CC Lecture 22

Prepared for: 7th Sem, CE, DDU

Prepared by: Niyati J. Buch

Example 2

```
N \rightarrow L.L
L \rightarrow BL \mid B
B \rightarrow 0 \mid 1
```

- AS(N)= AS(B) ={val个:real}
- AS(L) ={cnt个:integer, val个:real}

```
1. N \rightarrow L_1.L_2 \{N.val = L_1.val + (L_2.val / 2^L_2.cnt)\}
```

- 2. $L \rightarrow BL_1$ {L.cnt=L₁.cnt+1; L.val=L₁.val+(B.val* 2^L₁.cnt)}
- 3. $L \rightarrow B$ {L.cnt = 1; L.val = B.val}
- 4. $B \rightarrow 0$ {B.val = 0}
- 5. $B \rightarrow 1$ {B.val = 1}

Binary to Decimal

Example 2 (second method)

```
Binary to Decimal
   N \rightarrow L.L
   L \rightarrow LB \mid B
   B \rightarrow 0|1
AS(N)={val个:real}

    AS(L) = AS(B) ={cnt↑:integer, val↑:real}

1. N \rightarrow L_1.L_2 {N.val = L_1.val + (L_2.val / 2^L_2.cnt)}
2. L \rightarrow L_1B {L.cnt=L_1.cnt+1; L.val=L_1.val*2 + B.val}
3. L \rightarrow B {L.cnt = 1; L.val = B.val}
4. B \rightarrow 0 {B.val = 0}
5. B \rightarrow 1 {B.val = 1}
```

Example 2 (second method)

```
1. N \rightarrow L_1.L_2 {N.val = L_1.val + (L_2.val / 8^L_2.cnt)} Octal to Decimal
2. L \rightarrow L_1B {L.cnt=L_1.cnt+1; L.val=L_1.val*8 + B.val}
3. L \rightarrow B
                  {L.cnt = 1 ; L.val = B.val}
4. B \rightarrow 0 {B.val = 0}
5. B \rightarrow 1 {B.val = 1}
6. B \rightarrow 2
                  \{B.val = 2\}
7. B \rightarrow 3
                  \{B.val = 3\}
8. B \rightarrow 4
                  \{B.val = 4\}
                  \{B.val = 5\}
9. B \rightarrow 5
10. B → 6
                  \{B.val = 6\}
11. B \rightarrow 7
                   \{B.val = 7\}
```

Example 2 (second method)

```
1. N \rightarrow L_1.L_2 {N.val = L_1.val + (L_2.val / 16^L_2.cnt)}
                                                                                   Hexadecimal to Decimal
2. L \rightarrow L_1B
                        \{L.cnt=L_1.cnt+1; L.val=L_1.val*16 + B.val\}
3. L \rightarrow B
                        \{L.cnt = 1 ; L.val = B.val\}
     B \rightarrow 0
4.
                        \{B.val = 0\}
                                                    12. B \rightarrow 8
                                                                            \{B.val = 8\}
5. B \rightarrow 1
                        \{B.val = 1\}
                                                    13. B \rightarrow 9
                                                                            \{B.val = 9\}
6.
     B \rightarrow 2
                        \{B.val = 2\}
                                                    14. B \rightarrow a
                                                                            {B.val = 10}
7. B \rightarrow 3
                        \{B.val = 3\}
                                                    15. B \rightarrow b
                                                                            {B.val = 11}
8. B \rightarrow 4
                        \{B.val = 4\}
                                                    16. B \rightarrow c
                                                                            \{B.val = 12\}
9. B \rightarrow 5
                        \{B.val = 5\}
                                                    17. B \rightarrow d
                                                                            {B.val = 13}
10. B \rightarrow 6
                        \{B.val = 6\}
                                                    18. B \rightarrow e
                                                                            \{B.val = 14\}
11. B \rightarrow 7
                        \{B.val = 7\}
                                                    19. B \rightarrow f
                                                                            {B.val = 15}
```

Examples so far

- 1. $L = \{a^nb^nc^n \mid n \ge 1\}$ for input aabbcc
- 2. Binary to decimal (octal to decimal, hexadecimal to decimal)
- 3. Simple calculator
- 4. Infix to postfix

```
E \rightarrow E + T
E \rightarrow T
T \rightarrow T * F
T \rightarrow F
F \rightarrow X ^ F
F \rightarrow X
```

 $X \rightarrow num$

```
E \rightarrow E_1 + T {E.value = E_1.value + T.value}

E \rightarrow T {E.value = T.value}

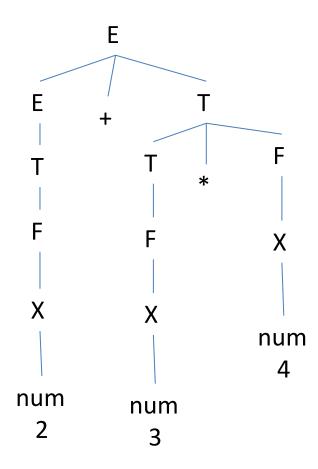
T \rightarrow T_1 * F {T.value = T_1.value * F.value}

T \rightarrow F {T.value = F.value}

F \rightarrow X ^F_1 {F.value = X.value ^ F_1.value}

F \rightarrow X {F.value = X.value}

X \rightarrow num {X.value = num.lexvalue}
```



```
E \rightarrow E_1 + T {E.value = E_1.value + T.value} Parse tree???

E \rightarrow T {E.value = T.value}

T \rightarrow T_1 * F {T.value = F_1.value * F.value}

T \rightarrow F {T.value = F.value}

F \rightarrow X \wedge F_1 {F.value = X.value F_1.value}

F \rightarrow X {F.value = X.value}

F \rightarrow X {X.value = num.lexvalue}
```

```
E \rightarrow E_1 + T {E.value = E_1.value + T.value}

E \rightarrow T {E.value = T.value}

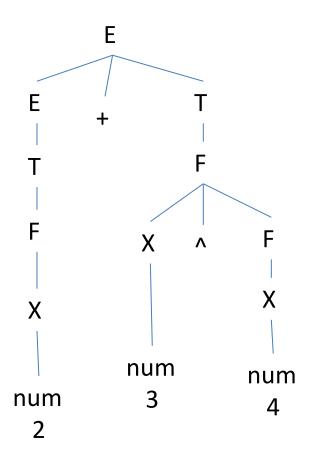
T \rightarrow T_1 * F {T.value = T_1.value * F.value}

T \rightarrow F {T.value = F.value}

F \rightarrow X \land F_1 {F.value = X.value \uparrow F_1.value}

F \rightarrow X {F.value = X.value}

X \rightarrow num {X.value = num.lexvalue}
```



```
E \rightarrow E_1 + T {E.value = E_1.value + T.value}

E \rightarrow T {E.value = T.value}

T \rightarrow T_1 * F {T.value = T_1.value * F.value}

T \rightarrow F {T.value = F.value}

F \rightarrow X \wedge F_1 {F.value = X.value T_1.value}

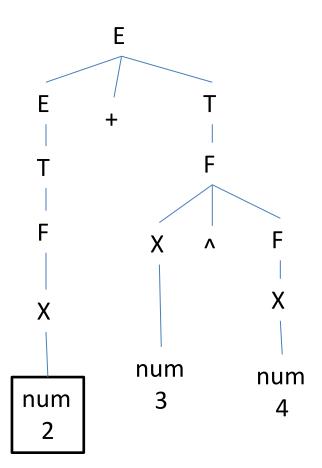
T \rightarrow T_1 * F {T.value = F.value}

T \rightarrow T_1 * F {T.value = F.value}

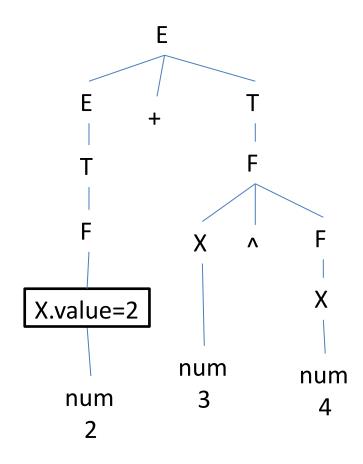
T \rightarrow T_1 * F {T.value = F.value}

T \rightarrow T_1 * F {F.value = F.value}

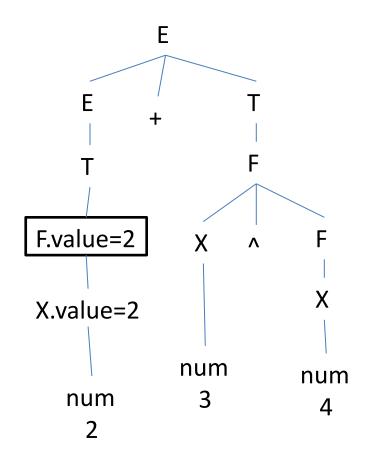
T \rightarrow T_1 * F {F.value = F.value}
```



```
E \rightarrow E_1 + T \qquad \{E.value = E_1.value + T.value\}
E \rightarrow T \qquad \{E.value = T.value\}
T \rightarrow T_1 * F \qquad \{T.value = T_1.value * F.value\}
T \rightarrow F \qquad \{T.value = F.value\}
F \rightarrow X \land F_1 \qquad \{F.value = X.value \land F_1.value\}
F \rightarrow X \qquad \{F.value = X.value\}
X \rightarrow num \qquad \{X.value = num.lexvalue\}
```



```
E \rightarrow E_1 + T \qquad \{E.value = E_1.value + T.value\}
E \rightarrow T \qquad \{E.value = T.value\}
T \rightarrow T_1 * F \qquad \{T.value = T_1.value * F.value\}
T \rightarrow F \qquad \{T.value = F.value\}
F \rightarrow X \land F_1 \qquad \{F.value = X.value \land F_1.value\}
F \rightarrow X \qquad \{F.value = X.value\}
X \rightarrow num \qquad \{X.value = num.lexvalue\}
```



