1. State which of the following production rules are *left-regular*, *right-regular*, *left-recursive*, *right-recursive*, *context-free* (or more than one, or none, of these):

   1. Thsi⟶This (a rewrite auto-applied by Microsoft Word).

	Yes	*No*
Left - Regular	✘	✓
Right - Regular	✘	✓
Left - Recursive	✘	✓
Right – Recursive	✘	✓
Context – Free	✘	✓

   2. Sentence⟶Subject Verb Object. (Here Sentence, Subject, Verb, and Object are non-terminal symbols.)

	Yes	*No*
Left - Regular	✘	✓
Right - Regular	✘	✓
Left - Recursive	✓	✘
Right – Recursive	✓	✘
Context – Free	✓	✘

   3. X⟶Xa.

	Yes	*No*
Left - Regular	✓	✘
Right - Regular	✘	✓
Left - Recursive	✓	✘
Right – Recursive	✘	✓
Context – Free	✓	✘

   4. X⟶XaX.

	Yes	*No*
Left - Regular	✘	✓
Right - Regular	✘	✓
Left - Recursive	✓	✘
Right – Recursive	✓	✘
Context – Free	✓	✘

2. What is the object language generated by $X \longrightarrow Xa$ (see lecture 2)? Explain your answer.

The Object Language would be the finite String of a's.

However there would be no output as the terminal is on the right hand side, thus the interpreter is constantly trying to end the non-terminal "X" on the left hand side of the production.

i.e Xa -> Xaa -> Xaaa -> Xaaaa ->…………..

If you ran this production it would fill up the memory of the computer and cause a memory leak.

3. Construct context-free grammars that generate the following languages:
   1. $(ab|ba)^*$.

      G=({S},{a,b, ε },S, with following production rules:

      S::= ab | ba | abS | baS | ε

      }

   2. $\{(ab)^n a^n \mid n \geq 1\}$.

      G=({S},{a,b },S, with following production rules:

      S::= aba | abSa

      }

   3. $\{w \in \{a,b\}^* \mid \text{ w is a palindrome }\}$.

      G=({P},{a,b, ε },P, with following production rules:

      P ::= a | b | aPa | bPb | ε

      }

4. $\{w \in \{a,b\}^* \mid w$ contains exactly two bs and any number of as $\}$.

G=({S, A},{a,b, ε },S, with following production rules:

S::= aS | bbA | Abb | AbbA
A::= aA | ε

}

5. $\{a^n b^m \mid 0 \leq n \leq m \leq 2n\}$.

G=({S},{a,b, ε },S, with following production rules:

S::= ε. | aSb | aSbb

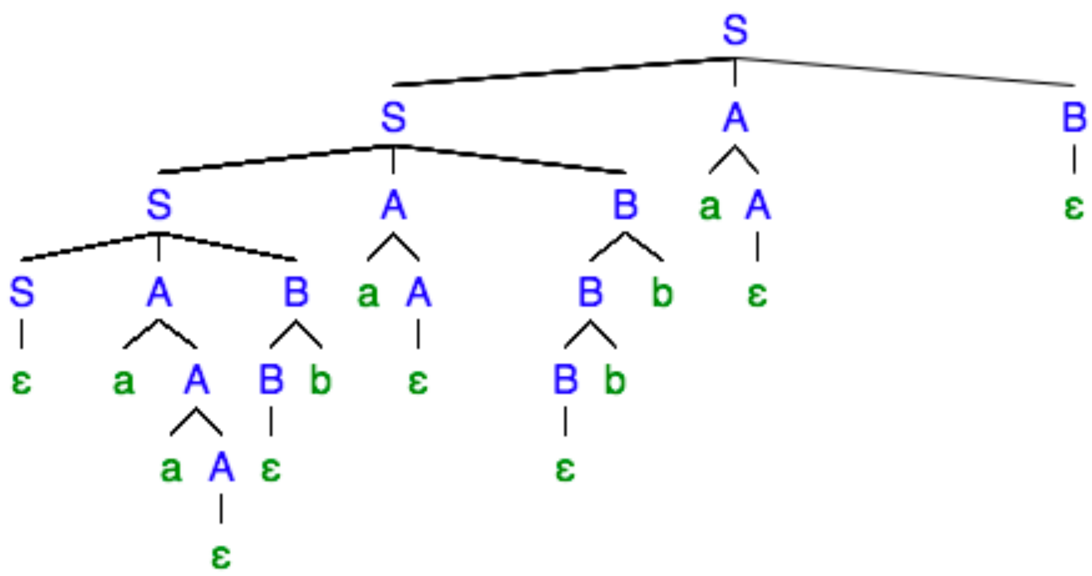
}

4. Consider the grammar $G=(\{S,A,B\},\{a,b\},P,S)$ with productions
   $S \longrightarrow SAB \mid \varepsilon$
   $A \longrightarrow aA \mid \varepsilon$
   $B \longrightarrow Bb \mid \varepsilon$

