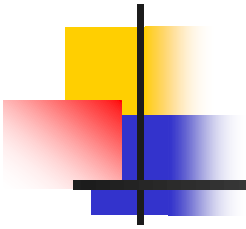
CSC 3201 Compiler Construction

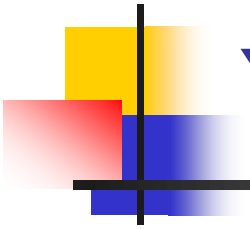
Department of Computer Science
SZABIST (Islamabad Campus)

Week 5 (Lecture 1)



Parser Generators

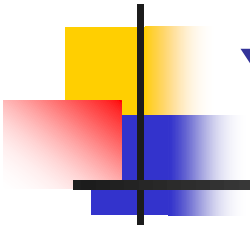
- LALR(1):
  - YACC
  - Bison
  - CUPDirect
- LL(1):
  - ANTLR
- Recursive Descent:
  - Java CC



YACC

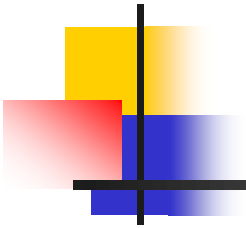
- Yet Another Compiler Compiler
- Unix application – 1975
- Structure of specification text file:

definitions
%%
rules
%%
C/C++ functions



YACC File

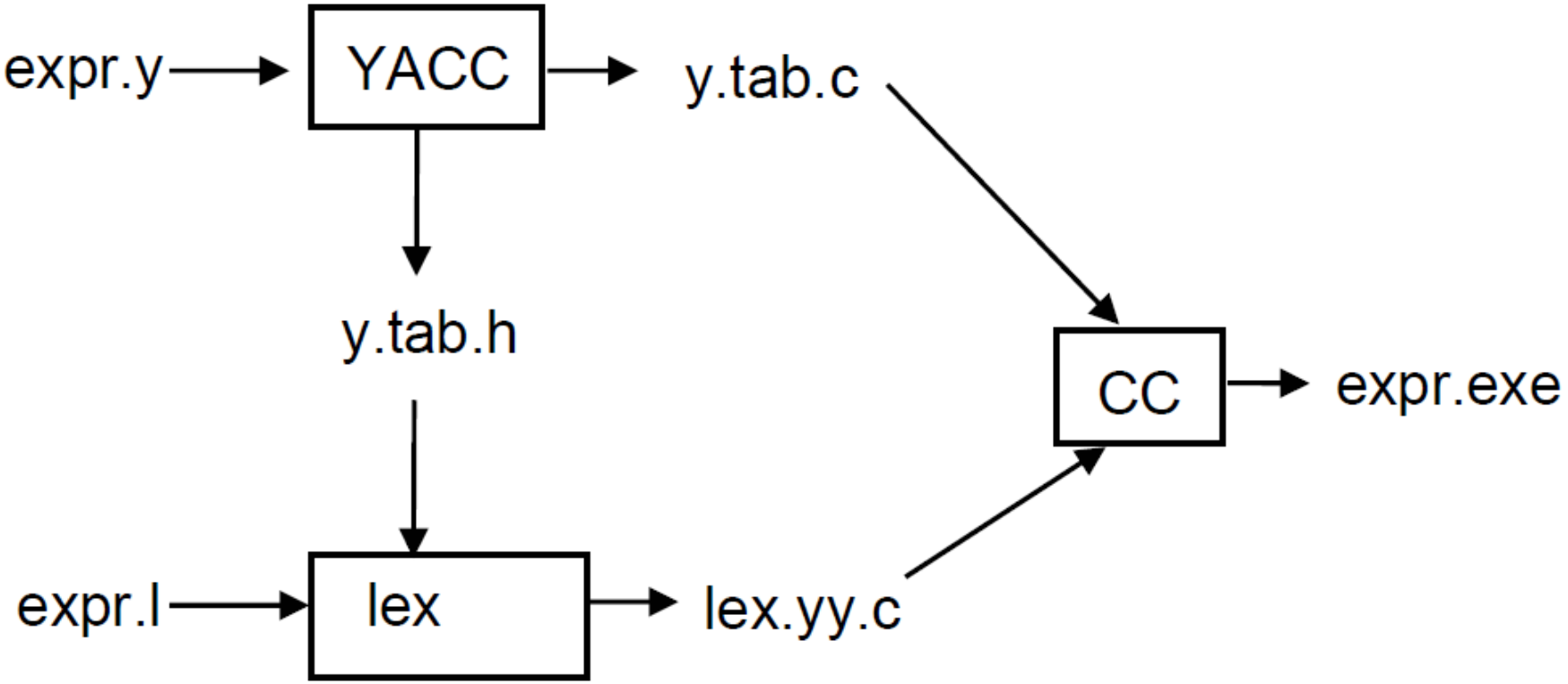
```
%token NUMBER LPAREN RPAREN
%token PLUS MINUS TIMES DIVIDE
%%
expr : expr PLUS expr
                | expr MINUS expr
                | expr TIMES expr
                | expr DIVIDE expr
                | LPAREN expr RPAREN
                | MINUS expr
                | NUMBER
                ;
%%
```



