id + id * id $

	id	num	(	)	+	*	$
E	E→TE'	E→TE'	E→TE'				
E'				E'→ε	E'→+TE'		E'→ε
T	T→FT'	T→FT'	T→FT'				
T'				T'→ε	T'→ε	T'→*FT'	T'→ε
F	F→id	F→num	F→(E)				

STACK	INPUT	MATCHED	ACTION
$ E	id + id * id $		E → T E'
$ E' T	id + id * id $		T → F T'
$ E' T' F	id + id * id $		F → id
$ E' T' id	id + id * id $	id	match
$ E' T'	+ id * id $		T' → ε
$ E'	+ id * id $		E' → + T E'
$ E' T +	+ id * id $	+	match
$ E' T	id * id $		T → F T'
$ E' T' F	id * id $		F → id
$ E' T' id	id * id $	id	match
$ E' T'	* id $		T' → * F T'
$ E' T' F *	* id $	*	match
$ E' T' F	id $		F → id
$ E' T' id	id $	id	match
$ E' T'	$		T' → ε
$ E'	$		E' → ε
$	$		ACCEPT

PARSE STRING

LEFTMOST DERIVATION FOR

$S \to aSb \mid Ad \mid Bc$

$A \to Aa \mid c$

$B \to ddA \mid dC$

$C \to aC$

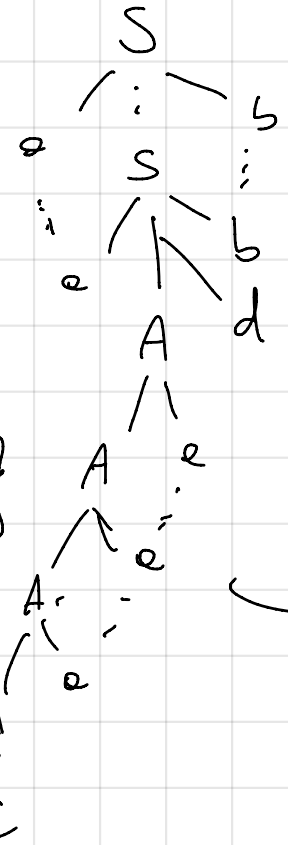
$$L = \{ a^n c a^m b^n \mid n \geq 0, m \geq 0 \}$$

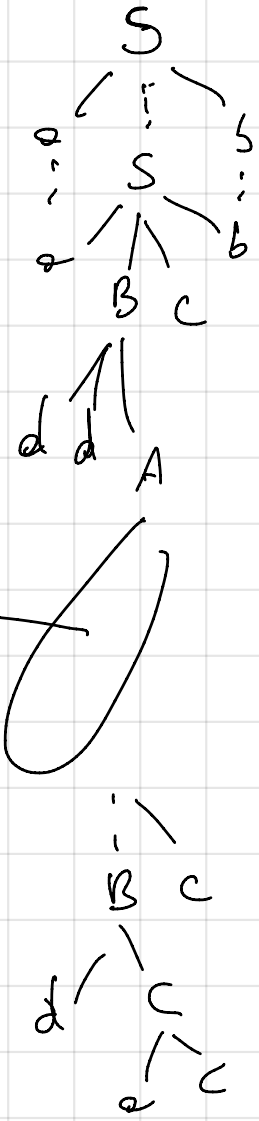
$$\cup$$

$$\{ a^n dd c a^m c b^n \mid n \geq 0, m \geq 0 \}$$

$$\cup$$

$$\{ a^n d a c c b^n \mid n \geq 0 \}$$





The grammar is left-recursive ($A \to Aa$), so it cannot be LL(K) for any K.

$S \to aSb \mid cA \mid dB$

$A \to aA \mid \varepsilon$

$B \to dC \mid aD$

$C \to cAc$

$D \to cc$

FIRST(S) = { a, c, d }

FIRST(A) = { a, ε }

FIRST(B) = { d, a }

FIRST(C) = { c }

FIRST(D) = { c }

FOLLOW(S) = { $, b }

FOLLOW(A) = { $, b, c }

FOLLOW(B) = { $, b }

FOLLOW(C) = { $, b }

FOLLOW(D) = {

	a	b	c	d	$
S	$S \to aSb$		$S \to cA$	$S \to dB$	
A	$A \to aA$	$A \to \varepsilon$	$A \to \varepsilon$		$A \to \varepsilon$
B	$B \to aD$			$B \to dC$	
C			$C \to cAc$		
D			$D \to cc$		

The given grammar is LL(1) and this $\mathcal{G}$ is a table for a predictive Top-down parser.

