Mansoura University
Faculty of Computers and Information Sciences
Department of Computer Science
Second Semester- 2020-2021

[CS422p] Compiler Construction

Grade: Fourth grade

By : Zeinab Awad

Faculty of Computers and Information
Mansoura University

Compiler construction

Syntax-Directed Translation

BY :ZEINAB AWAD

Computer Science Department

Zeinab_awad@mans.edu.eg

semantic Analysis

Syntax –directed translation

SYNTAX DIRECTED TRANSLATION

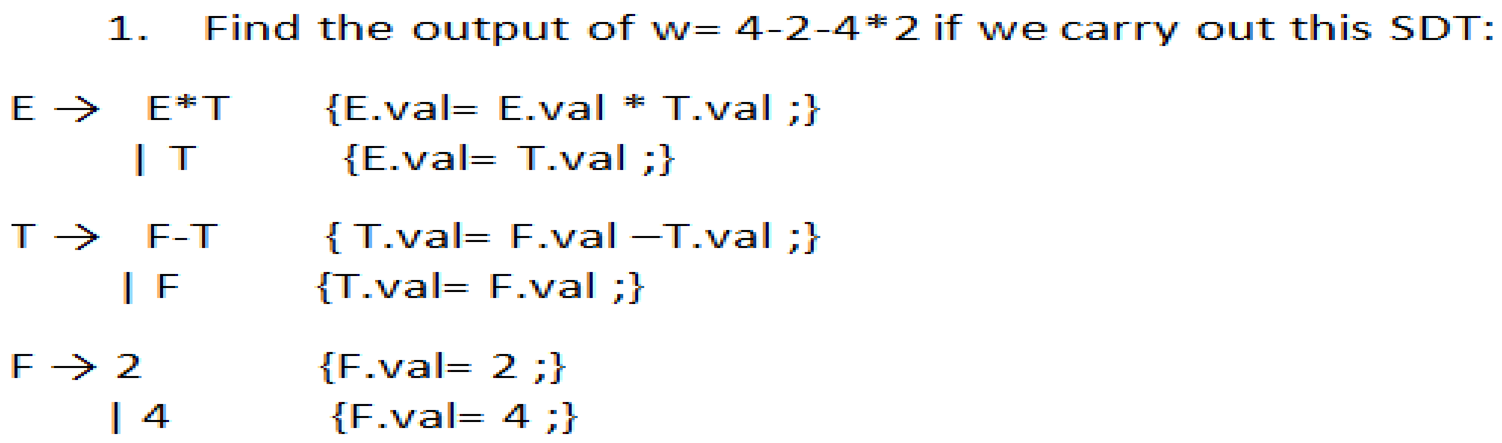
- SYNTAX DIRECTED TRANSLATION (SDT) :

I. What is SDT? (Grammer+semantic rules)

II. generating an output by applying the SDT to specific input.

III. Evaluating an input the with or without constructing a parse tree using STD semantic rules .

SDT – question 1.



1. Find the output of w= 4-2-4*2 if we carry out this SDT:

```
E →   E*T       {E.val= E.val * T.val ;}
     | T         {E.val= T.val ;}

T →   F-T       { T.val= F.val –T.val ;}
     | F        {T.val= F.val ;}

F → 2           {F.val= 2 ;}
    | 4          {F.val= 4 ;}
```

Solution:

By examine the grammar , The – operator has higher precedence than the * precedence .E in the first rule is left associative(left recursion) thus the * operator is left associative. and T is right associative (right recursion) thus – is right associative.