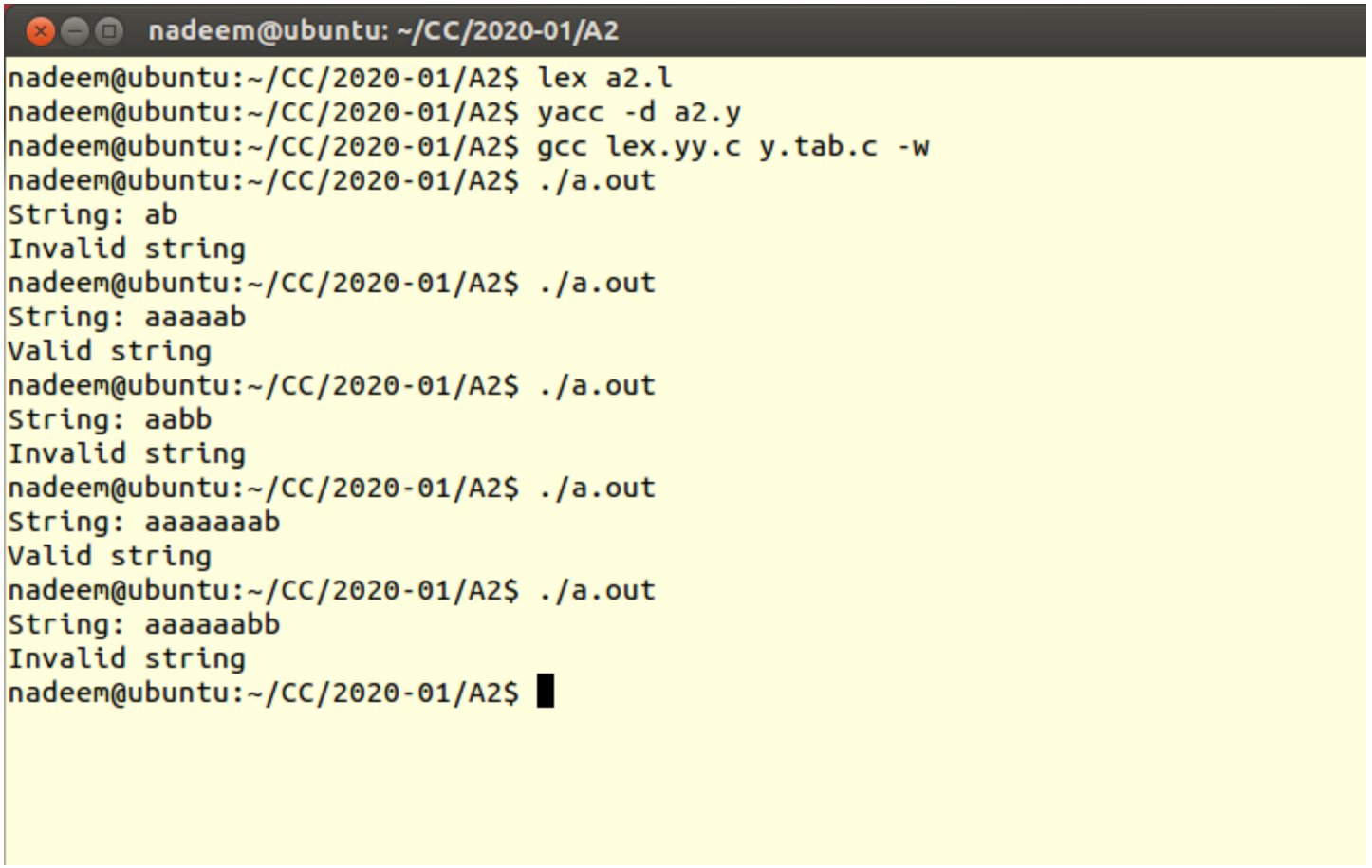
Lex File

```
%{
#include "y.tab.h"
%}
digit           [0-9]
ws              [ \t\n]+
%%
{ws}            ;
{digit}+   {return NUMBER;}
"+"             {return PLUS;}
"*"             {return TIMES;}
"/"             {return DIVIDE;}
"-"             {return MINUS;}
%%
```