   1. Give a leftmost derivation for aababba.

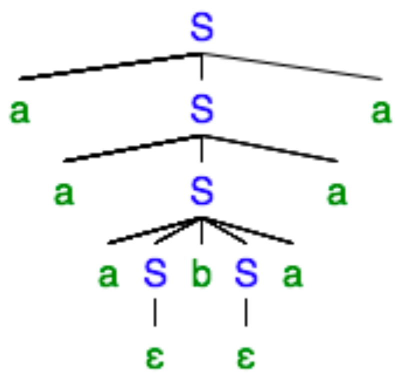
      S -> SAB -> SABAB -> SABABAB -> ABABAB -> aABABAB -> aaABABAB ->aaBABAB -> aaBbABAB -> aabABAB -> aabaABAB -> aabaBAB -> aabaBbAB -> aababbAB -> aababbaAB -> aababbaB -> aababba

   2. Draw the derivation tree corresponding to your derivation.

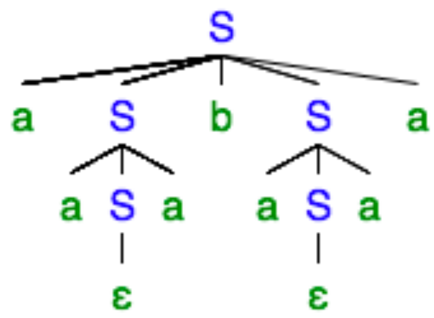


5. State with proof whether the following grammars are ambiguous or unambiguous:
   1. G=({S},{a,b},P,S) with productions
      S⟶ aSa | aSbSa | ε.

      The grammar here is ambiguous because there is multiple ways of creating the same sentence using different routes of production rules.
      For example the Sentence "aaabaaa" can be produced in two ways.



      Or like so.



   2. G=({S},{a,b},P,S) with productions
      S⟶ aaSb | abSbS | ε.

      The Grammar here is unambiguous due to the location and arrangement of the "terminal" letters "aa" and "ab". This means that each sentence that can be created will be unique.

6. Rewrite the following grammar to eliminate left-recursion:
   $\text{exp} \longrightarrow \text{exp+term}$ | exp – term | term.

   exp ::= term exp'
   exp' ::= + term exp' | - term exp' | ε.

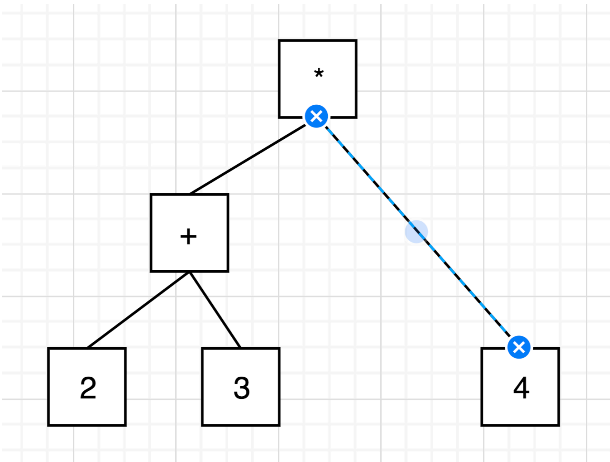
7. Write a grammar to parse arithmetic expressions (with + and ×) on integers (such as 0, 42, -1). This is a little harder than it first sounds because you will need to write a grammar to generate correctly-formatted positive and negative numbers; a grammar that generates -01 is unacceptable. You may use dots notation to indicate obvious replication of rules, as in "$\text{D} \longrightarrow 0|\cdots|9$", without comment.
   (Note that a parser is not an evaluator. This question is **not** asking you to write an evaluator. Also note that the grammar only needs to be unambiguous if the question asks you to provide an unambiguous grammar.)

   G=({S,D,A,B},{0,1,2,3,4,5,6,7,8,9,+, },S, with following production rules:*

   S ::= -D | D | 0 | 0B
   D ::= 1 | … | 9
   D ::= 1D | … | 9D
   D ::= 1A | … | 9A
   A ::= 0 | 0A | 0D | 0B
   *B ::= +S | *S*

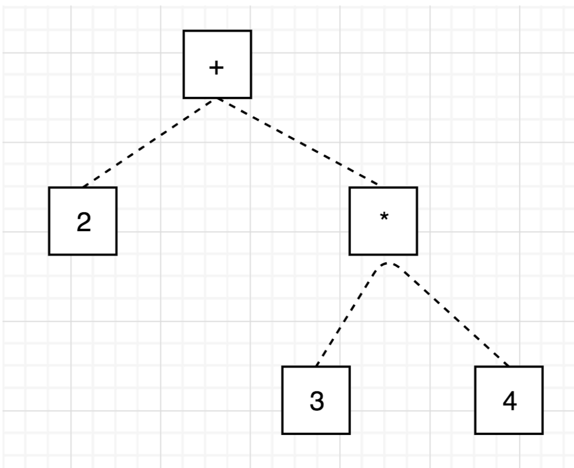
   }

8. Write two distinct parse trees for $2+3*4$ and explain in intuitive terms the significance of the two different parsing's to their denotation.



*The term 2 + 3 * 4 is ambiguous for the reader as they, themselves can't understand what the writer is intending.*

*The first tree shows the parse tree of (2 +3) * 4 which is equal to 20.*



*The next tree shows the expression 2 + (3 * 4) which is 14.*

In the real world we are taught through BODMAS/BIDMAS that we add brackets to the expression to denote to do multiplication before addition etc.

Not all interpreters however are a custom to this rule however and thus causes ambiguity.