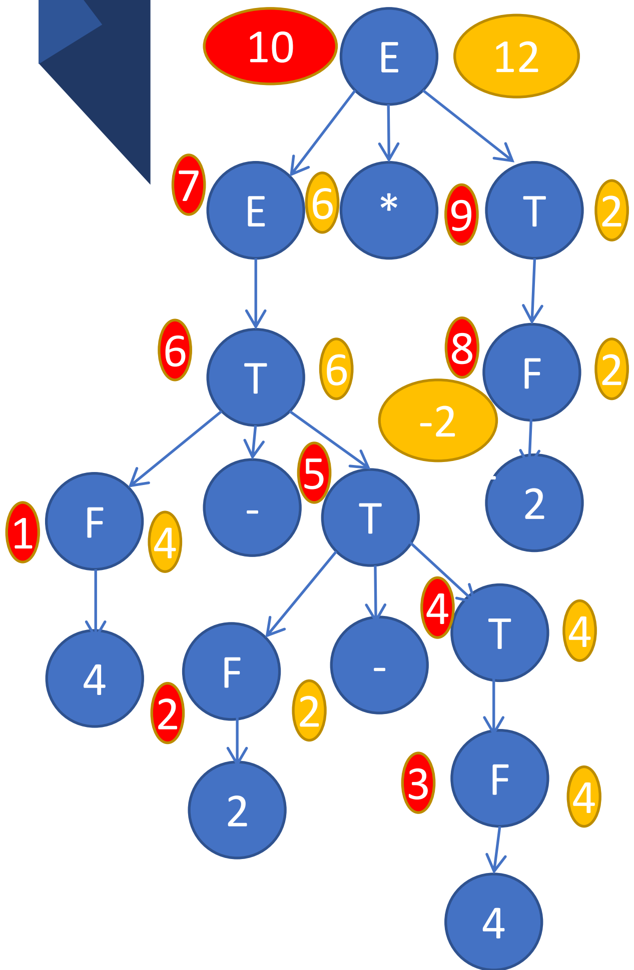
SDT – question 1.

The input 4-2-4*2 will be evaluated as follow:

$$\underbrace{(\underbrace{(4-\underbrace{(2-4)}_{-2})}_{6}*2)}_{12}$$

The output will be : 12

SDT – question 1.



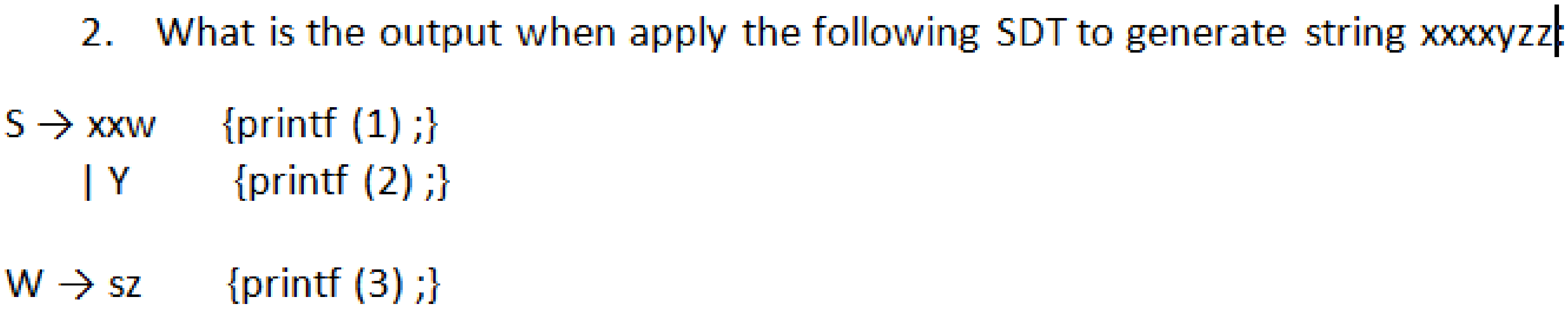
By constructing the parse tree for the input 4-2-4*2 it could be evaluated by traverse the tree using the LR bottom up parsing and execute the action associated with each reduction. Indicated by the orange circle.

The output will be : 12

NOTE: 🟡 indicates the result after executing the action of each reduction.

🔴 indicates the reduction number.

SDT(1) – question 2.