Lex and YACC



Lex and YACC


nadeem@ubuntu: ~/CC/2020-01/A2
nadeem@ubuntu:~/CC/2020-01/A2$ lex a2.l
nadeem@ubuntu:~/CC/2020-01/A2$ yacc -d a2.y
nadeem@ubuntu:~/CC/2020-01/A2$ gcc lex.yy.c y.tab.c -w
nadeem@ubuntu:~/CC/2020-01/A2$ ./a.out
String: ab
Invalid string
nadeem@ubuntu:~/CC/2020-01/A2$ ./a.out
String: aaaaab
Valid string
nadeem@ubuntu:~/CC/2020-01/A2$ ./a.out
String: aabb
Invalid string
nadeem@ubuntu:~/CC/2020-01/A2$ ./a.out
String: aaaaaaab
Valid string
nadeem@ubuntu:~/CC/2020-01/A2$ ./a.out
String: aaaaaabb
Invalid string
nadeem@ubuntu:~/CC/2020-01/A2$

Grammars

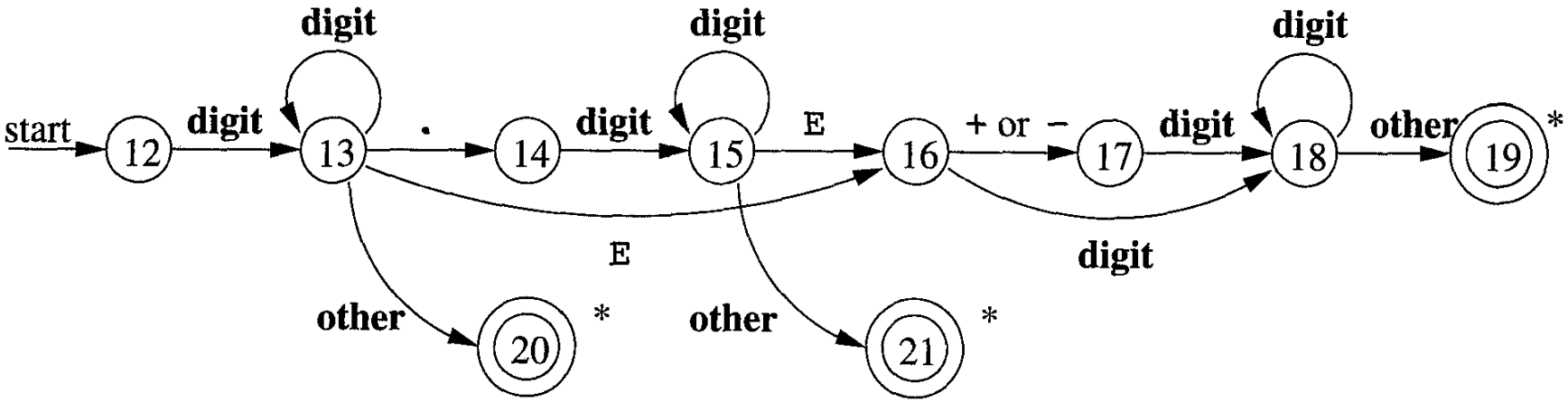
- RE:

```
digit = 0|1|2|3|4|5|6|7|8|9
number = digit digit*
```

