

The Structure of a Type

AST: Abstract Syntax tree



Class Cell {
int info; cell next;...}

int [2] [3]



Types

$D \rightarrow T \text{ id} ; D \mid \epsilon$

$T \rightarrow B C \mid \text{record } \{ D \}$

$B \rightarrow \text{int} \mid \text{float}$

$C \rightarrow \epsilon \mid [\text{num}] C$

boolean
char
float
void

② Array

③ Node

④ Record

int [num] [num] x
float d ;

SDT for variable width and offset calculation

$$P \rightarrow \{ \text{offset} = 0; \}$$
 D
$$D \rightarrow \underline{T \text{ id} ;} \quad \{ \text{top.put}(\text{id.lexeme}, T.\text{type}, \text{offset}); \\ \text{offset} = \text{offset} + T.\text{width}; \}$$
 D_1
$$D \rightarrow \epsilon$$
$$T \rightarrow \begin{matrix} B \\ C \end{matrix}$$

```
{ t = B.type; w = B.width; }
T.type=C.type;T.width=c.width
```

```
B.type=float;
B.width=8
```

$$B \rightarrow \text{int}$$

```
{ B.type = integer; B.width = 4; }
```

$$B \rightarrow \text{float}$$

```
{ B.type = float; B.width = 8; }
```

$$C \rightarrow \epsilon$$
$$\{ C.type = t; C.width = w; \}$$
$$C \rightarrow [\text{num}] C_1 \quad \begin{array}{l} C.\text{type} = \\ \quad \{ \text{array}(\text{num.value}, C_1.\text{type}); \\ \quad \quad C.\text{width} = \text{num.value} \times C_1.\text{width}; \} \end{array}$$

```
Input: float x;  
int [2] [3] y;
```



offset = 8

32

✗	7	6	0
5	4		2

```
T.type=float;
T.width=8
```

```
B.type=float;
B.width=8
```

```
C.type=float;
C.width=8
```

```
T.type= Array(2,Array(3,int))
T.width=24
```

```
C.Type = Array(2,Array(3,int))
c.width=24
```

```
B.type=int;
B.width=4
```

```
C.Type = Array(3,int)
c.width=12
```