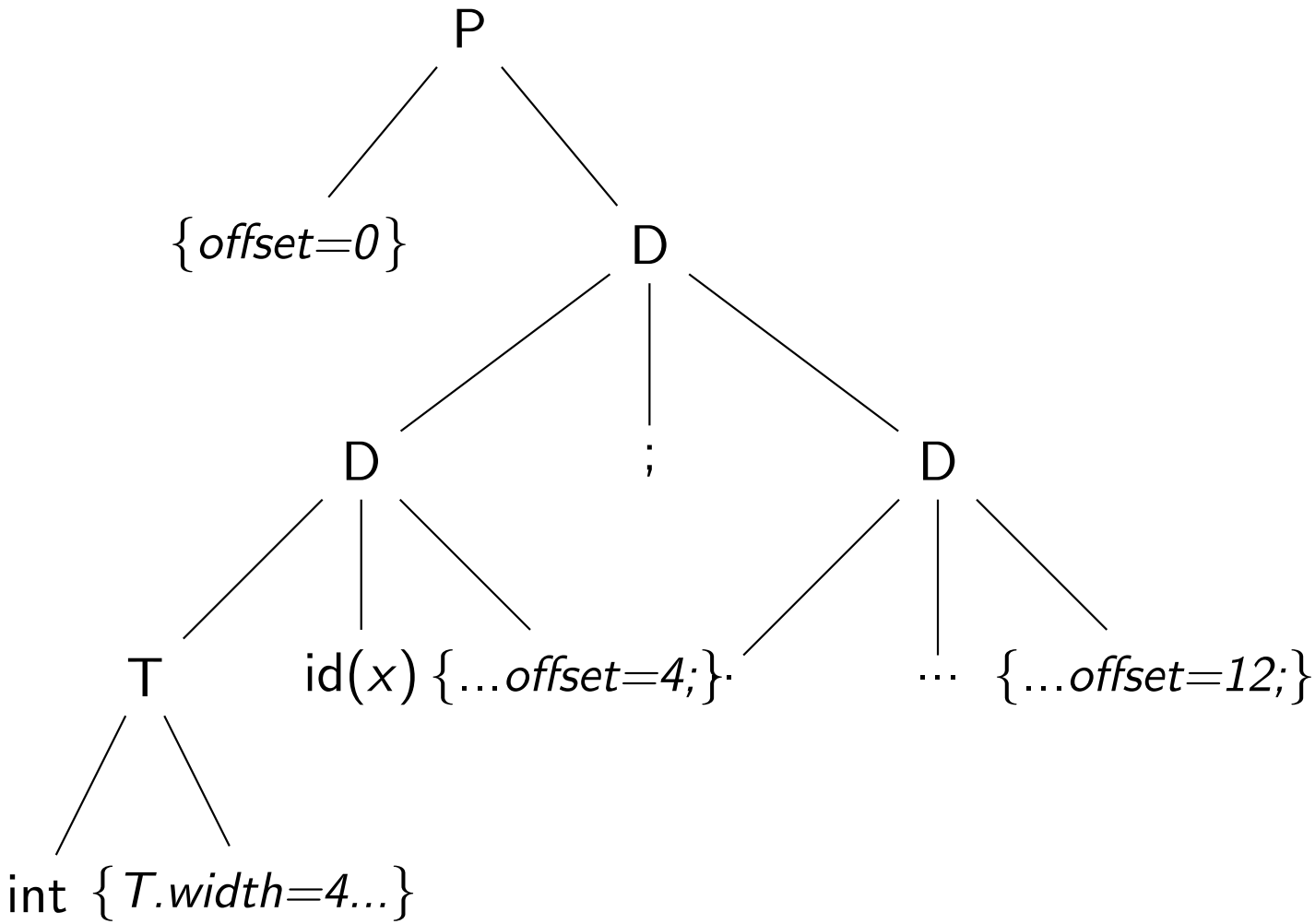
$D \rightarrow D; D$

$D \rightarrow T\ id\ \{enter(id.lexeme,\ T.type,\ offset);$
$offset = offset + T.width)\}$

$T \rightarrow int\ \{T.type = int;\ T.width = 4\}$

$T \rightarrow float\ \{T.type = float;\ T.width = 8\}$

int x; float y



Arrays

int [2] x;
float [3] [2] y;

Type Expression?

Width?