$\oplus$, $\otimes$ id ( )

$\oplus$ has precedence over $\otimes$

$\oplus$ is right-associative

$\otimes$ is left-associative

$\otimes$	left	E
$\oplus$	right	T
operand	id	F

$E \to E \otimes T \mid T$

$T \to F \oplus T \mid F$

$F \to id \mid (E)$

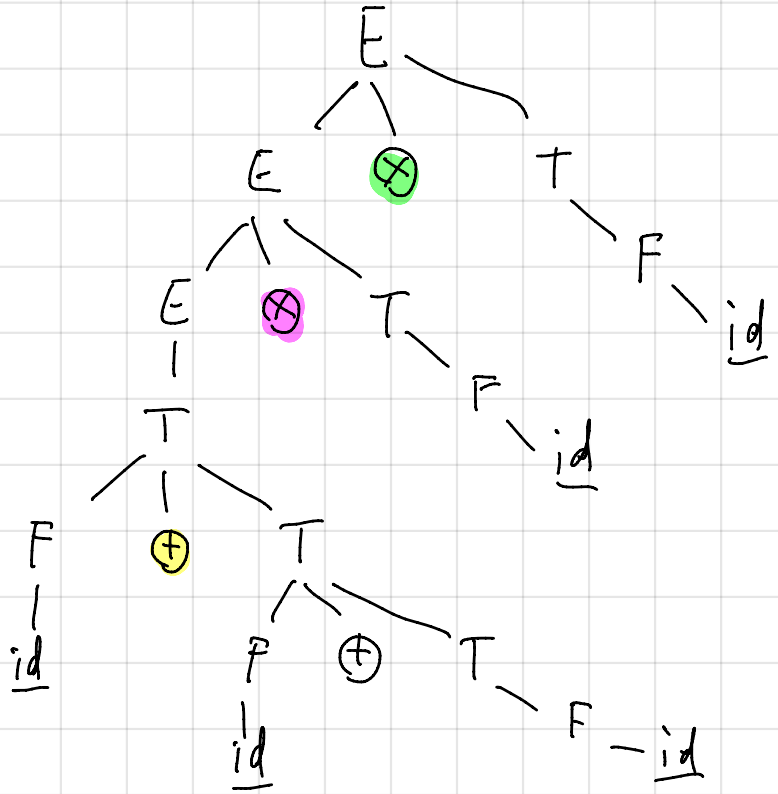
id $\oplus$ id $\oplus$ id $\otimes$ id $\otimes$ id

$E \to E \otimes T \mid T$

$T \to F \oplus T \mid F$

$F \to id \mid (E)$

id ⊕ id ⊕ id ⊗ id ⊗ id



order of operations according to the rules:

$$\Big[\big[id \oplus \big[id \oplus id\big]\big] \otimes id\Big] \otimes id$$

$L = \{a^n b c \mid n > 0\} \cup \{b^n c b \mid n \geq 0\} \cup \{c a^n \mid n > 0\}$

$S \to aA \mid bB \mid cC$

$A \to aA \mid bc$

$B \to bB \mid cb$

$C \to b \mid aD$

$D \to aD \mid \varepsilon$

FIRST (S) = { a, b, c }
FIRST (A) = { a, b }
FIRST (B) = { b, c }
FIRST (C) = { b, a }
FIRST (D) = { a, ε }

FOLLOW(S) : { $ }
FOLLOW(A) : { $ }
FOLLOW(B) = { $ }
FOLLOW(C) = { $ }
FOLLOW(D) = { $ }

	a	b	c	$
S	S → aA	S → bB	S → cC	
A	A → aA	A → bc		
B		B → bB	B → cb	
C	C → aD	C → b		
D	D → aD			D → ε

The given grammar is LL(1), so the Language is LL(1).

STACK	INPUT	MATCH	ACTION
$ S	c b $		S → cC
$ C c	c b $	c	match
$ C	b $		C → b
$ b	b $	b	match
$	$		ACCEPT

STACK	INPUT	MATCH	ACTION
$ S	cad $		S → c C
$ C c	cad $	c	match
$ C	ad $		C → a D
$ D a	ad $	a	match
$ D	d $		D → a D
$ D a	d $	a	match
$ D	$		D → ε
$	$		ACCEPT

Bottom-up Parsing

$E \to E + T \mid T$

$T \to T * F \mid F$

$F \to id \mid (E)$

id * id

$E \Rightarrow_{rm} T \Rightarrow_{rm} T * F \Rightarrow_{rm} T * id$

$\Rightarrow_{rm} F * id \Rightarrow_{rm} id * id$

handle

id * id can be reduced to F

handle

F * id can be reduced to T

handle

T * id can be reduced to F

T * F can be reduced to T

↓

T can be reduced to E

↓

E

$S \to 0S1 \mid 01$

handle

000111

The parser will have to "shift" the first two 00, then shift 01 and reduce 01 to S

handle

00S11