1. (10) Given the grammar below, identify which sentences are in the language (which are valid sentence).
   1. baab

<S> -> <A>a<B>b

->ba<B>b

->baab

Valid Sentence

* 1. bbbab

<S> -> <A>a<B>b

No way to change first <A> to get b,

Invalid sentence

* 1. bbaaaaaa

<S> -> <A>a<B>b

No way to end sentence with a b,

Invalid sentence

* 1. bbaab

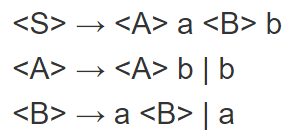
<S> -> <A>a<B>b

-> <A>ba<B>b

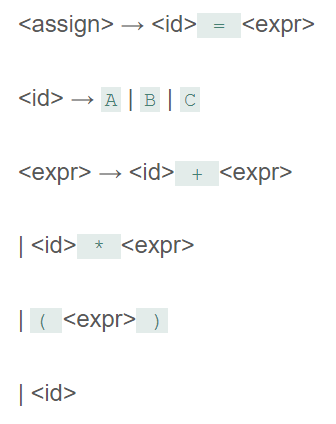
-> bba<B>b

-> bbaab

Valid Sentence



1. (10) Identify all of the tokens (categories of lexemes) in the grammar below, and which lexemes they categorize. Put them in a table.



|  |  |
| --- | --- |
| Token | Lexeme |
| EQ\_OPERATOR | “=” |
| VAR\_ID | “A”, “B”, “C” |
| MULTI\_OPERATOR | “\*” |
| ADD\_OPERATOR | “+” |
| LPAREN | “(“ |
| RPAREN | “)” |

1. (10) Given the grammar from question 2, show a left-most derivation and draw the parse tree for the following statement.
   1. B = B + (C + (A \* A) )

<assign>

→ <id> = <expr>

→ B = <expr>

→ B = <id> + <expr>

→ B = B + <expr>

→ B = B + (<expr>)

→ B = B + (<id> + <expr>)

→ B = B + (C + <expr>)

→ B = B + (C + (<expr>))

→ B = B + (C + (<id> \* <expr>))

→ B = B + (C + (A \* <expr>))

→ B = B + (C + (A \* <id>))

→ B = B + (C + (A \* A))

Diagram

Description automatically generated

1. (10) Remove all of the recursion from the following grammar:

S -> Aa | Bb

A -> Aa | AbC | C

B -> S | bb

C -> c

Answer:

S → Aa | Bb   
A → CA’   
A’ → bCA’ | aA’ |   
B → AaB’ | bbB’   
B’ → bB’ |   
C → c

1. (10) Use left factoring to resolve the pairwise disjointness problems in the following grammar:

A -> aBc | ac | a

B -> b | aB

Answer:

A → aA’   
A’ → A’’ |   
A’’ → Bc | c   
B → b | aB

1. (20 pts) Create an LR(0) parse table for the following grammar. Show all steps (creating closures, the DFA, the transition table, and finally the parse table):

E -> E + T | E \* T | T

T -> ( E ) | id

r0: S’ → E$   
r1: E → E + T   
r2: E → E \* T   
r3: E → T   
r4: T → (E)   
r5: T → id

Diagram

Description automatically generated

A picture containing text, white

Description automatically generated

Table, calendar

Description automatically generated

1. (20 pts) Show a complete bottom-up parse, including the parse stack contents, input string, and action for the string below using the parse table you created in step 6. Think about how I went through this in class.

**A picture containing table

Description automatically generated**

**A picture containing graphical user interface

Description automatically generated**

(id + id) \* id

1. (10 pts) Show a rightmost derivation for the string above, and show how the bottom-up parse you completed in step 7 correctly finds all of the handles for the input string above.

Table

Description automatically generated

The bottom-up parse from above gives an output that is the reverse of the order of rules that the rightmost derivation uses. This confirms that it finds the correct handles in the input string.