1. (Aho, Sethi, Ullman) Consider the grammar:

S ::= aSbS | bSaS | ε

a. Show that this grammar is ambiguous by constructing two **leftmost** derivations for the sentence abab.
b. Show that this grammar is ambiguous by constructing two **rightmost** derivations for the sentence abab.
c. Construct the corresponding parse trees for your derivations.

===

Rules:

1	aSbS
2	bSaS
3	ε

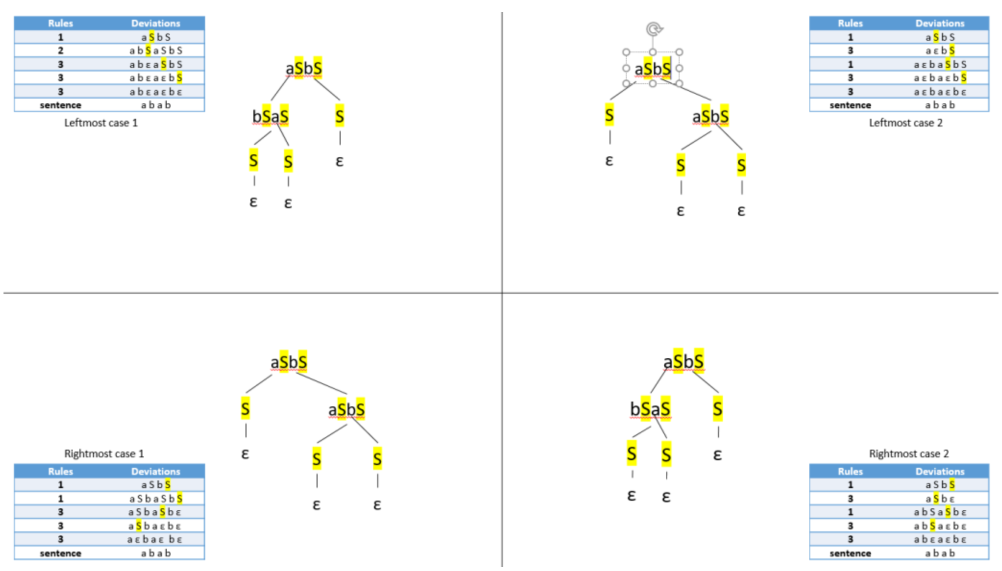
a. Leftmost derivations

Rules	Deviations	Rules	Deviations
1	a S b S	1	a S b S
2	a b S a S b S	3	a ε b S
3	a b ε a S b S	1	a ε b a S b S
3	a b ε a ε b S	3	a ε b a ε b S
3	a b ε a ε b ε	3	a ε b a ε b ε
sentence	a b a b	sentence	a b a b

b. Rightmost derivations

Rules	Deviations	Rules	Deviations
1	a S b S	1	a S b S
1	a S b a S b S	3	a S b ε
3	a S b a S b ε	1	a b S a S b ε
3	a S b a ε b ε	3	a b S a ε b ε
3	a ε b a ε b ε	3	a b ε a ε b ε
sentence	a b a b	sentence	a b a b

c. Parse trees:



2. (Aho, Sethi, Ullman) Given the following grammar:

S ::= `(` L `)` | `x`
L ::= L `,` S | S

a. Give a left-most derivation of `(x, (x, x))`.
b. Give a right-most derivation of `(x, (x, x))`.
c. Show the steps that a shift-reduce parser goes through when it parses `(x, x, x)`. That is, show the contents of the stack and remaining input at each step.
d. Suppose we replace the left-recursive production L::=L`,`S with a right-recursive one L::=S`,`L. What general effect does this have on the depth of the stack during a shift-reduce parse? (You might work through the parse of `(x, x, x)` again to see what changes.)

---

a. (see below)
b. (see below)

Leftmost derivation	Rightmost derivation
S	**S**
(**S**, S)	(S, **S**)
(x, **S**)	(S, (**L**))
(x, (**L**))	(S, (L, **S**))
(x, (**L**, S))	(S, (**L**, x))
(x, (**S**, S))	(S, (**S**, x))
(x, (x, **S**))	(**S**, (x, x))
(x, (x, x))	(x, (x, x))

c. L::=L, S|S

Steps	Stacks
0	(x, x, x)
1	(S, x, x)
2	(L, x, x)
3	(L, S, x)
4	(L, x)
5	(L, S)
6	(L)
7	S

d. L::=S, L|S

Steps	Stacks
0	(x, x, x)
1	(S, x, x)
2	(S, x, x)
3	(S, S, x)
4	(S, S, S)
5	(S, S, L)
6	(S, L)
7	(L)
8	S

The depth of the stack increases until we can find the handle, L::= S, L.

~~3. Postponed until hw3 (LR construction).~~

4. (Appel) Write a grammar for English sentences using the words

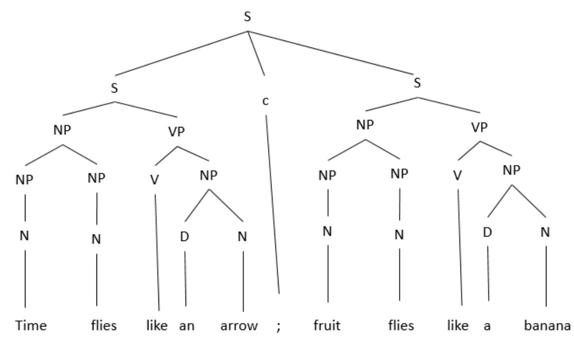
`time   arrow   banana   flies   like   a   an   the   fruit`

and the semicolon. Be sure to include all the senses (noun, verb, etc.) of each word. Then show that this grammar is ambiguous by exhibiting more than one parse tree for the sentence "time flies like an arrow; fruit flies like a banana". (Hint: do not go overboard. This is not an exercise in computational linguistics or Natural Language Processing. If you've done "sentence diagrams" in grade school, that's the right level of complexity for the grammar here.)

Time flies like an arrow; fruit flies like a banana.

G:

```
S::= NP VP
NP::= N | DN | NN
VP::= V PP| V NP | V
PP::= P NP
V::= time | flies | like
P:: = like
D::= a | an | the
N::= time | arrow | banana | fruit | flies
C::= ;
```



The grammar is ambiguous because there are more than one parse tree can be generated by leftmost derivation.