2. What is the output when apply the following SDT to generate string xxxxyzz

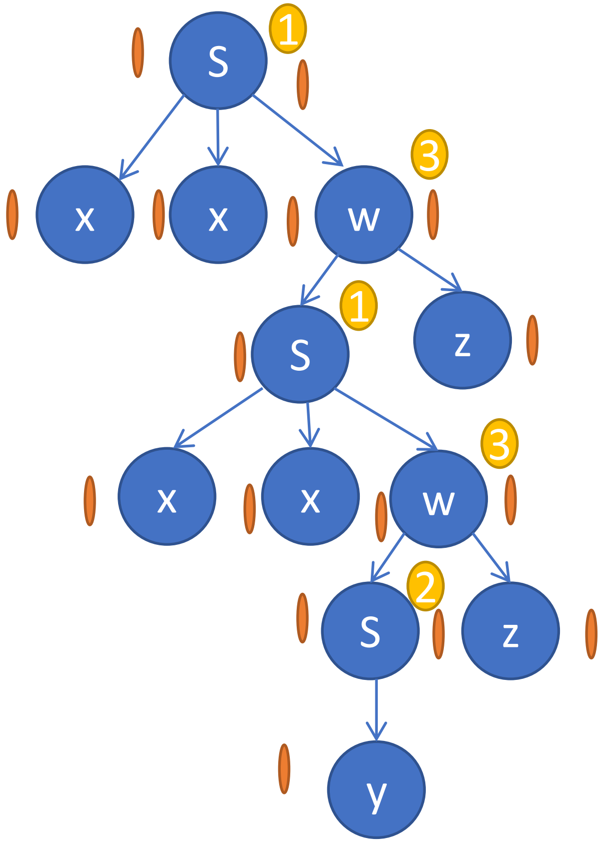
```
S → xxw     {printf (1) ;}
    | Y      {printf (2) ;}

W → sz      {printf (3) ;}
```

Solution:

If we use the bottom up parser (LR) :

Step-1: generate the string xxxxyzz with a parse tree

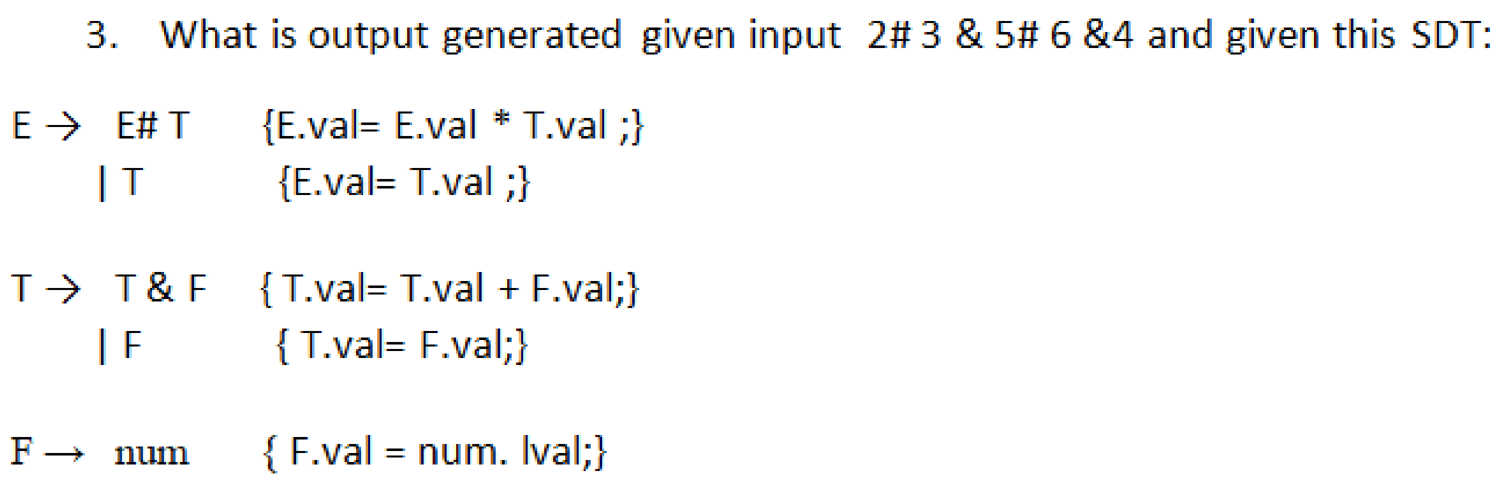
SDT – question 2.



