Homework of Chapter 3

Ex 3.2

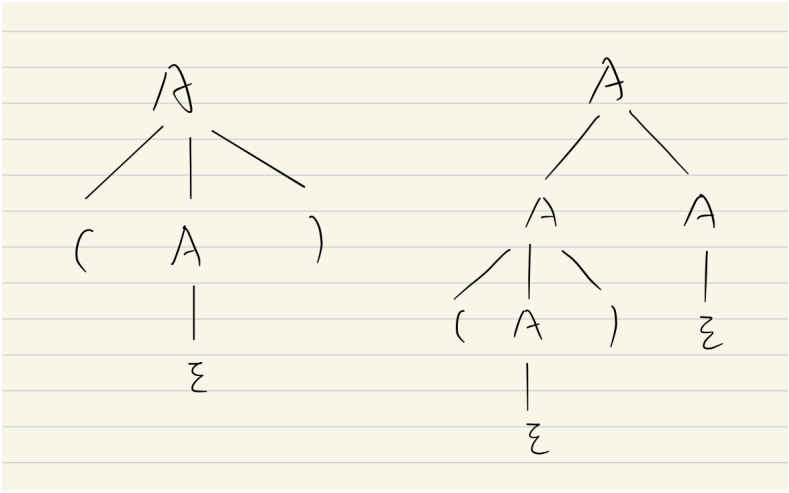
> Given the grammar A→AA|(A)|ε,
> a. Describe the language it generates.
> b. Show it is ambiguous.

Solution:

a.

This grammar generates a series of matched parentheses and the empty string.

b.



Ex 3.3

> Given the grammar:
> exp→ exp addop term | term
> addop→ + | -
> term → term mulop factor | factor
> mulop→ *
> factor → (exp) | number

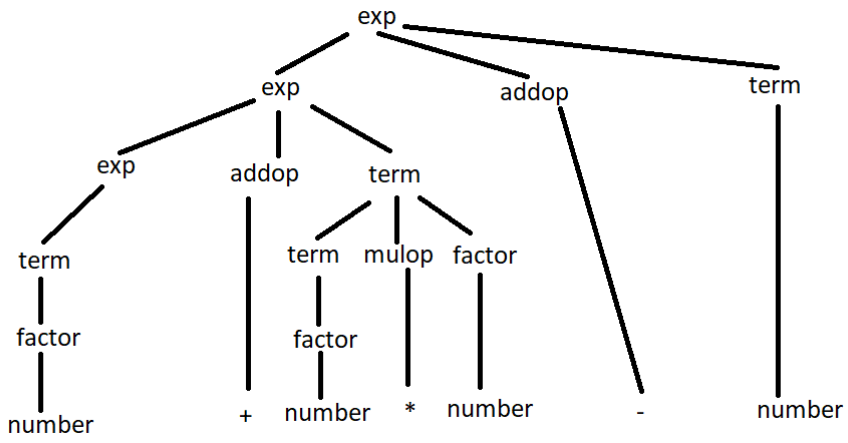
> Write down leftmost derivation, parse trees, and absctractsyntax trees for the following expressions:
> (a) 3+4*5-6

Solution:

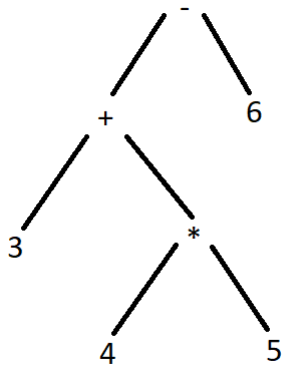
(a)

The leftmost derivation:
exp => exp addop term
=> exp addop term addop term
=> term addop term addop term
=> factor addop term mulop factor addop term
=> 3 + factor * 5 - factor
=> 3 +4 * 5 - 6

parse tree:



abstract syntax tree:



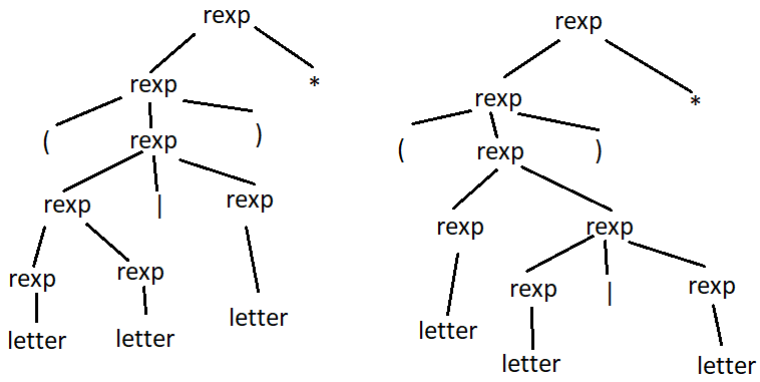
Ex 3.4

> The following grammar generates all regular expressions over the alphabet of letters (we have to use quotes to surround operators, since the vertical bar is an operator as well as a meta symbol):
> rexp→ rexp“|” rexp
> | rexp rexp
> | rexp “*”
> | “(” rexp “)”
> | letter
> a. Give a derivation for the regular expression (ab|a)* using this grammar.
> b. Show that this grammar is ambiguous.
> c. Rewrite this grammar to establish the correct precedencesfor the operations (see chapter 2).
> d. What associativity does your answer in part (c) give to the binary operations? Why?

Solution:

a.
rexp => rexp*
=> (rexp)*
=> (rexp | rexp)*
=> (rexp rexp | rexp)*
=> (letter letter | letter)*
=> (ab|a)*

b.



c.

rexp → rexp”*” |“(”rexp”)”” |term“|”term

term → term term | letter

d.

The “or” operation is processed after the apposition.