Arrays

$int\ [2]\ x;$

Type Expression: $array(2,\ int)$

Width: 2×4

Arrays

float [3] [2] y;

Type Expression: $array(3,\ array(2,\ float))$

Width: $3 \times 2 \times 8$

Array Declaration: one-dimensional integer array

$$T \rightarrow int\ [num]\ \ \{T.type = array(num.lexval,\ int);\ T.width = num.lexval \times 4\ \}$$

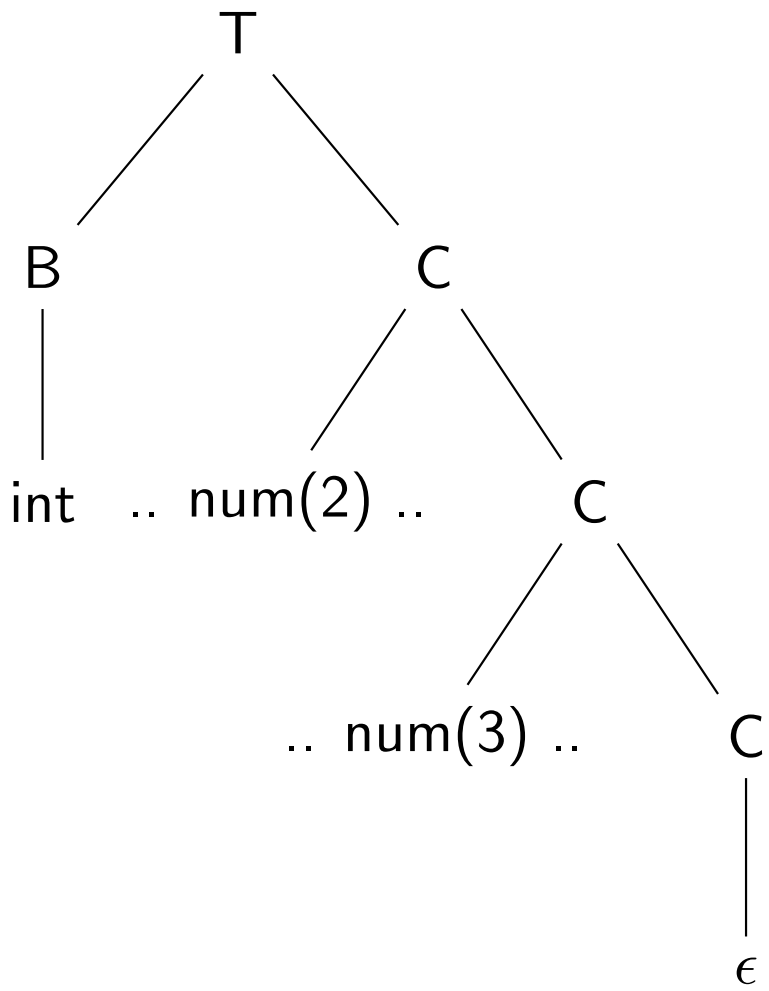
Parse Tree with semantic actions for *int* [3] ?

Array Declaration: general

$T \rightarrow B\ C$
$B \rightarrow int \mid float$
$C \rightarrow [num]\ C \mid \epsilon$

Draw the parse tree for *int* [2] [3]

int [2] [3]



Array Declaration: computing type and width

$$T \rightarrow B \ \{t = B.type; w = B.width\} \ \ C$$

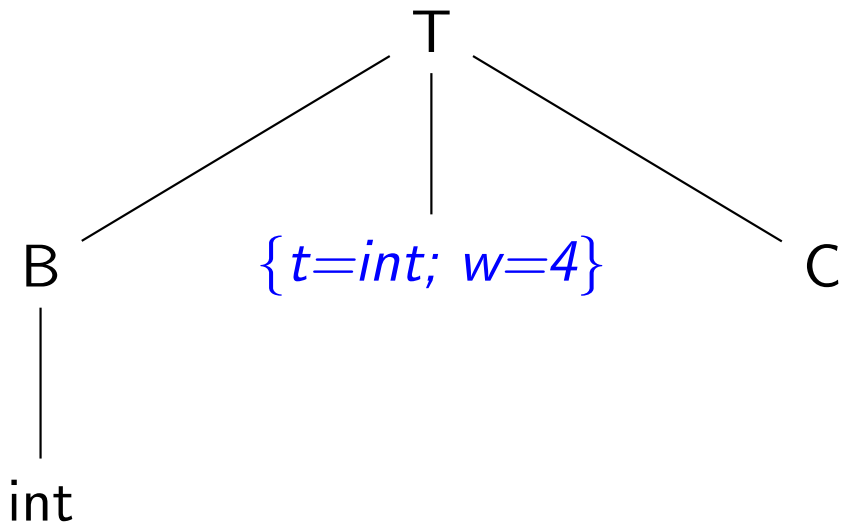
$$B \rightarrow int \ \ \{B.type = int; B.width = 4\}$$