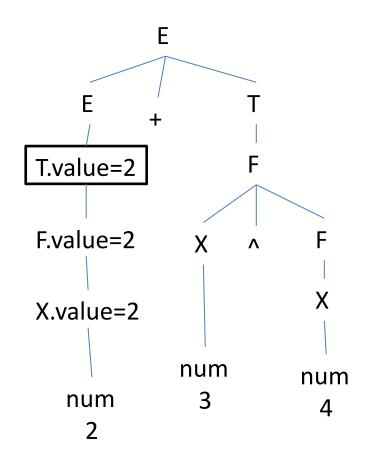
```
E \rightarrow E_1 + T {E.value = E_1.value + T.value}

E \rightarrow T {E.value = T.value}

T \rightarrow T_1 * F {T.value = E_1.value * F.value}

E \rightarrow T {T.value = E_1.value}

E \rightarrow T {T.value = E_1.value}
```



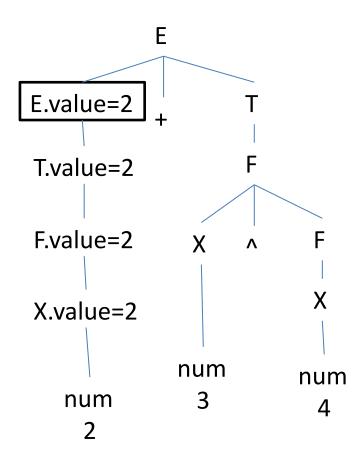
```
E \rightarrow E_1 + T {E.value = E_1.value + T.value}

E \rightarrow T {E.value = T.value}

T \rightarrow T_1 * F {T.value = E_1.value * F.value}

E \rightarrow T {T.value = E_1.value}

E \rightarrow T {T.value = E_1.value}
```



```
E \rightarrow E_1 + T {E.value = E_1.value + T.value}

E \rightarrow T {E.value = T.value}

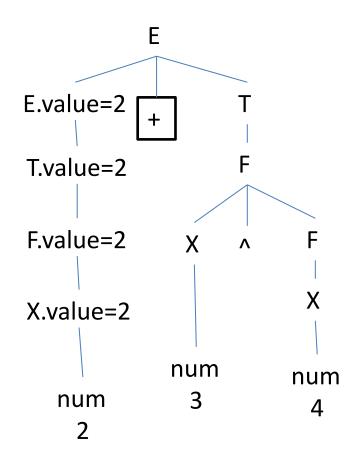
T \rightarrow T_1 * F {T.value = T_1.value * F.value}

T \rightarrow F {T.value = F.value}

F \rightarrow X \land F_1 {F.value = X.value \uparrow F_1.value}

F \rightarrow X {F.value = X.value}

X \rightarrow num {X.value = num.lexvalue}
```



```
E \rightarrow E_1 + T {E.value = E_1.value + T.value}

E \rightarrow T {E.value = T.value}

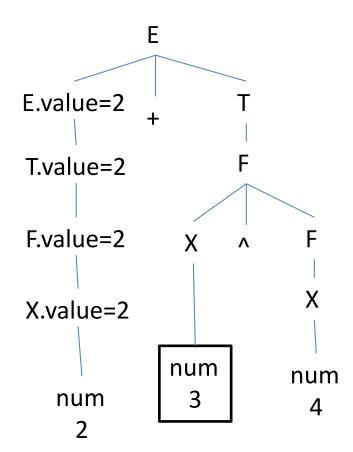
T \rightarrow T_1 * F {T.value = T_1.value * F.value}

T \rightarrow F {T.value = F.value}

F \rightarrow X \land F_1 {F.value = X.value \uparrow F_1.value}

F \rightarrow X {F.value = X.value}

X \rightarrow num {X.value = num.lexvalue}
```



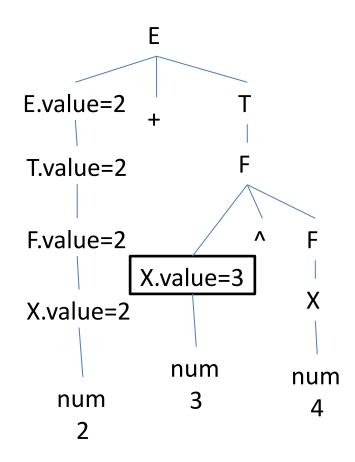
```
E \rightarrow E_1 + T {E.value = E_1.value + T.value}

E \rightarrow T {E.value = T.value}

T \rightarrow T_1 * F {T.value = E_1.value * F.value}

E \rightarrow T {T.value = E_1.value}

E \rightarrow T {T.value = E_1.value}
```



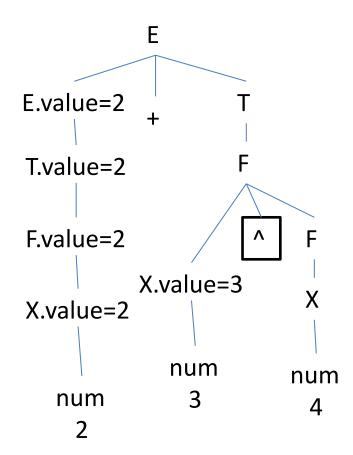
```
E \rightarrow E_1 + T {E.value = E_1.value + T.value}

E \rightarrow T {E.value = T.value}

T \rightarrow T_1 * F {T.value = E_1.value * F.value}

E \rightarrow T {T.value = E_1.value}

E \rightarrow T {T.value = E_1.value}
```



```
E \rightarrow E_1 + T {E.value = E_1.value + T.value}

E \rightarrow T {E.value = T.value}

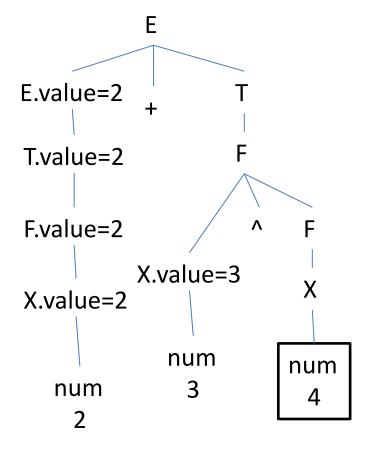
T \rightarrow T_1 * F {T.value = T_1.value * F.value}

T \rightarrow F {T.value = F.value}

F \rightarrow X \land F_1 {F.value = X.value \uparrow F_1.value}

F \rightarrow X {F.value = X.value}

X \rightarrow num {X.value = num.lexvalue}
```



