Notes
<symbol> ::= __expression__
<symbol> is a non terminal
__expression__ consists of one or more sequences of terminal or non-terminal symbols
| indicates choice
ε indicates an empty string of length 0

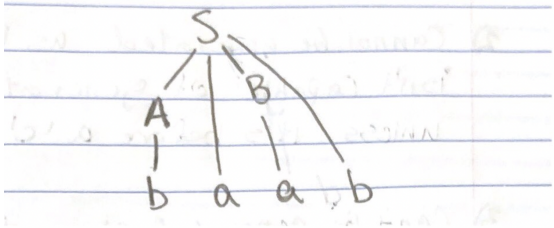
2a)
A ::= a|aA
This language dictates that A is a sentence of at least on ‘a’, but can be followed up by many ‘a’ characters.

B ::= ε|bBc|bb
This language dictates that B can be an empty sentence or it can be ‘bb’ or it can be a sentence starting with ‘b’ and ending with ‘c’ that can have several ‘b’ and ‘c’ in the middle.

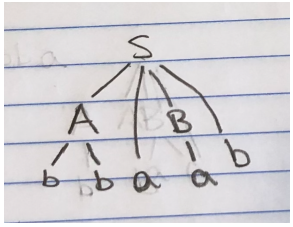
S ::= ABA
The language dictates that S starts with at least one ‘a’ and is followed up by a sentence that starts with ‘b’ and ends with ‘c’ with possibly many ‘b’ and ‘c’ characters in between. It ends with at least one ‘a’ but can be followed by many ‘a’ characters.

2b)

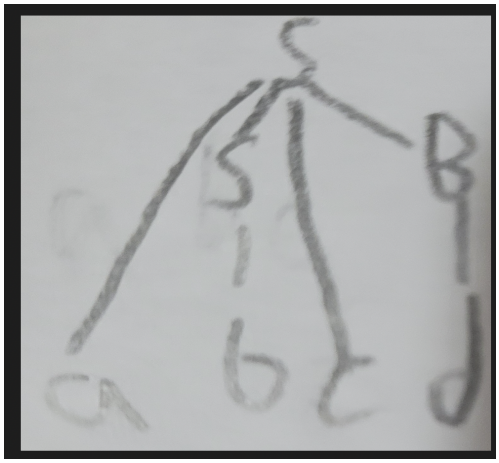
1) AaBb => baBb => baab

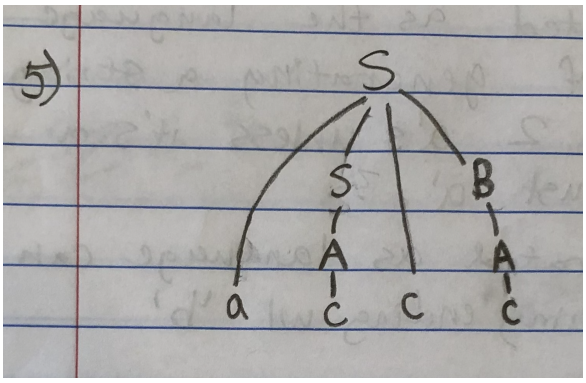


2) Cannot be generated since language cannot generate a string with less than 2 a’s unless it’s a string of only one a
3) Cannot be generated as language can only generate string with both b and c if it ends with b
4) AaBb => AbaBb => bbaBb => bbaab

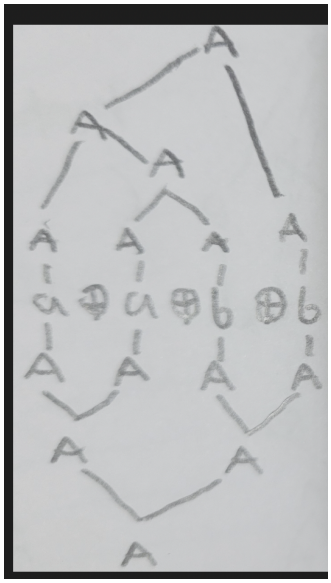


2c)

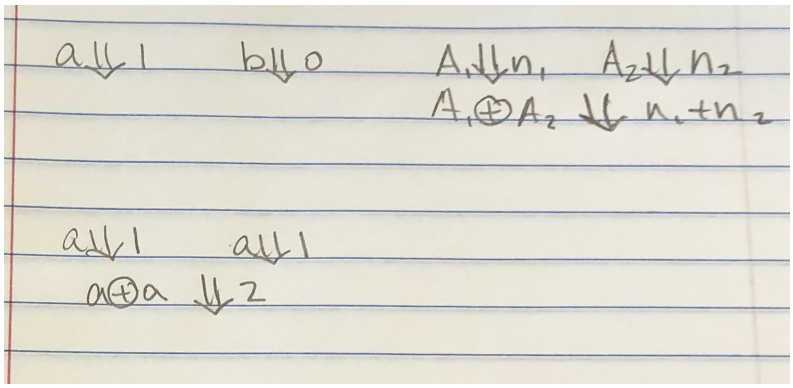
1) 

2) Cannot be generated as language isn't capable of generating a 'b' unless it's before a 'c'
3) Cannot be generated since there can't exist a string of c's with a b followed by two c's unless there's at least two a's in front of it
4) Cannot be generated as there is a string of length of 3 that starts with an a
5) 

2d)



2e)



3a)

i. The expressions generated by the first grammar is either a single operand or a number of multiple operands being operated on in a left-associative way. The second grammar generates either empty expressions, operands, or a number of multiple operands being operated on in a right-associative way. An operator can also be at the front of the expression in the second grammar, and a single operator can act on a single operand in the second grammar.

ii. No, they don't generate the same expressions as the first e expression is incapable of producing an empty result, while the second expression can. The first expression is also left-associative, while the second expression is right-associative. An operator can also be at the front of the expression in the second grammar, and a single operator can act on a single operand in the second grammar.

e operator operand => operand operator operand
operand esuffix => operand operator operand esuffix => operand operator operand ε

3b)
(10-4) << 2 returns value of 24
10 - (4 << 2) returns value of -6
10 - 4 << 2 returns value of 24

Therefore minus operation has higher precedence than left shift operator

3c)
F ::= N.N|-N.N|N.T|0.0
E ::= F^N|F^-N
T ::= 0|0T|TN
N ::= D|DN
D ::= 1|2|3|4|5|6|7|8|9