- BNF:

$$digit \rightarrow 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$$
$$number \rightarrow number\ digit \mid digit$$

Transition Diagrams



- Excercises:
  - Identifiers
  - Relational Operators
  - if

Syntax Diagrams

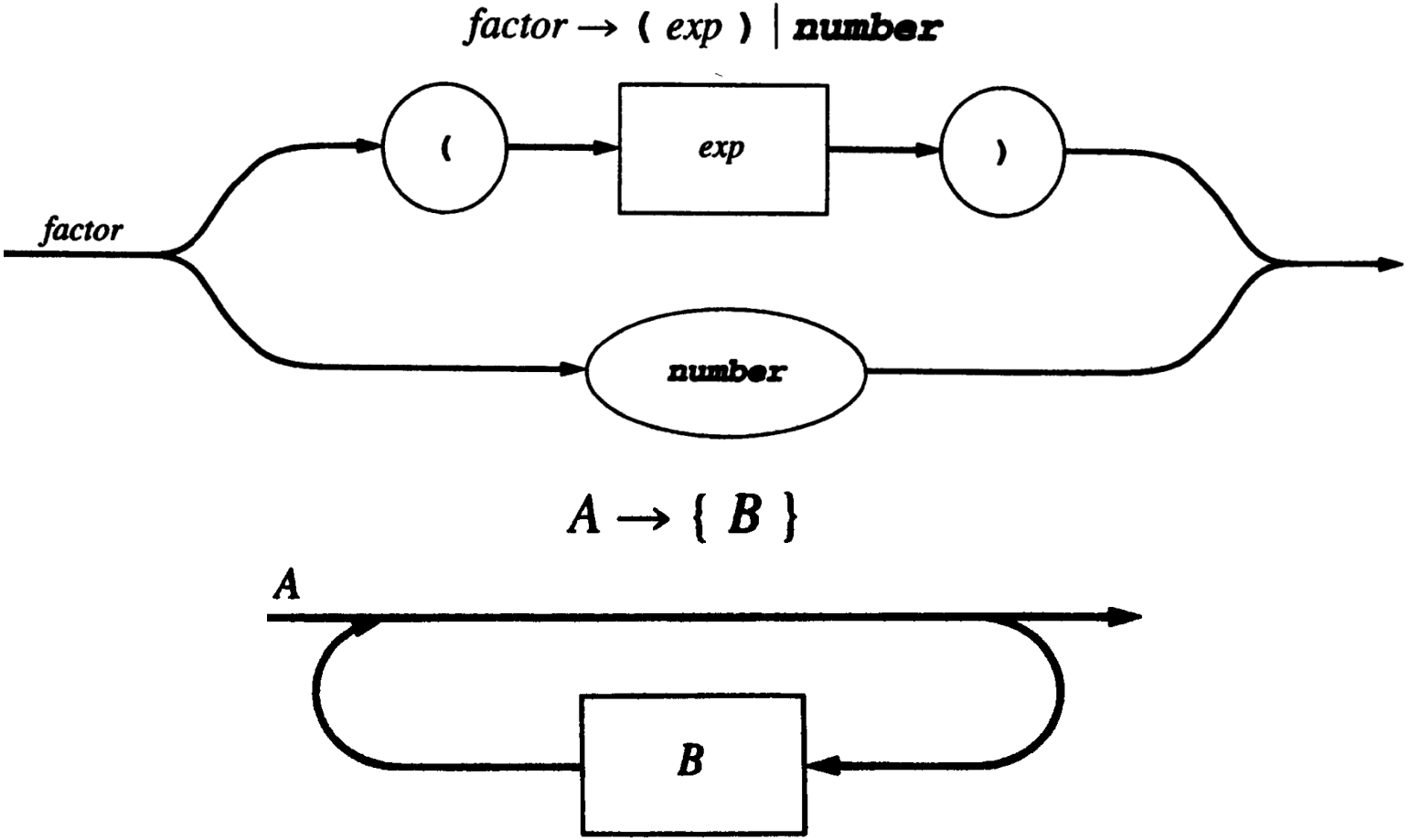
- BNF:

statement → *if-stmt* | **other**
if-stmt → **if** (*exp*) *statement*
| **if** (*exp*) *statement* **else** *statement*
exp → **0** | **1**

