17 April 2025
Thursday

Sequence of Declaration:

- Refers to how declarations are handeled with a grammar's production rules.

- P produces { Offset = 0 } D
  আগে offset = 0 set হচ্ছে পরে D সাইডে যাচ্ছি।

- D produces Tid ; Top নামক stack এ id lexeme, T এর type রাখছে . offser রাখছে, offset value calculate করার পরে D সাইডে যাচ্ছি।

# এই sequence matter করে কার পরে কে আসছে।

# Production Rule কোনটা কোন sequence এ declare হয়েছে সেটা importanl.

P → { offset = 0 } D

D → T id ; { top.push(id.lexeme T.type , offset) ; offset = offset + T.width;} D_1

D → ϵ

T → B C { T.type = C.type ; T. width = C.width }

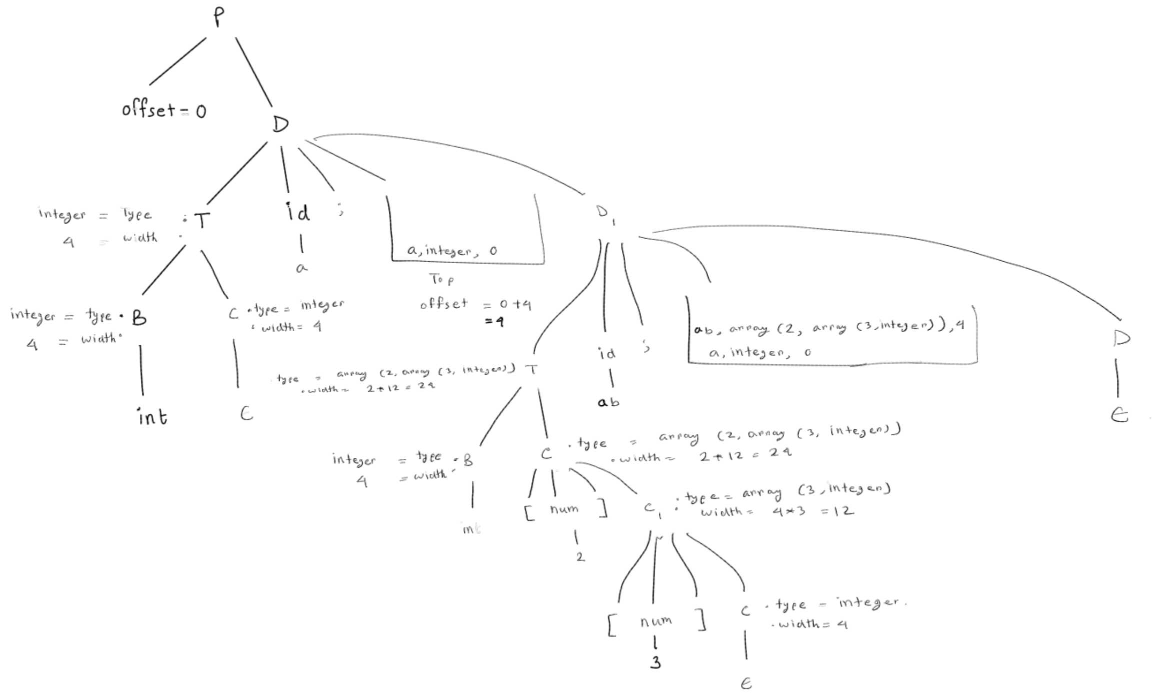
B → int { B.type = integer ; B.width = 4;}

C → [num] C_1 { C.type = array (num.value, C_1.type); C.width = C_1 width * num.value ;}

C → ϵ { C.type = B.type , C.width = B.width }

input: int a;
int [2] [3] ab ;

Generate the Type expression for the input string.
⇒



Record Statement

$D \rightarrow T\,id\,;\,D \mid \epsilon$

$T \rightarrow Bc \mid record\ '\{'\ D\ '\}'$

$B \rightarrow int \mid float$ {put.

pop

$C \rightarrow \epsilon \mid [num]\ c$

record is one kind of structure in c Language.

c int ab {

}

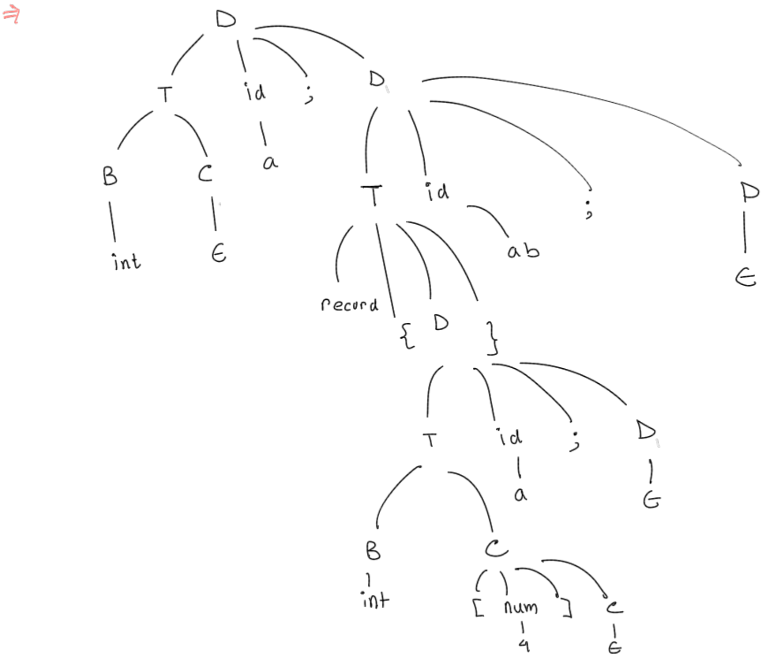
given টাতে record structure এর name ab.

input string:

```
int a ;
record {
    int [4] a ;
  } ab ;
```

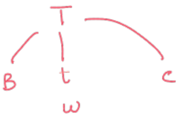
Generate Type expression for the input string.

⇒



Dummy Production Rule:

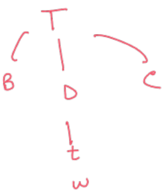
$T \rightarrow B\ \{t = B.type,\ w = B.width\}\ c$ → Original Production Rule.



Dummy production rule use করে লিখতে চাচ্ছি,

$T \rightarrow BDc$

$D \rightarrow \{t = B.type\ \ w - B.width\}$

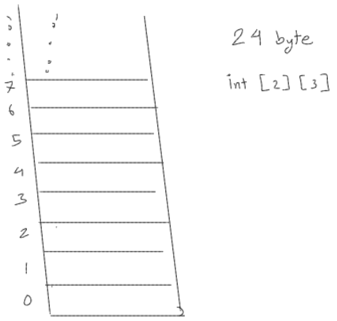


The application of types can be grouped under checking and translation.

1. **Type checking** uses logical rules to reason about the behaviour of a program at run time. Specifically, it ensures that, the types of operands match the type expected by an operator. For example, && operator in Java expects it's two operands to be boolean. and the result is also boolean.

array (2, array (3, integer))

2. **Translation application:** From the type of a name, a compiler can determine the storage, that will be needed for that name at runtime. Type information is also needed to calculate the address denoted by an array reference to insert explicit type conversions, and to choose the right version of an arithmetic operators among other things.


24 byte
int [2] [3]
7
6
5
4
3
2
1
0