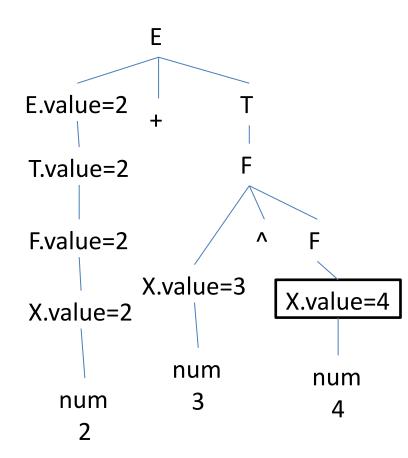
```
E \rightarrow E_1 + T {E.value = E_1.value + T.value}

E \rightarrow T {E.value = T.value}

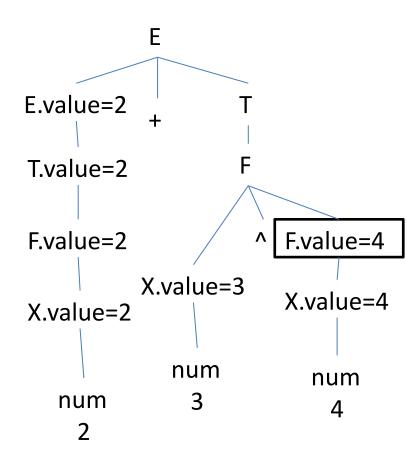
T \rightarrow T_1 * F {T.value = E_1.value * F.value}

E \rightarrow T {F.value = E_1.value}

E \rightarrow T {T.value = E_1.value}
```



```
E \rightarrow E_1 + T \qquad \{E.value = E_1.value + T.value\}
E \rightarrow T \qquad \{E.value = T.value\}
T \rightarrow T_1 * F \qquad \{T.value = T_1.value * F.value\}
T \rightarrow F \qquad \{T.value = F.value\}
F \rightarrow X \land F_1 \qquad \{F.value = X.value \land F_1.value\}
F \rightarrow X \qquad \{F.value = X.value\}
X \rightarrow num \qquad \{X.value = num.lexvalue\}
```



```
E \rightarrow E_1 + T {E.value = E_1.value + T.value}

E \rightarrow T {E.value = T.value}

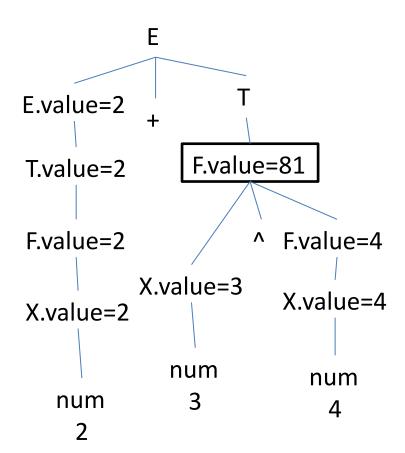
T \rightarrow T_1 * F {T.value = T_1.value * F.value}

T \rightarrow F {T.value = F.value}

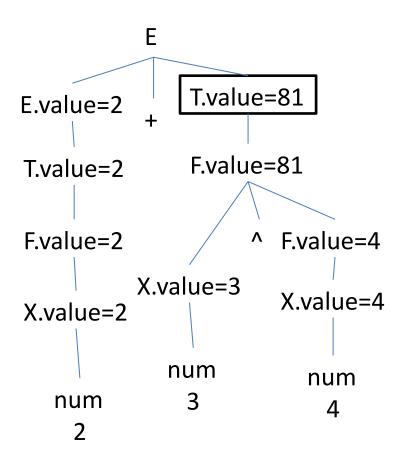
F \rightarrow X \land F_1 {F.value = X.value \land F_1.value}

F \rightarrow X {F.value = X.value}

X \rightarrow num {X.value = num.lexvalue}
```

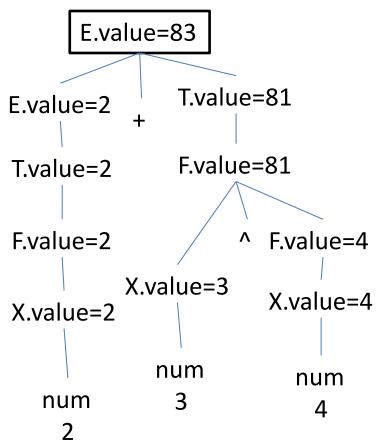


```
E \rightarrow E_1 + T \qquad \{E.value = E_1.value + T.value\}
E \rightarrow T \qquad \{E.value = T.value\}
T \rightarrow T_1 * F \qquad \{T.value = T_1.value * F.value\}
T \rightarrow F \qquad \{T.value = F.value\}
F \rightarrow X \wedge F_1 \qquad \{F.value = X.value \wedge F_1.value\}
F \rightarrow X \qquad \{F.value = X.value\}
X \rightarrow num \qquad \{X.value = num.lexvalue\}
```



```
E \rightarrow E_1 + T \qquad \{E.value = E_1.value + T.value\}
E \rightarrow T \qquad \{E.value = T.value\}
T \rightarrow T_1 * F \qquad \{T.value = T_1.value * F.value\}
T \rightarrow F \qquad \{T.value = F.value\}
F \rightarrow X \wedge F_1 \qquad \{F.value = X.value \wedge F_1.value\}
F \rightarrow X \qquad \{F.value = X.value\}
X \rightarrow num \qquad \{X.value = num.lexvalue\}
```

For input, 2 + 3 ^ 4 Output: 83



```
E \rightarrow E_1 + T {E.value = E_1.value + T.value} Parse tree???

E \rightarrow T {E.value = T.value}

T \rightarrow T_1 * F {T.value = F_1.value * F.value}

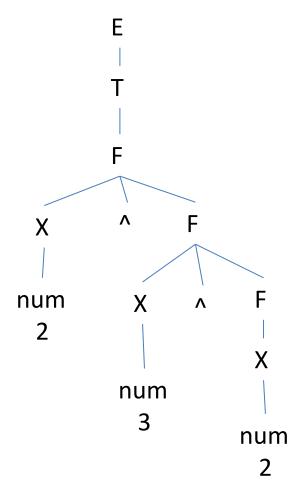
T \rightarrow F {T.value = F.value}

F \rightarrow X \land F_1 {F.value = X.value \hdots F_1.value}

F \rightarrow X {F.value = X.value}

X \rightarrow num {X.value = num.lexvalue}
```

```
E \rightarrow E_1 + T \qquad \{E.value = E_1.value + T.value\}
E \rightarrow T \qquad \{E.value = T.value\}
T \rightarrow T_1 * F \qquad \{T.value = T_1.value * F.value\}
T \rightarrow F \qquad \{T.value = F.value\}
F \rightarrow X \wedge F_1 \qquad \{F.value = X.value \wedge F_1.value\}
F \rightarrow X \qquad \{F.value = x.value\}
X \rightarrow \text{num} \qquad \{X.value = \text{num.lexvalue}\}
```



```
E \rightarrow E_1 + T {E.value = E_1.value + T.value}

E \rightarrow T {E.value = T.value}

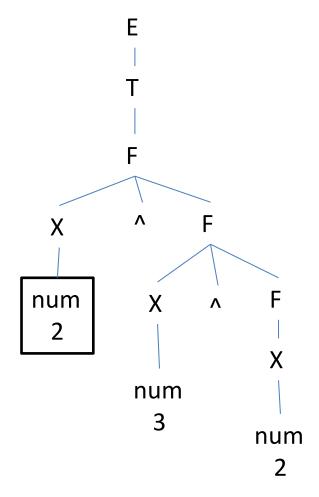
T \rightarrow T_1 * F {T.value = T_1.value * F.value}

T \rightarrow F {T.value = F.value}

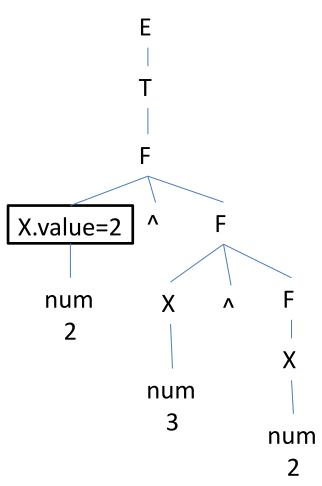
F \rightarrow X \land F_1 {F.value = X.value \uparrow F_1.value}

F \rightarrow X {F.value = X.value}

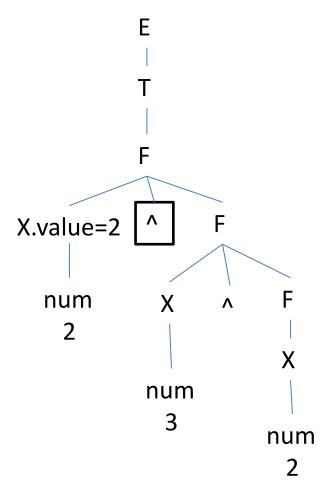
X \rightarrow num {X.value = num.lexvalue}
```



```
E \rightarrow E_1 + T \qquad \{E.value = E_1.value + T.value\}
E \rightarrow T \qquad \{E.value = T.value\}
T \rightarrow T_1 * F \qquad \{T.value = T_1.value * F.value\}
T \rightarrow F \qquad \{T.value = F.value\}
F \rightarrow X \wedge F_1 \qquad \{F.value = X.value \wedge F_1.value\}
F \rightarrow X \qquad \{F.value = X.value\}
X \rightarrow num \qquad \{X.value = num.lexvalue\}
```



```
E \rightarrow E_1 + T \qquad \{E.value = E_1.value + T.value\}
E \rightarrow T \qquad \{E.value = T.value\}
T \rightarrow T_1 * F \qquad \{T.value = T_1.value * F.value\}
T \rightarrow F \qquad \{T.value = F.value\}
F \rightarrow X \wedge F_1 \qquad \{F.value = X.value \wedge F_1.value\}
F \rightarrow X \qquad \{F.value = X.value\}
X \rightarrow num \qquad \{X.value = num.lexvalue\}
```



```
E \rightarrow E_1 + T {E.value = E_1.value + T.value}

E \rightarrow T {E.value = T.value}

T \rightarrow T_1 * F {T.value = T_1.value * F.value}

T \rightarrow F {T.value = F.value}

F \rightarrow X \land F_1 {F.value = X.value \uparrow F_1.value}

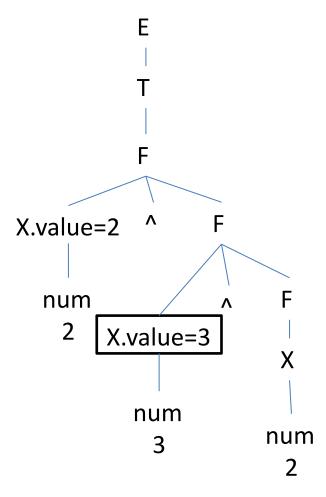
F \rightarrow X {F.value = X.value}

X \rightarrow num {X.value = num.lexvalue}
```

