LR(K) GRAMMARS

- A grammar G is said to be **LR(k)** if, given any right-most derivation of a string in G, and any of its derivation steps uAv ⇒ uxv where v is a string of terminals, the production A → x can be inferred by scanning ux and at most the first k symbols of v.
- LR(k) grammars are **unambiguous** and **can be left-recursive**
- Every LL(k) grammar is LR(k)
- LR(k) grammars can be parsed using shift-reduce parsers: table driven bottom-up parsers which can be automatically generated from LR(k) grammars.

SHIFT-REDUCE PARSING

- Bottom-up or shift-reduce parsers consist of
  - An input stream of tokens which will be read from left to right
  - A symbol stack
  - A state stack
  - A parsing table used to decide what kind of operation should be performed depending on what is on the stacks
- Shift-reduce parsers loop until the following two conditions are satisfied:
  - There are no more tokens to read
  - There is only one element on each stack: S on the symbol stack.
- At each step of the loop they do one of the following:
  - Shift: S<state>
    - read a token and push it on the symbol stack
    - push state on state stack
  - Reduce: R<production>
    - pop the N top elements off each stack (where N is the number of terminals and non-terminals on the RHS of production)
    - insert production LHS in input stream
    - build parse tree

EXAMPLE

Grammar

(1) S → **real** IDLIST
(2) IDLIST → IDLIST**,** ID
(3) IDLIST → ID
(4) ID → **a** | **b** | **c** | **d**

State	S	IDLIST	ID	real	,	a b c d	$
1	HALT			S2			
2		S5	S4			S3	
3					R4		R4
4					R3		R3
5					S6		R1
6			S7			S3	
7					R2		R2

<table>
<tr><th>State Stack</th><th>Input</th><th>Symbol Stack</th><th>Operation</th></tr>
<tr><td>1 |</td><td>real a, b, c $</td><td></td><td>S2</td></tr>
<tr><td>1 | 2 |</td><td>a, b, c $</td><td>real |</td><td>S3</td></tr>
<tr><td>1 | 2 | 3 |</td><td>, b, c $</td><td>real | a |</td><td>R4</td></tr>
<tr><td>1 | 2 |</td><td>ID, b, c $</td><td>real |</td><td>S4</td></tr>
<tr><td>1 | 2 | 4 |</td><td>, b, c $</td><td>real | ID |</td><td>R3</td></tr>
<tr><td>1 | 2 |</td><td>IDLIST, b, c $</td><td>real |</td><td>S5</td></tr>
<tr><td>1 | 2 | 5 |</td><td>, b, c $</td><td>real | IDLIST |</td><td>S6</td></tr>
<tr><td>1 | 2 | 5 | 6 |</td><td>b, c $</td><td>real | IDLIST | , |</td><td>S3</td></tr>
<tr><td>1 | 2 | 5 | 6 | 3 |</td><td>, c $</td><td>real | IDLIST | , | b |</td><td>R4</td></tr>
<tr><td>1 | 2 | 5 | 6 |</td><td>ID , c $</td><td>real | IDLIST | , |</td><td>S7</td></tr>
<tr><td>1 | 2 | 5 | 6 | 7 |</td><td>, c $</td><td>real | IDLIST | , | ID |</td><td>R2</td></tr>
<tr><td>1 | 2 |</td><td>IDLIST, c $</td><td>real |</td><td>S5</td></tr>
<tr><td>1 | 2 | 5 |</td><td>, c $</td><td>real | IDLIST |</td><td>S6</td></tr>
<tr><td>1 | 2 | 5 | 6 |</td><td>c $</td><td>real | IDLIST | , |</td><td>S3</td></tr>
<tr><td>1 | 2 | 5 | 6 | 3 |</td><td>$</td><td>real | IDLIST | , | c |</td><td>R4</td></tr>
<tr><td>1 | 2 | 5 | 6 |</td><td>ID $</td><td>real | IDLIST | ,</td><td>S7</td></tr>
<tr><td>1 | 2 | 5 | 6 | 7 |</td><td>$</td><td>real | IDLIST | , | ID |</td><td>R2</td></tr>
<tr><td>1 | 2 |</td><td>IDLIST $</td><td>real |</td><td>S5</td></tr>
<tr><td>1 | 2 | 5 |</td><td>$</td><td>real | IDLIST |</td><td>R1</td></tr>
<tr><td>1 |</td><td>S $</td><td></td><td>HALT</td></tr>
</table>