MO403/MC900

Implementation of Programming Languages

2nd semester 2020

Grammar and Syntax Diagrams for SL – Simple Language

Tomasz Kowaltowski

Instituto de Computação

UNICAMP

Notation

$[\ \ldots\ ]$	optional
$\ldots^*$	repetition possibly zero times
$\ldots^+$	repetition at least once
$\ldots \mid \ldots$	alternative
$(\ \ldots\ )$	grouping
♣	full Simple Language

Predefined identifiers

integer
boolean
true
false
read
write

Reserved words

else
functions
goto
if
labels
return
types
var
vars
void
while

Symbols

```
(     +
)     -
{     *
}     /
[     ,
]     &&
:     ||
;     <=
=     <
!     ==
!=    >=
      >
```

Remarks

- Identifiers and reserved words are case insensitive.
- Lexical items are shown for completeness only; they will be implemented separately from the grammar productions.
- Any characters between '`//`' and end-of-line are considered to be a comment.

Grammar productions

program ⇒ *function*

function ⇒ *function_header* *block*

function_header ⇒ (*identifier* | **void**) *identifier* *formal_parameters*

block ⇒ [*labels*] [♣ *types*] [*variables*] [*functions*] body

labels ⇒ **labels** *identifier_list* ;

♣ *types* ⇒ **types** (*identifier* = *type* ;)$^+$

variables ⇒ **vars** (*identifier_list* : *type*)$^+$;

functions ⇒ **functions** *function*$^+$

body ⇒ { *statement* * }

type ⇒ *identifier* ♣ ([*integer*])*

formal_parameters ⇒ (*formal_parameter* (, *formal_parameter*)*) | ()

formal_parameter ⇒ *expression_parameter* ♣ | *function_parameter*

expression_parameter ⇒ [**var**] *identifier_list* : *identifier*

♣ *function_parameter* ⇒ *function_header*

statement ⇒ [*identifier* :] *unlabeled_statement*

variable ⇒ *identifier* ♣ ([*expression*])*

unlabeled_statement ⇒ *assignment* | *function_call_statement* | *goto* | *return* |
conditional | *repetitive* | *compound* | *empty_statement*

assignment ⇒ *variable* = *expression* ;

function_call_statement ⇒ *function_call* ;

goto ⇒ **goto** *identifier* ;

return ⇒ **return** [*expression*] ;

conditional ⇒ **if** (*expression*) *compound* [**else** *compound*]

repetitive ⇒ **while** (*expression*) *compound*

compound ⇒ { *unlabeled_statement** }

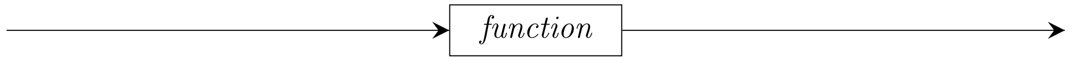
$$\begin{array}{lcl}
\mathit{empty\_statement} & \Rightarrow & \texttt{;} \\
\mathit{expression} & \Rightarrow & \mathit{simple\_expression}\ \Big[\ \mathit{relational\_operator}\ \ \mathit{simple\_expression}\ \Big]\ \ \Big| \\
 & & \mathit{unop\_expression}\ \Big[\ \mathit{relational\_operator}\ \ \mathit{simple\_expression}\ \Big] \\
\mathit{simple\_expression} & \Rightarrow & \mathit{term}\ \Big(\ \mathit{additive\_operator}\ \ \mathit{term}\ \Big)^* \\
\mathit{unop\_expression} & \Rightarrow & \mathit{unary\_operator}\ \ \mathit{term}\ \Big(\ \mathit{additive\_operator}\ \ \mathit{term}\ \Big)^* \\
\mathit{relational\_operator} & \Rightarrow & \texttt{<=}\ \Big|\ \texttt{<}\ \Big|\ \texttt{==}\ \Big|\ \texttt{!=}\ \Big|\ \texttt{>=}\ \Big|\ \texttt{>} \\
\mathit{additive\_operator} & \Rightarrow & \texttt{+}\ \Big|\ \texttt{-}\ \Big|\ \texttt{||} \\
\mathit{unary\_operator} & \Rightarrow & \texttt{+}\ \Big|\ \texttt{-}\ \Big|\ \texttt{!} \\
\mathit{term} & \Rightarrow & \mathit{factor}\ \Big(\ \mathit{multiplicative\_operator}\ \ \mathit{factor}\ \Big)^* \\
\mathit{multiplicative\_operator} & \Rightarrow & \texttt{*}\ \Big|\ \texttt{/}\ \Big|\ \texttt{\&\&} \\
\mathit{factor} & \Rightarrow & \mathit{variable}\ \Big|\ \mathit{integer}\ \Big|\ \mathit{function\_call}\ \Big|\ \texttt{(}\ \ \mathit{expression}\ \ \texttt{)} \\
\mathit{function\_call} & \Rightarrow & \mathit{identifier}\ \ \texttt{(}\ \ \mathit{expression\_list}\ \ \texttt{)} \\
\mathit{identifier\_list} & \Rightarrow & \mathit{identifier}\ \Big(\ ,\ \mathit{identifier}\Big)^* \\
\mathit{expression\_list} & \Rightarrow & \mathit{expression}\ \Big(\ ,\ \mathit{expression}\Big)^* \\
\mathit{empty} & \Rightarrow &
\end{array}$$

