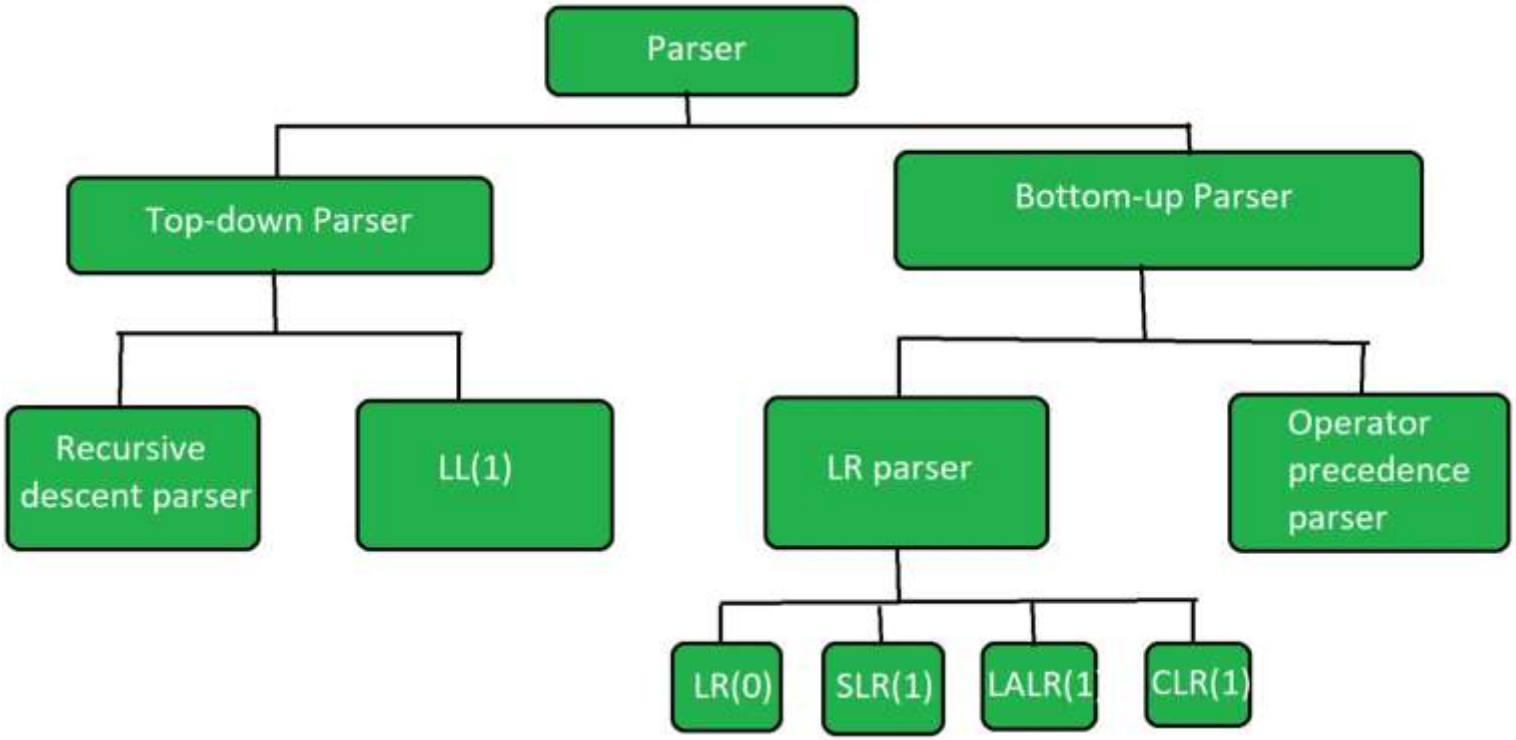
Types of Parser

- Parser - token →parse tree (CFG)