For input, 2 ^ 3 ^ 2

2

```
E \rightarrow E_1 + T \qquad \{E.value = E_1.value + T.value\}
E \rightarrow T \qquad \{E.value = T.value\}
T \rightarrow T_1 * F \qquad \{T.value = T_1.value * F.value\}
T \rightarrow F \qquad \{T.value = F.value\}
F \rightarrow X \land F_1 \qquad \{F.value = X.value \land F_1.value\}
F \rightarrow X \qquad \{F.value = X.value\}
X \rightarrow num \qquad \{X.value = num.lexvalue\}
```



```
E \rightarrow E_1 + T {E.value = E_1.value + T.value}

E \rightarrow T {E.value = T.value}

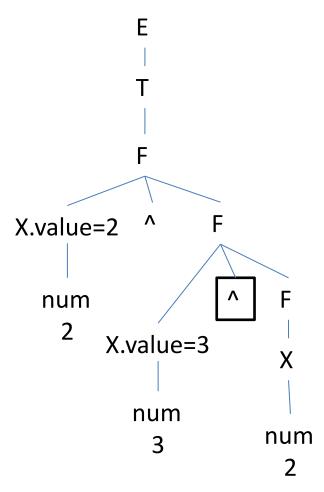
T \rightarrow T_1 * F {T.value = T_1.value * F.value}

T \rightarrow F {T.value = F.value}

F \rightarrow X \land F_1 {F.value = X.value \uparrow F_1.value}

F \rightarrow X {F.value = X.value}

X \rightarrow num {X.value = num.lexvalue}
```



```
E \rightarrow E_1 + T {E.value = E_1.value + T.value}

E \rightarrow T {E.value = T.value}

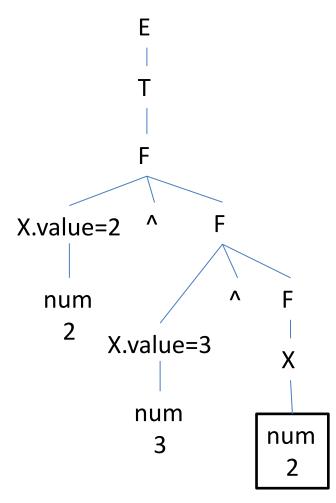
T \rightarrow T_1 * F {T.value = T_1.value * F.value}

T \rightarrow F {T.value = F.value}

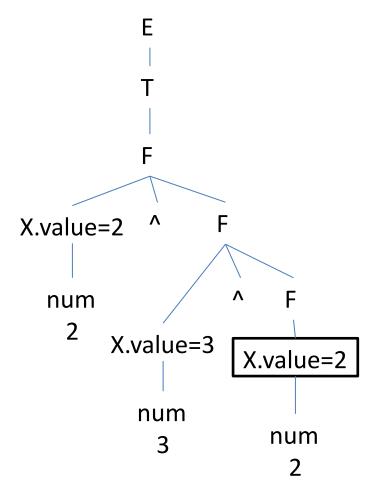
F \rightarrow X \land F_1 {F.value = X.value \uparrow F_1.value}

F \rightarrow X {F.value = X.value}

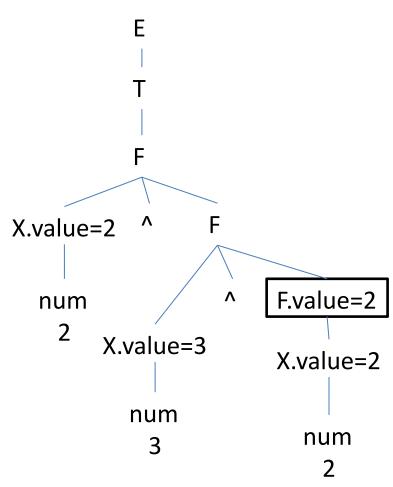
X \rightarrow num {X.value = num.lexvalue}
```



```
E \rightarrow E_1 + T \qquad \{E.value = E_1.value + T.value\}
E \rightarrow T \qquad \{E.value = T.value\}
T \rightarrow T_1 * F \qquad \{T.value = T_1.value * F.value\}
T \rightarrow F \qquad \{T.value = F.value\}
F \rightarrow X \wedge F_1 \qquad \{F.value = X.value \wedge F_1.value\}
F \rightarrow X \qquad \{F.value = X.value\}
X \rightarrow num \qquad \{X.value = num.lexvalue\}
```

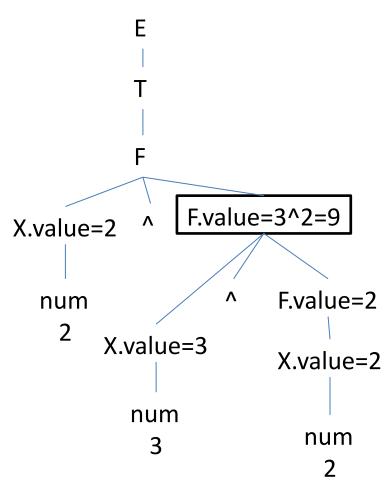


```
E \rightarrow E_1 + T \qquad \{E.value = E_1.value + T.value\}
E \rightarrow T \qquad \{E.value = T.value\}
T \rightarrow T_1 * F \qquad \{T.value = T_1.value * F.value\}
T \rightarrow F \qquad \{T.value = F.value\}
F \rightarrow X \land F_1 \qquad \{F.value = X.value \land F_1.value\}
F \rightarrow X \qquad \{F.value = X.value\}
X \rightarrow num \qquad \{X.value = num.lexvalue\}
```



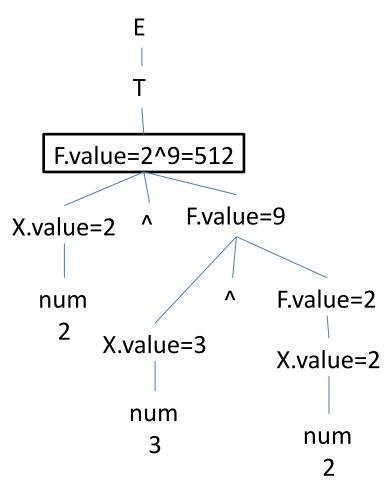
```
E \rightarrow E_1 + T \qquad \{E.value = E_1.value + T.value\}
E \rightarrow T \qquad \{E.value = T.value\}
T \rightarrow T_1 * F \qquad \{T.value = T_1.value * F.value\}
T \rightarrow F \qquad \{T.value = F.value\}
F \rightarrow X \wedge F_1 \qquad \{F.value = X.value \wedge F_1.value\}
F \rightarrow X \qquad \{F.value = X.value\}
X \rightarrow num \qquad \{X.value = num.lexvalue\}
```