$$B \rightarrow float \ \ \{B.type = float; B.width = 8\}$$

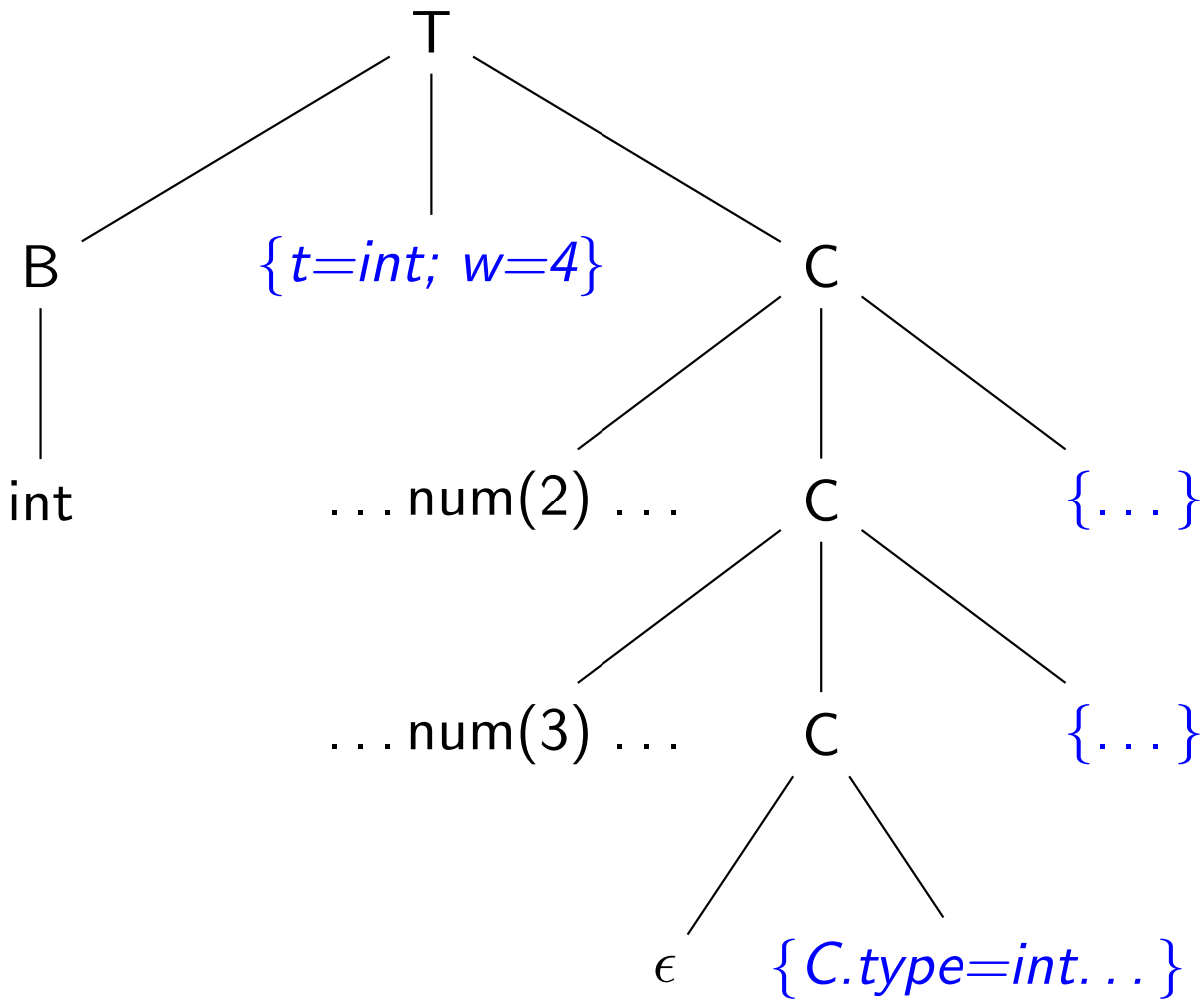
$$C \rightarrow [num] \ C_1 \quad \{C.type = array(num.lexval, \ C_1.type) \\ C.width = num.lexval \times C_1.width\}$$