Step-2: traverse the tree using the LR bottom up parsing and execute the action associated with each reduction.

The generated output is 23131 as indicated by the tree.

SDT– question 3.



3. What is output generated given input 2# 3 & 5# 6 &4 and given this SDT:

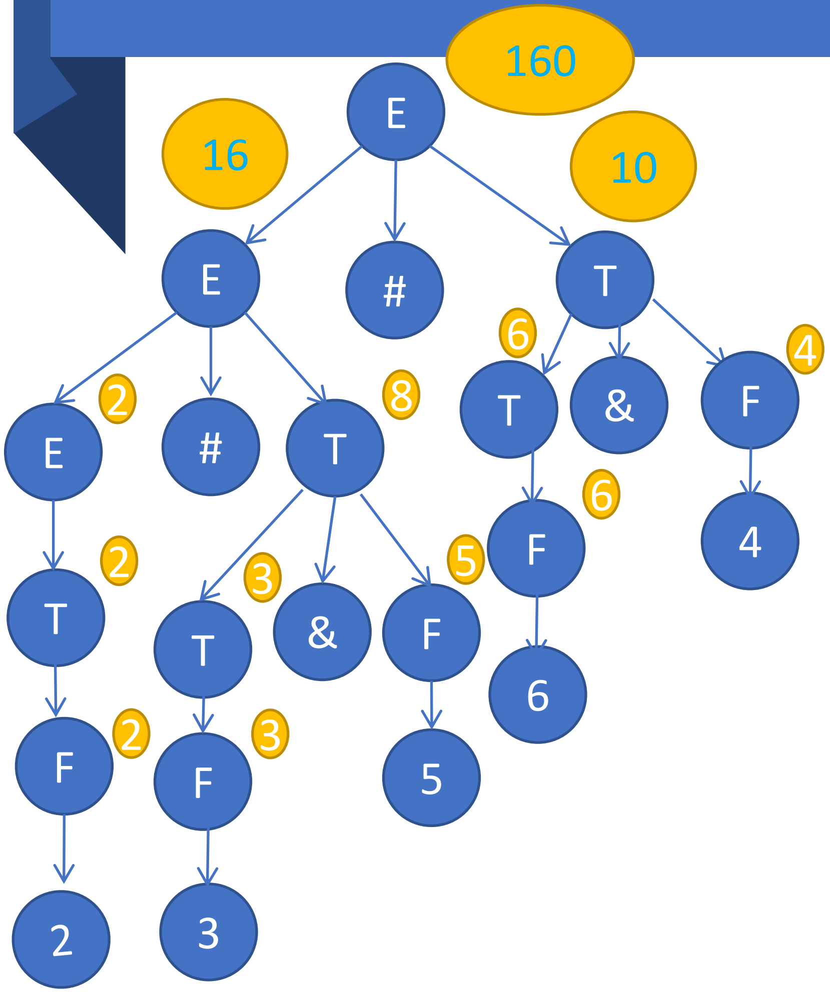
```
E →  E# T     {E.val= E.val * T.val ;}
     | T       {E.val= T.val ;}

T →  T & F    { T.val= T.val + F.val;}
     | F       { T.val= F.val;}

F →  num      { F.val = num. lval;}
```

Solution:

If we use the bottom up parser (LR) :

Step-1: generate the string 2#3&5#6&4 with a parse tree

SDT– question 3.



Step-2: traverse the tree using the LR bottom up parsing and execute the action associated with each reduction.

After evaluating the input 2#3&5#6&4 it will be reduced to(2*(3+5)*(6+4))=((2*8)*10)=160.

Note:+ has higher precedence than * .and both *,+ are left associative .
# replaced by *,& replaced by + .

SYNTAX DIRECTED TRANSLATION

THANK YOU