For input, 2 ^ 3 ^ 2



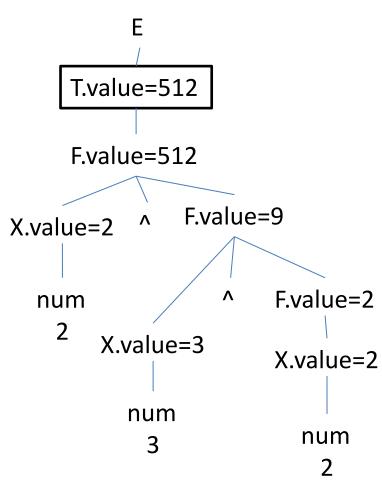
```
E \rightarrow E_1 + T \qquad \{E.value = E_1.value + T.value\}
E \rightarrow T \qquad \{E.value = T.value\}
T \rightarrow T_1 * F \qquad \{T.value = T_1.value * F.value\}
T \rightarrow F \qquad \{T.value = F.value\}
F \rightarrow X \wedge F_1 \qquad \{F.value = X.value \wedge F_1.value\}
F \rightarrow X \qquad \{F.value = X.value\}
X \rightarrow num \qquad \{X.value = num.lexvalue\}
```

For input, 2 ^ 3 ^ 2



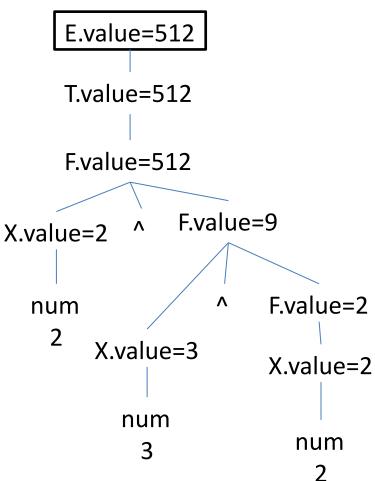
```
E \rightarrow E_1 + T \qquad \{E.value = E_1.value + T.value\}
E \rightarrow T \qquad \{E.value = T.value\}
T \rightarrow T_1 * F \qquad \{T.value = T_1.value * F.value\}
T \rightarrow F \qquad \{T.value = F.value\}
F \rightarrow X \land F_1 \qquad \{F.value = X.value \land F_1.value\}
F \rightarrow X \qquad \{F.value = X.value\}
X \rightarrow num \qquad \{X.value = num.lexvalue\}
```

For input, 2 ^ 3 ^ 2



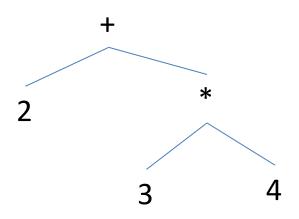
```
E \rightarrow E_1 + T \qquad \{E.value = E_1.value + T.value\}
E \rightarrow T \qquad \{E.value = T.value\}
T \rightarrow T_1 * F \qquad \{T.value = T_1.value * F.value\}
T \rightarrow F \qquad \{T.value = F.value\}
F \rightarrow X \land F_1 \qquad \{F.value = X.value \land F_1.value\}
F \rightarrow X \qquad \{F.value = X.value\}
X \rightarrow num \qquad \{X.value = num.lexvalue\}
```

For input, 2 ^ 3 ^ 2 Output: 512

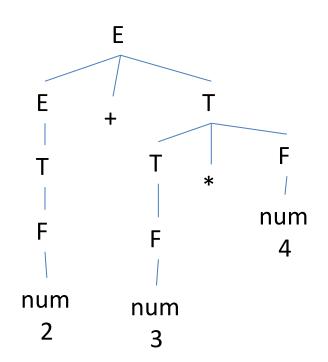


SDT to build a syntax tree

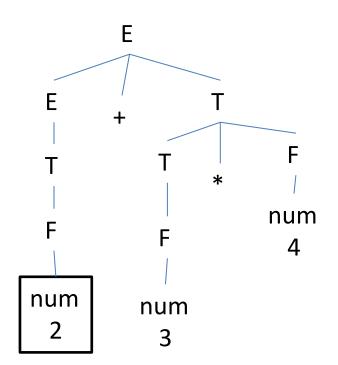
SDT to build a syntax tree



```
    SDT to build a syntax tree
    E → E + T {E.nptr=mknode(E.nptr, '+', T.nptr);}
    E → T {E.nptr=T.nptr;}
    T → T * F {T.nptr=mknode(T.nptr, *, F.nptr);}
    T→F {T.nptr=F.nptr}
    F→ num {F.nptr=mknode(null, idname, null);}
```



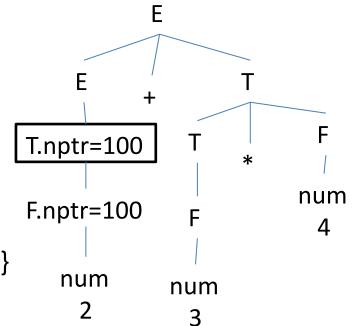
```
    SDT to build a syntax tree
    E → E + T {E.nptr=mknode(E.nptr, '+', T.nptr);}
    E → T {E.nptr=T.nptr;}
    T → T * F {T.nptr=mknode(T.nptr, *, F.nptr);}
    T→F {T.nptr=F.nptr}
    F→ num {F.nptr=mknode(null, idname, null);}
```



```
Ε
   SDT to build a syntax tree
E \rightarrow E + T \{E.nptr=mknode(E.nptr, '+', T.nptr);\}
           {E.nptr=T.nptr;}
                                                                                      F
E \rightarrow T
T \rightarrow T * F \{T.nptr=mknode(T.nptr, *, F.nptr);\}
                                                                                    num
                                                         F.nptr=100
           {T.nptr=F.nptr}
T \rightarrow F
                                                                                      4
F→ num {F.nptr=mknode(null, idname, null);}
                                                            num
                                                                       num
                                                              2
                                                                         3
 For input, 2 + 3 * 4
```

100				
	null	2	null	

```
    SDT to build a syntax tree
    E → E + T {E.nptr=mknode(E.nptr, '+', T.nptr);}
    E → T {E.nptr=T.nptr;}
    T → T * F {T.nptr=mknode(T.nptr, *, F.nptr);}
    T → F {T.nptr=F.nptr}
    F→ num {F.nptr=mknode(null, idname, null);}
```

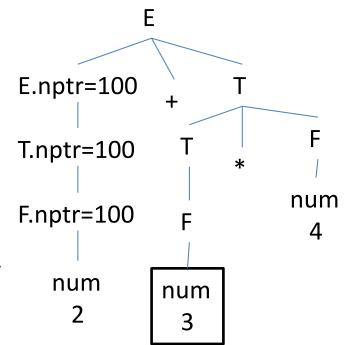


100)		
	null	2	null

```
Ε
   SDT to build a syntax tree
                                                      E.nptr=100
E→ E + T {E.nptr=mknode(E.nptr, '+', T.nptr);}
                                                                                  F
E \rightarrow T {E.nptr=T.nptr;}
                                                      T.nptr=100
T \rightarrow T * F \{T.nptr=mknode(T.nptr, *, F.nptr);\}
                                                                                num
                                                      F.nptr=100
T \rightarrow F
           {T.nptr=F.nptr}
                                                                                  4
F→ num {F.nptr=mknode(null, idname, null);}
                                                         num
                                                                   num
                                                                     3
```

100 null 2 null

SDT to build a syntax tree
 E → E + T {E.nptr=mknode(E.nptr, '+', T.nptr);}
 E → T {E.nptr=T.nptr;}
 T → T * F {T.nptr=mknode(T.nptr, *, F.nptr);}
 T→F {T.nptr=F.nptr}
 F→ num {F.nptr=mknode(null, idname, null);}