$$C \rightarrow \epsilon \quad \{C.type = t; C.width = w\}$$

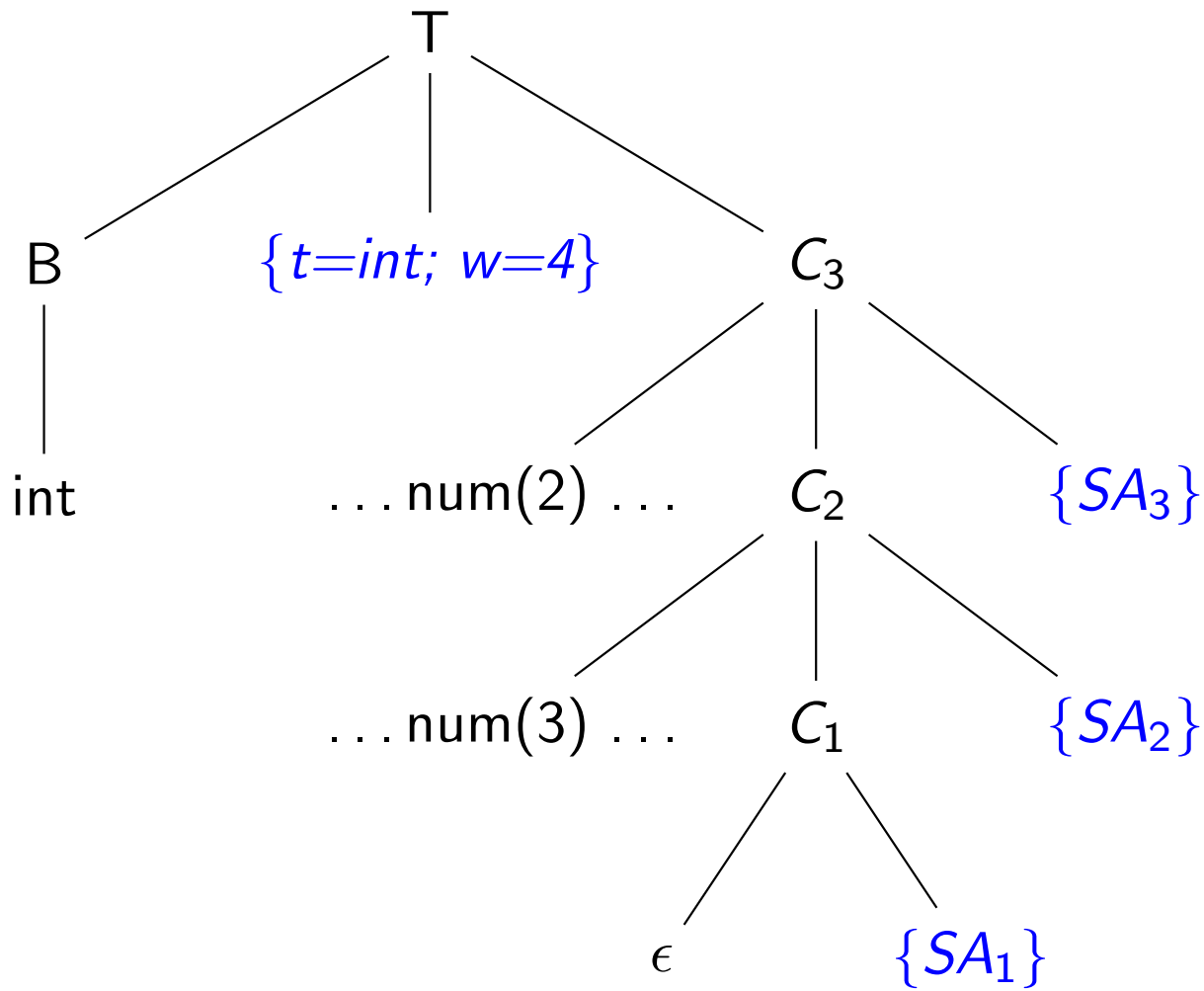
int [2] [3]



int [2] [3]



int [2] [3]



SA_1 : $\{C_1.type = int; C_1.width = 4\}$

SA_2 : $\{C_2.type = array(3,\ int); C_2.width = 12\}$

SA_3 : $\{C_3.type = array(2,\ array(3,\ int)); C_3.width = 24\}$

Array Declaration: computing type and width

$$T \rightarrow B \ \{t = B.type; w = B.width\} \ \ C$$

$$B \rightarrow int \ \ \{B.type = int; B.width = 4\}$$

$$B \rightarrow float \ \ \{B.type = float; B.width = 8\}$$

$$C \rightarrow [num] \ C_1 \ \ \{C.type = array(num.lexval, \ C_1.type) \\ C.width = num.lexval \times C_1.width\}$$

$$C \rightarrow \epsilon \ \ \{C.type = t; C.width = w\}$$

Semantic Actions for setting $T.type$ and $T.width$?

References

References:

- Aho A.V., Lam M.S., Sethi R., and Ullman J.D. Compilers: Principles, Techniques, and Tools (ALSU). Pearson Education, 2007.

Further reading:

- ALSU Section 6.3