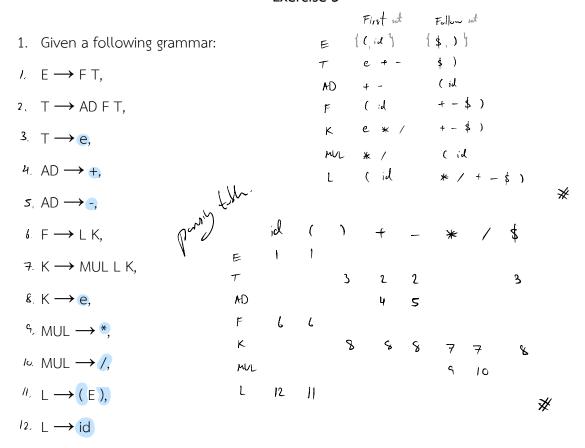
Exercise 3

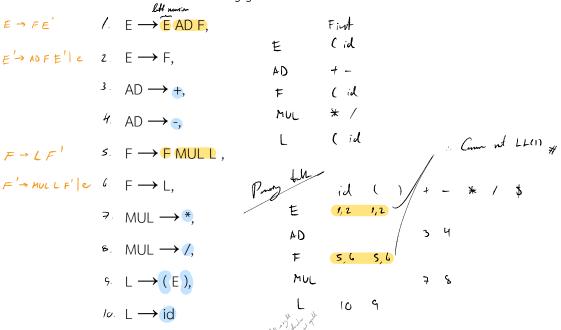


- a. Find the first and follow sets of the grammar.
- b. The parsing table of the grammar
- 2. From the parsing table in (1), use stack to simulate leftmost derivation as the LL(1) parsing for stream of tokens id + id * (id + id).

No.	Stack	Tokens	Action
1.	\$	id + id * (id + id)\$	
2.	\$E	"	€→ FT
	\$TF		F→ LK.

\$ + 11 +	* (iJ+id)\$	K) *
\$ T K L	(id+id) \$	L→ (E)
\$ TK)E((ideil)\$	<i>A</i> (
\$ T N) E	id+id) \$	E → FT
9 T K) T F	″	F → LK
\$ TK) TKL	\$ (Ki = Ki	L -> id
\$ TK) TK il	il * id > \$	as id
\$ TK) TK	+ id) \$	K→C
\$ 7 K) T	\$ (hi +	T -> AD F T
\$ TK) TFAD	+ 2)\$	AD -> +
\$ T K) T F +	4 2) \$	ns +
\$ T K) T F	\$ (Li	F → LK
\$ TK) TKL	4	L- id
d TK) TKid	;d)\$	d rid
\$ T R) T R) \$	KAC
\$ 7 K) T	h	ナウヒ
\$ 7 N >) \$	A)
\$ 7 n	\$	n oc
4 +		1→€
\$	\$	Augt *

3. Given a following grammar:



- a. Is the grammar LL(1)? Justify your answer. No, became it has left number and
- b. If it's not LL(1), how to change the grammar to LL(1)?
- 4. The following is a grammar for regular expressions over symbols a and b only, using + in place of | for union, to avoid conflict with the use of vertical bar as a metasymbol in grammars:

- a. Left factor this grammar.
- b. Does left factoring make the grammar suitable for top-down parsing?
- c. In addition to left factoring, eliminate left recursion from the original grammar.
- d. Is the resulting grammar suitable for top-down parsing? Justify your answer.