Top Down Parsing

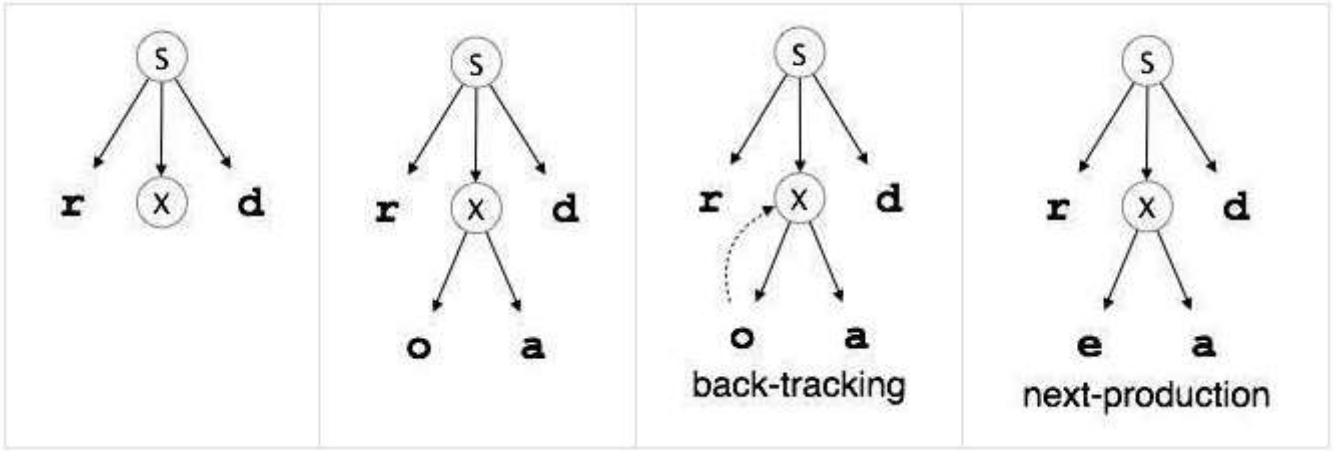
- Non Terminal (Start Symbol) → terminals
- Left most derivation
- Top Down parsing
  - Recursive Descent parser *(Brute force parse/backtracking)*
    - (it processes the input string more than once)
  - Non - Recursive Descent parser *(without backtracking)*
    - Predictive Parsing LL(1)

Top Down Parsing

Recursive Descent parser *(Brute force parse/backtracking)*

- ***Example :***

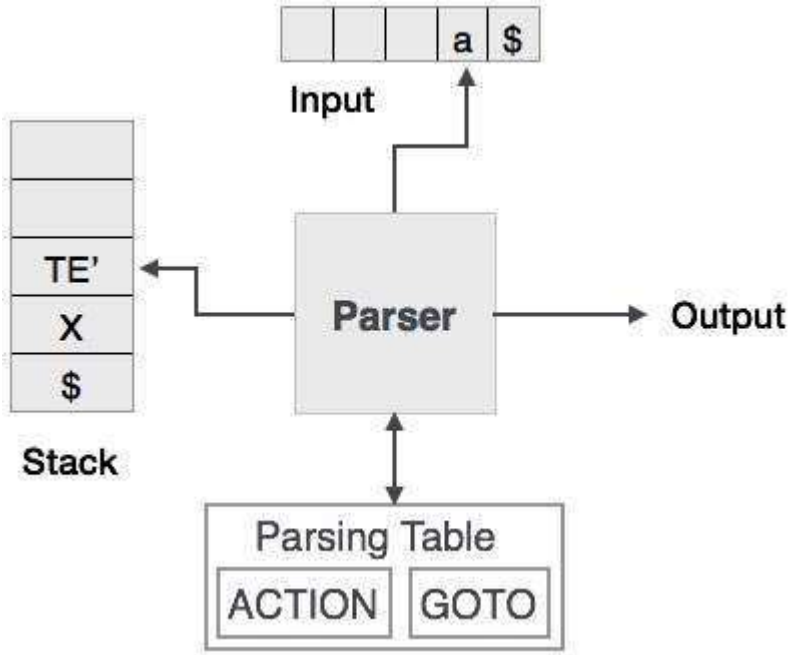
```
S → rXd | rZd
X → oa | ea
Z → ai
```



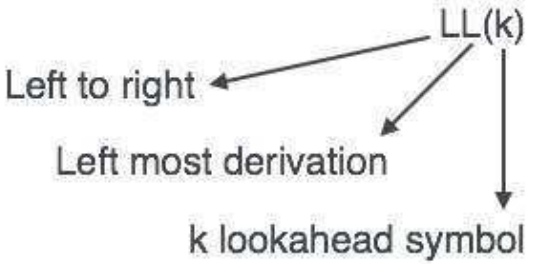
Top Down Parsing

Predictive Parser *(Without Backtracking)*

- Production – replace the string
- Look ahead pointer –next input symbol
- LL(K) Grammar
- $ → input buffer and stack



LL(1) / Predictive parser



LL(1) Parser – Example-1

$S \rightarrow aBa$

$B \rightarrow bB \mid \varepsilon$

	a	b	$
S	$S \rightarrow aBa$		
B	$B \rightarrow \varepsilon$	$B \rightarrow bB$	

LL(1) Parsing Table

stack	**input**	**output**
$S	abba$	$S \rightarrow aBa$
$aBa	abba$	
$aB	bba$	$B \rightarrow bB$
$aBb	bba$	
$aB	ba$	$B \rightarrow bB$
$aBb	ba$	
$aB	a$	$B \rightarrow \varepsilon$
$a	a$	
$	$	accept, successful completion

Bottom Up Parsing

- Leaf to Root Node (String → Start Symbol)
- Right most derivation
- Bottom up parsing
  - LR parser
    - LR(0)
    - SLR(1)
    - CLR
    - LALR
  - Operator Precedence parser

Bottom Up Parsing - Example

S → aABe

A → Abc/b

B → d

Input : abbcde