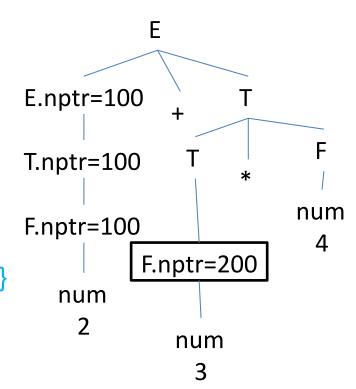
```
    SDT to build a syntax tree
```

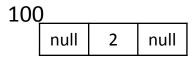
$$E \rightarrow E + T \{E.nptr=mknode(E.nptr, '+', T.nptr);\}$$

$$E \rightarrow T$$
 {E.nptr=T.nptr;}

$$T \rightarrow F$$
 {T.nptr=F.nptr}

F→ num {F.nptr=mknode(null, idname, null);}







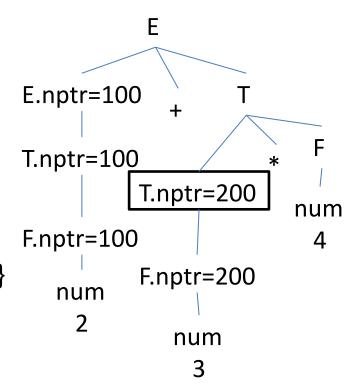
SDT to build a syntax tree

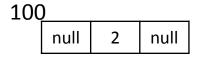
$$E \rightarrow E + T \{E.nptr=mknode(E.nptr, '+', T.nptr);\}$$

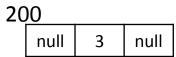
$$E \rightarrow T$$
 {E.nptr=T.nptr;}

$$T \rightarrow F$$
 {T.nptr=F.nptr}

F→ num {F.nptr=mknode(null, idname, null);}







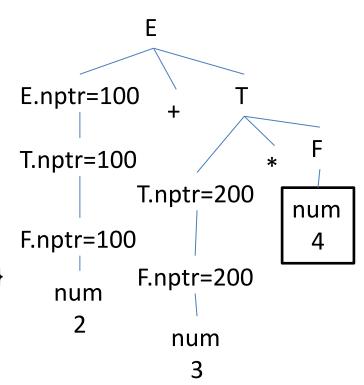
SDT to build a syntax tree

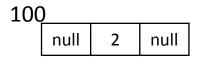
$$E \rightarrow E + T \{E.nptr=mknode(E.nptr, '+', T.nptr);\}$$

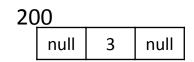
$$E \rightarrow T$$
 {E.nptr=T.nptr;}

$$T \rightarrow F$$
 {T.nptr=F.nptr}

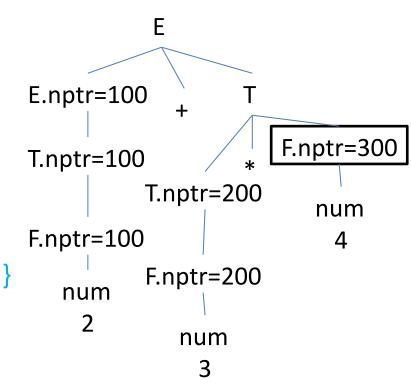
F→ num {F.nptr=mknode(null, idname, null);}

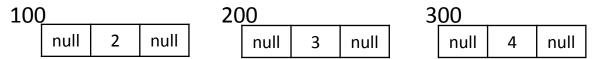




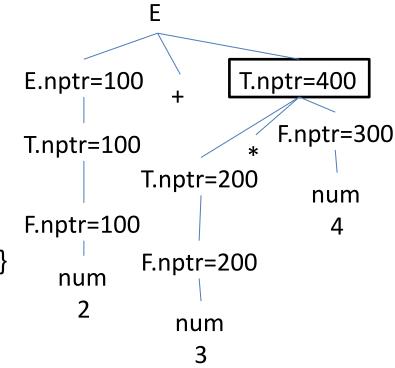


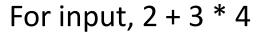
```
    SDT to build a syntax tree
    E → E + T {E.nptr=mknode(E.nptr, '+', T.nptr);}
    E → T {E.nptr=T.nptr;}
    T → T * F {T.nptr=mknode(T.nptr, *, F.nptr);}
    T→F {T.nptr=F.nptr}
    F→ num {F.nptr=mknode(null, idname, null);}
```

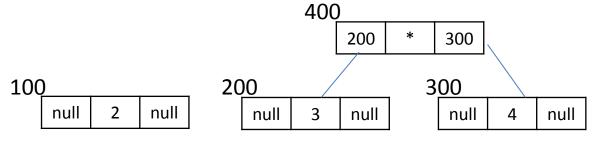




```
    SDT to build a syntax tree
    E → E + T {E.nptr=mknode(E.nptr, '+', T.nptr);}
    E → T {E.nptr=T.nptr;}
    T → T * F {T.nptr=mknode(T.nptr, *, F.nptr);}
    T → F {T.nptr=F.nptr}
    F → num {F.nptr=mknode(null, idname, null);}
```

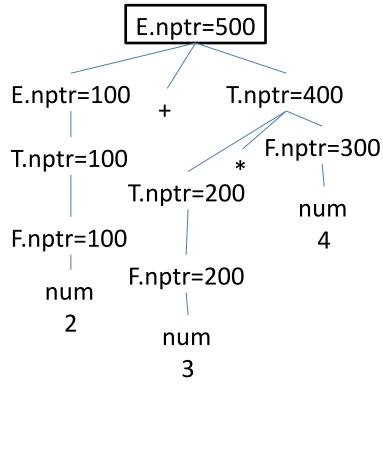






SDT to build a syntax tree

```
E \rightarrow E + T \{E.nptr=mknode(E.nptr, '+', T.nptr);\}
E \rightarrow T {E.nptr=T.nptr;}
T \rightarrow T * F \{T.nptr=mknode(T.nptr, *, F.nptr);\}
            {T.nptr=F.nptr}
F→ num {F.nptr=mknode(null, idname, null);}
 For input, 2 + 3 * 4
              500
                   100
                              400
                              400
                                  200
                                             300
                     200
100
                                           300
    null
              null
                        null
                               3
                                   null
                                             null
                                                       null
```



SDT to generate three address code

```
S \rightarrow id = E {gen(id.name = E.place}

E \rightarrow E_1 + T {E.place = newTemp(); gen(E.place = E_1.place + T.place);}

E \rightarrow T {E.place = T.place}

E \rightarrow T {T.place = newTemp(); gen(E.place = E_1.place * F.place);}

E \rightarrow T {T.place = newTemp(); gen(E.place = E_1.place * F.place);}

E \rightarrow T {T.place = id.name}
```

gen():- generates a statement in three address code

```
S \rightarrow id = E {gen(id.name = E.place}

E \rightarrow E_1 + T {E.place = newTemp(); gen(E.place = E_1.place + T.place);}

E \rightarrow T {E.place = T.place}

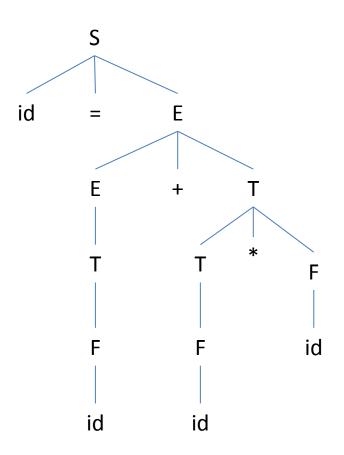
T \rightarrow T_1 * F {T.place = newTemp(); gen(E.place = E_1.place * F.place);}

E \rightarrow T {T.place = newTemp(); gen(E.place = E_1.place * F.place);}

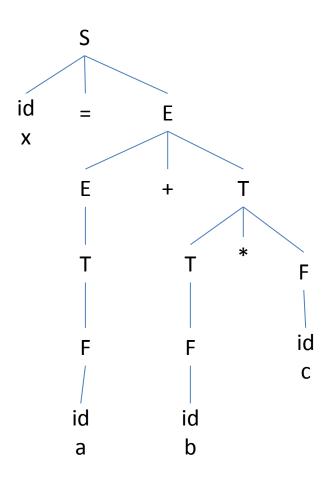
E \rightarrow T {T.place = id.name}
```

$$\begin{array}{c}
x = a + b * c \\
x = a + t1 \\
x = t2
\end{array}$$

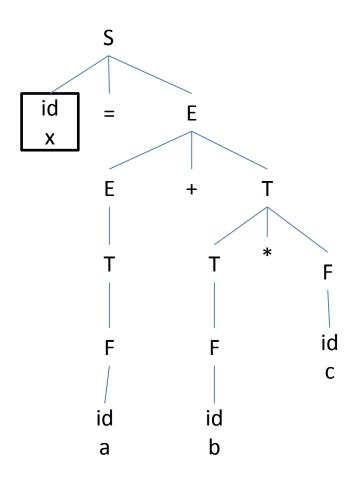
```
S \rightarrow id = E \quad \{gen(id.name = E.place\}\}
E \rightarrow E_1 + T \quad \{E.place = newTemp(); gen(E.place = E_1.place + T.place); \}
E \rightarrow T \quad \{E.place = T.place\}
T \rightarrow T_1 * F \quad \{T.place = newTemp(); gen(E.place = T_1.place * F.place); \}
T \rightarrow F \quad \{T.place = F.place\}
F \rightarrow id \quad \{F.place = id.name\}
```