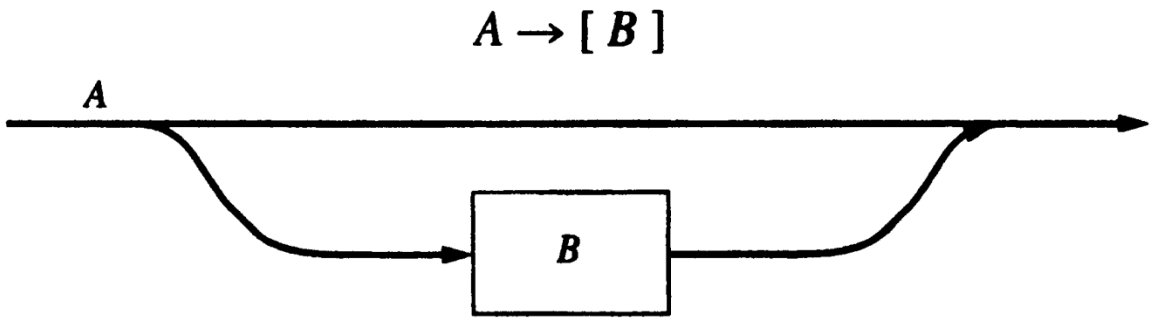
- EBNF:

statement → *if-stmt* | **other**
if-stmt → **if** (*exp*) *statement* [**else** *statement*]
exp → **0** | **1**