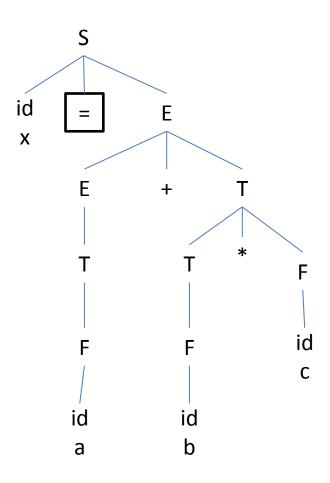
```
S \rightarrow id = E \{gen(id.name = E.place)\}
E \rightarrow E_1 + T  {E.place = newTemp();
              gen(E.place = E<sub>1</sub>.place + T.place);}
E \rightarrow T {E.place = T.place}
T \rightarrow T_1 * F \{T.place = newTemp();
              gen(E.place = T<sub>1</sub>.place * F.place);}
T \rightarrow F
             {T.place = F.place}
         {F.place = id.name}
F \rightarrow id
x = a + b * c
```



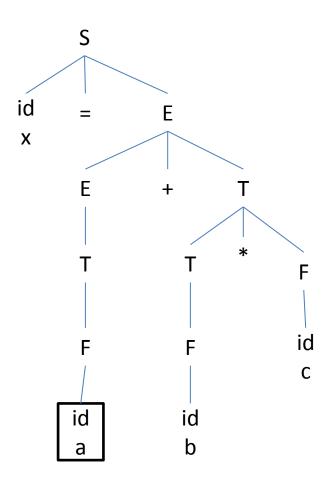
```
S \rightarrow id = E \{gen(id.name = E.place)\}
E \rightarrow E_1 + T  {E.place = newTemp();
               gen(E.place = E<sub>1</sub>.place + T.place);}
E \rightarrow T {E.place = T.place}
T \rightarrow T_1 * F \{T.place = newTemp();
               gen(E.place = T<sub>1</sub>.place * F.place);}
T \rightarrow F {T.place = F.place}
          {F.place = id.name}
F \rightarrow id
\mathbf{x} = \mathbf{a} + \mathbf{b} * \mathbf{c}
```



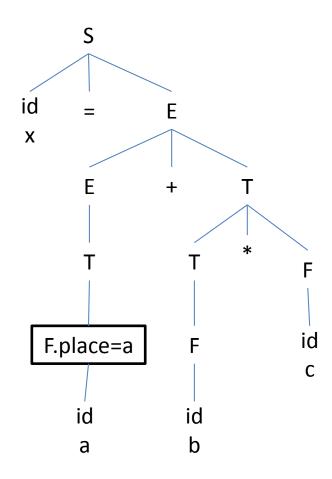
```
S \rightarrow id = E \{gen(id.name = E.place)\}
E \rightarrow E_1 + T  {E.place = newTemp();
              gen(E.place = E<sub>1</sub>.place + T.place);}
E \rightarrow T {E.place = T.place}
T \rightarrow T_1 * F \{T.place = newTemp();
              gen(E.place = T<sub>1</sub>.place * F.place);}
T \rightarrow F {T.place = F.place}
F \rightarrow id {F.place = id.name}
x = a + b * c
```



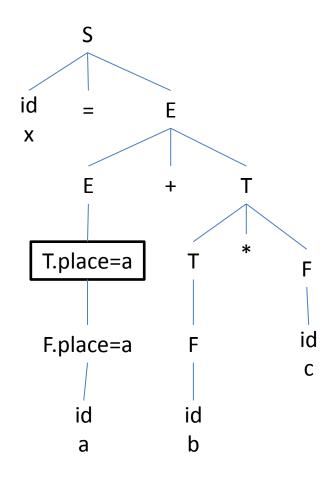
```
S \rightarrow id = E \quad \{gen(id.name = E.place\} \}
E \rightarrow E_1 + T \quad \{E.place = newTemp(); \\ gen(E.place = E_1.place + T.place); \}
E \rightarrow T \quad \{E.place = T.place\} \}
T \rightarrow T_1 * F \quad \{T.place = newTemp(); \\ gen(E.place = T_1.place * F.place); \}
T \rightarrow F \quad \{T.place = F.place\} \}
F \rightarrow id \quad \{F.place = id.name\}
```



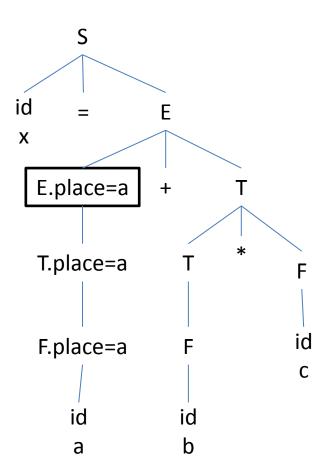
```
S \rightarrow id = E \quad \{gen(id.name = E.place\} \}
E \rightarrow E_1 + T \quad \{E.place = newTemp(); \\ gen(E.place = E_1.place + T.place); \}
E \rightarrow T \quad \{E.place = T.place\} \}
T \rightarrow T_1 * F \quad \{T.place = newTemp(); \\ gen(E.place = T_1.place * F.place); \}
T \rightarrow F \quad \{T.place = F.place\} \}
F \rightarrow id \quad \{F.place = id.name\} \}
```



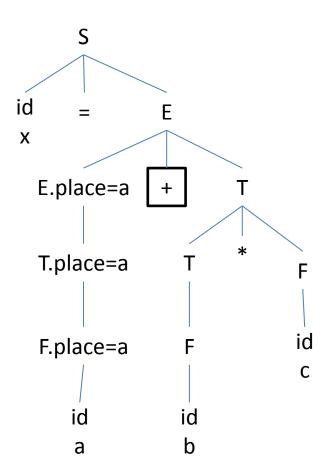
```
S \rightarrow id = E \{gen(id.name = E.place)\}
E \rightarrow E_1 + T  {E.place = newTemp();
              gen(E.place = E<sub>1</sub>.place + T.place);}
E \rightarrow T {E.place = T.place}
T \rightarrow T_1 * F \{T.place = newTemp();
              gen(E.place = T<sub>1</sub>.place * F.place);}
T \rightarrow F {T.place = F.place}
F \rightarrow id {F.place = id.name}
x = a + b * c
```