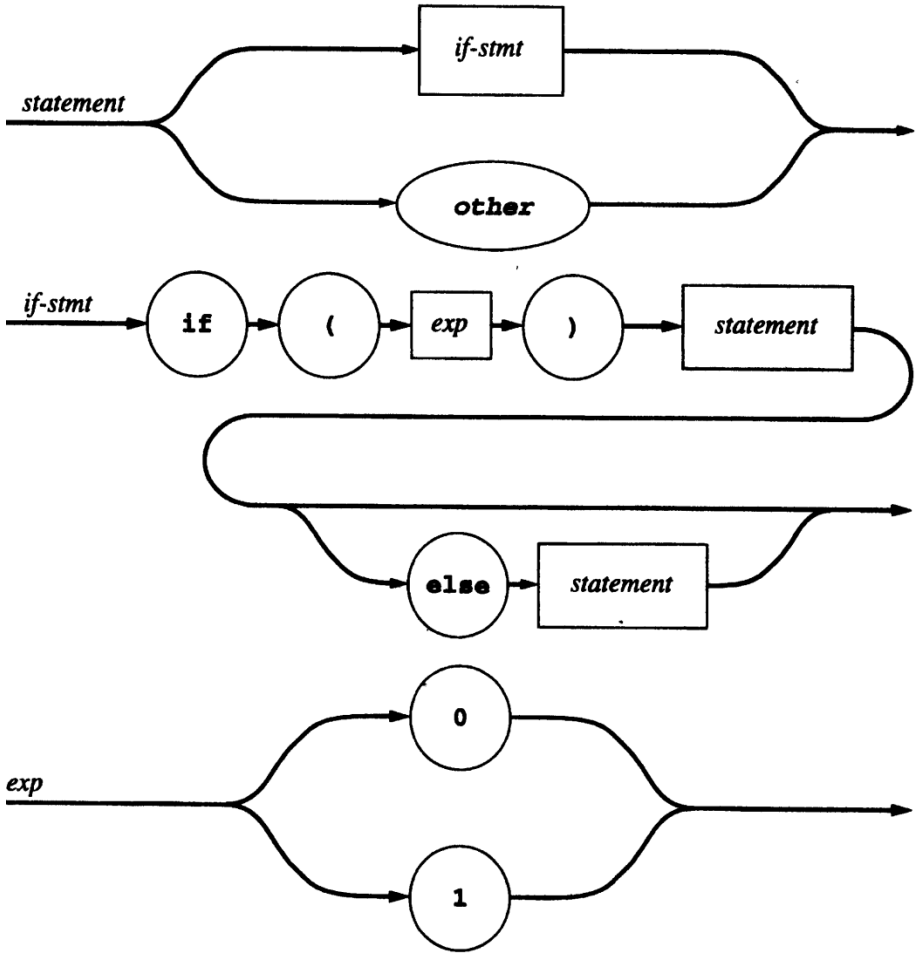
Syntax Diagrams



Syntax Diagrams



Syntax Diagrams



Syntax Diagrams

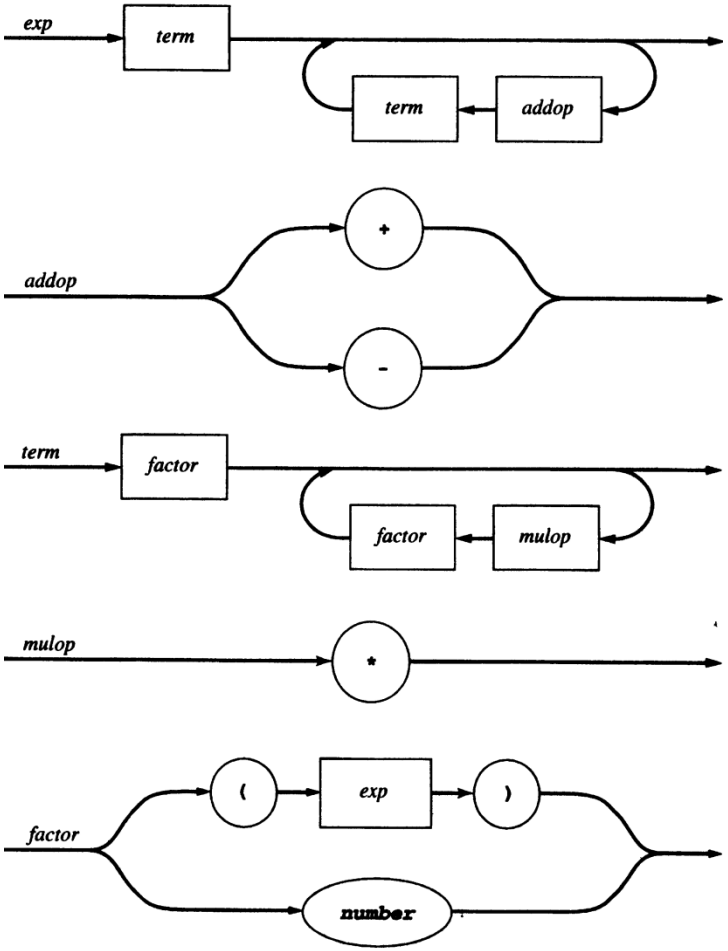
- BNF:

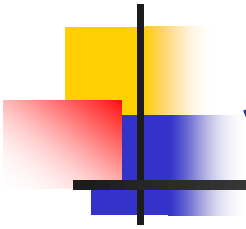
$$exp \rightarrow exp\ addop\ term \mid term$$
$$addop \rightarrow + \mid -$$
$$term \rightarrow term\ mulop\ factor \mid factor$$
$$mulop \rightarrow *$$
$$factor \rightarrow (\ exp\ ) \mid \mathbf{number}$$

- EBNF:

$$exp \rightarrow term\ \{\ addop\ term\ \}$$
$$addop \rightarrow + \mid -$$
$$term \rightarrow factor\ \{\ mulop\ factor\ \}$$
$$mulop \rightarrow *$$
$$factor \rightarrow (\ exp\ ) \mid \mathbf{number}$$

Syntax Diagrams





Syntax Trees

```
read x; { input an integer }
if 0 < x then { don't compute if x <= 0 }
   fact := 1;
   repeat
      fact := fact * x;
      x := x - 1
   until x = 0;
   write fact { output factorial of x }
end
```

Syntax Trees