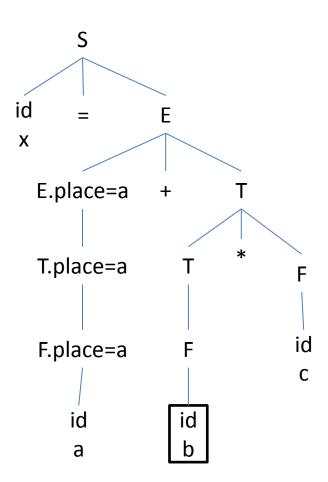
```
S \rightarrow id = E \{gen(id.name = E.place)\}
E \rightarrow E_1 + T \{E.place = newTemp();
              gen(E.place = E<sub>1</sub>.place + T.place);}
E \rightarrow T {E.place = T.place}
T \rightarrow T_1 * F \{T.place = newTemp();
              gen(E.place = T<sub>1</sub>.place * F.place);}
T \rightarrow F {T.place = F.place}
         {F.place = id.name}
F \rightarrow id
x = a + b * c
```



```
S \rightarrow id = E \{gen(id.name = E.place)\}
E \rightarrow E_1 + T  {E.place = newTemp();
              gen(E.place = E<sub>1</sub>.place + T.place);}
E \rightarrow T {E.place = T.place}
T \rightarrow T_1 * F \{T.place = newTemp();
              gen(E.place = T<sub>1</sub>.place * F.place);}
T \rightarrow F {T.place = F.place}
         {F.place = id.name}
F \rightarrow id
x = a + b * c
```



```
S \rightarrow id = E \{gen(id.name = E.place)\}
E \rightarrow E_1 + T \{E.place = newTemp();
              gen(E.place = E<sub>1</sub>.place + T.place);}
E \rightarrow T {E.place = T.place}
T \rightarrow T_1 * F \{T.place = newTemp();
              gen(E.place = T<sub>1</sub>.place * F.place);}
T \rightarrow F
             {T.place = F.place}
         {F.place = id.name}
F \rightarrow id
x = a + b * c
```



```
S → id = E {gen(id.name = E.place}

E → E<sub>1</sub> + T {E.place = newTemp();

gen(E.place = E<sub>1</sub>.place + T.place);}

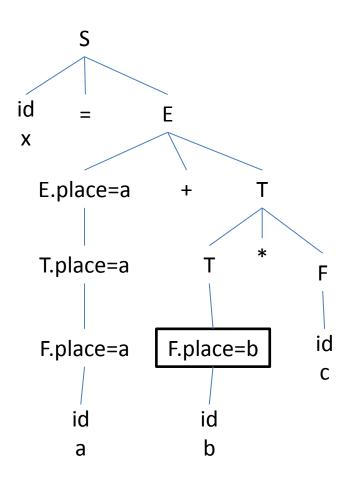
E → T {E.place = T.place}

T → T<sub>1</sub>* F {T.place = newTemp();

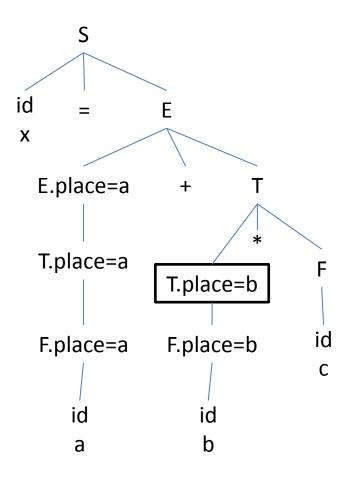
gen(E.place = T<sub>1</sub>.place * F.place);}

T → F {T.place = F.place}

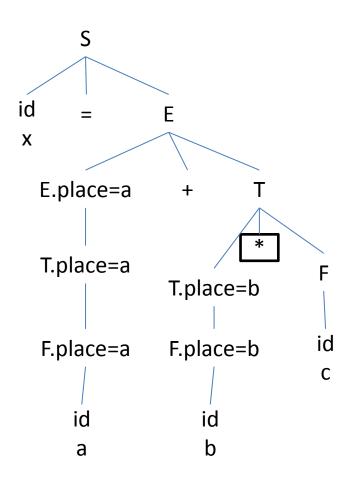
F → id {F.place = id.name}
```



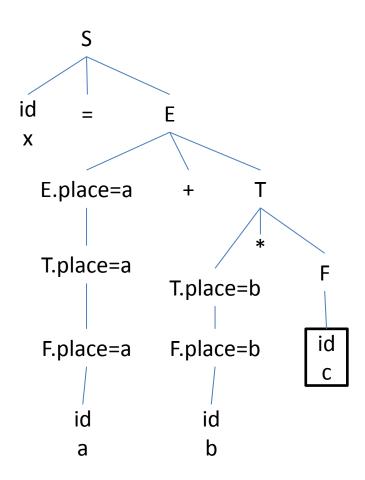
```
S \rightarrow id = E \{gen(id.name = E.place\}\}
E \rightarrow E_1 + T \{E.place = newTemp();
              gen(E.place = E<sub>1</sub>.place + T.place);}
E \rightarrow T {E.place = T.place}
T \rightarrow T_1 * F \{T.place = newTemp();
              gen(E.place = T<sub>1</sub>.place * F.place);}
T \rightarrow F {T.place = F.place}
         {F.place = id.name}
F \rightarrow id
x = a + b * c
```



```
S \rightarrow id = E \quad \{gen(id.name = E.place\} \}
E \rightarrow E_1 + T \quad \{E.place = newTemp(); \\ gen(E.place = E_1.place + T.place); \}
E \rightarrow T \quad \{E.place = T.place\} \}
T \rightarrow T_1 * F \quad \{T.place = newTemp(); \\ gen(E.place = T_1.place * F.place); \}
T \rightarrow F \quad \{T.place = F.place\} \}
F \rightarrow id \quad \{F.place = id.name\} \}
```



```
S \rightarrow id = E \quad \{gen(id.name = E.place\} \}
E \rightarrow E_1 + T \quad \{E.place = newTemp(); \\ gen(E.place = E_1.place + T.place); \}
E \rightarrow T \quad \{E.place = T.place\} \}
T \rightarrow T_1 * F \quad \{T.place = newTemp(); \\ gen(E.place = T_1.place * F.place); \}
T \rightarrow F \quad \{T.place = F.place\} \}
F \rightarrow id \quad \{F.place = id.name\} \}
```



```
S → id = E {gen(id.name = E.place}

E → E<sub>1</sub> + T {E.place = newTemp();

gen(E.place = E<sub>1</sub>.place + T.place);}

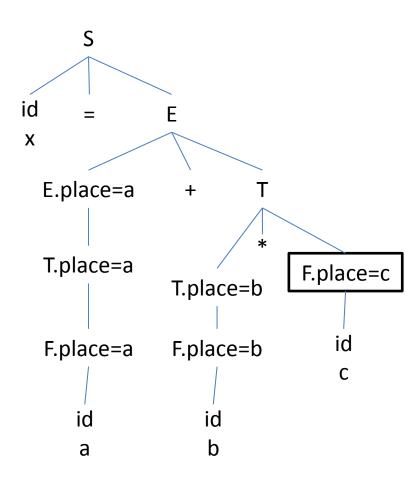
E → T {E.place = T.place}

T → T<sub>1</sub> * F {T.place = newTemp();

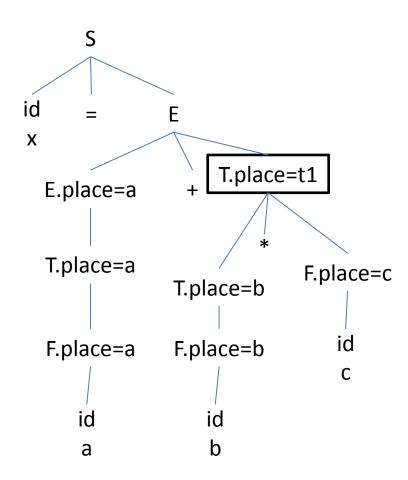
gen(E.place = T<sub>1</sub>.place * F.place);}

T → F {T.place = F.place}

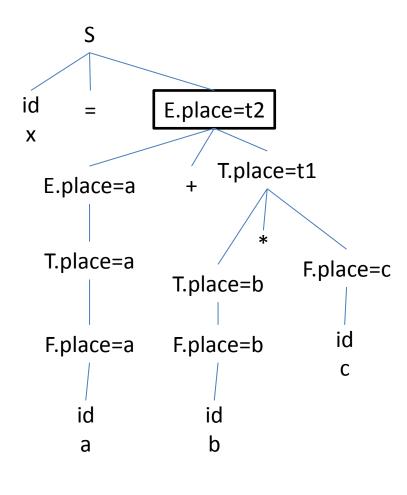
F → id {F.place = id.name}
```



```
S \rightarrow id = E \quad \{gen(id.name = E.place\} \}
E \rightarrow E_1 + T \quad \{E.place = newTemp(); \\ gen(E.place = E_1.place + T.place); \}
E \rightarrow T \quad \{E.place = T.place\} \}
T \rightarrow T_1 * F \quad \{T.place = newTemp(); \\ gen(E.place = T_1.place * F.place); \}
T \rightarrow F \quad \{T.place = F.place\} \}
F \rightarrow id \quad \{F.place = id.name\} \}
x = a + b * c
t1 = b * c
```



```
S \rightarrow id = E \quad \{gen(id.name = E.place\} \}
E \rightarrow E_1 + T \quad \{E.place = newTemp(); \\ gen(E.place = E_1.place + T.place); \}
E \rightarrow T \quad \{E.place = T.place\} \}
T \rightarrow T_1 * F \quad \{T.place = newTemp(); \\ gen(E.place = T_1.place * F.place); \}
T \rightarrow F \quad \{T.place = F.place\} \}
F \rightarrow id \quad \{F.place = id.name\} \}
x = a + b * c
```



```
S → id = E {gen(id.name = E.place}

E → E<sub>1</sub> + T {E.place = newTemp();

gen(E.place = E<sub>1</sub>.place + T.place);}

E → T {E.place = T.place}

T → T<sub>1</sub> * F {T.place = newTemp();

gen(E.place = T<sub>1</sub>.place * F.place);}

T → F {T.place = F.place}

F → id {F.place = id.name}
```

$$x = a + b * c$$