Lexical items:

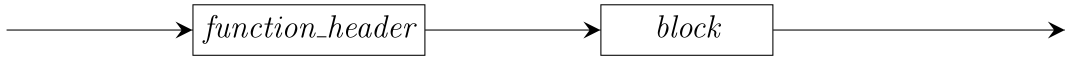
$$\begin{array}{lcl}
\mathit{integer} & \Rightarrow & \mathit{digit}^+ \\
\mathit{identifier} & \Rightarrow & \mathit{letter}\ \Big(\ \mathit{letter}\ \Big|\ \mathit{digit}\Big)^* \\
\mathit{digit} & \Rightarrow & \texttt{0}\ \Big|\ \texttt{1}\ \Big|\ \texttt{2}\ \Big|\ \texttt{3}\ \Big|\ \texttt{4}\ \Big|\ \texttt{5}\ \Big|\ \texttt{6}\ \Big|\ \texttt{7}\ \Big|\ \texttt{8}\ \Big|\ \texttt{9} \\
\mathit{letter} & \Rightarrow & \texttt{a}\ \Big|\ \texttt{b}\ \Big|\ \texttt{c}\ \Big|\ \texttt{d}\ \Big|\ \texttt{e}\ \Big|\ \texttt{f}\ \Big|\ \texttt{g}\ \Big|\ \texttt{h}\ \Big|\ \texttt{i}\ \Big|\ \texttt{j}\ \Big|\ \texttt{k}\ \Big|\ \texttt{l}\ \Big|\ \texttt{m}\ \Big| \\
 & & \texttt{n}\ \Big|\ \texttt{o}\ \Big|\ \texttt{p}\ \Big|\ \texttt{q}\ \Big|\ \texttt{r}\ \Big|\ \texttt{s}\ \Big|\ \texttt{t}\ \Big|\ \texttt{u}\ \Big|\ \texttt{v}\ \Big|\ \texttt{w}\ \Big|\ \texttt{x}\ \Big|\ \texttt{y}\ \Big|\ \texttt{z}
\end{array}$$

Syntax diagrams

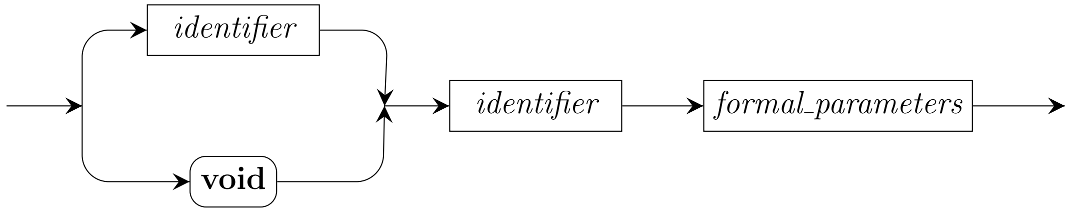
program



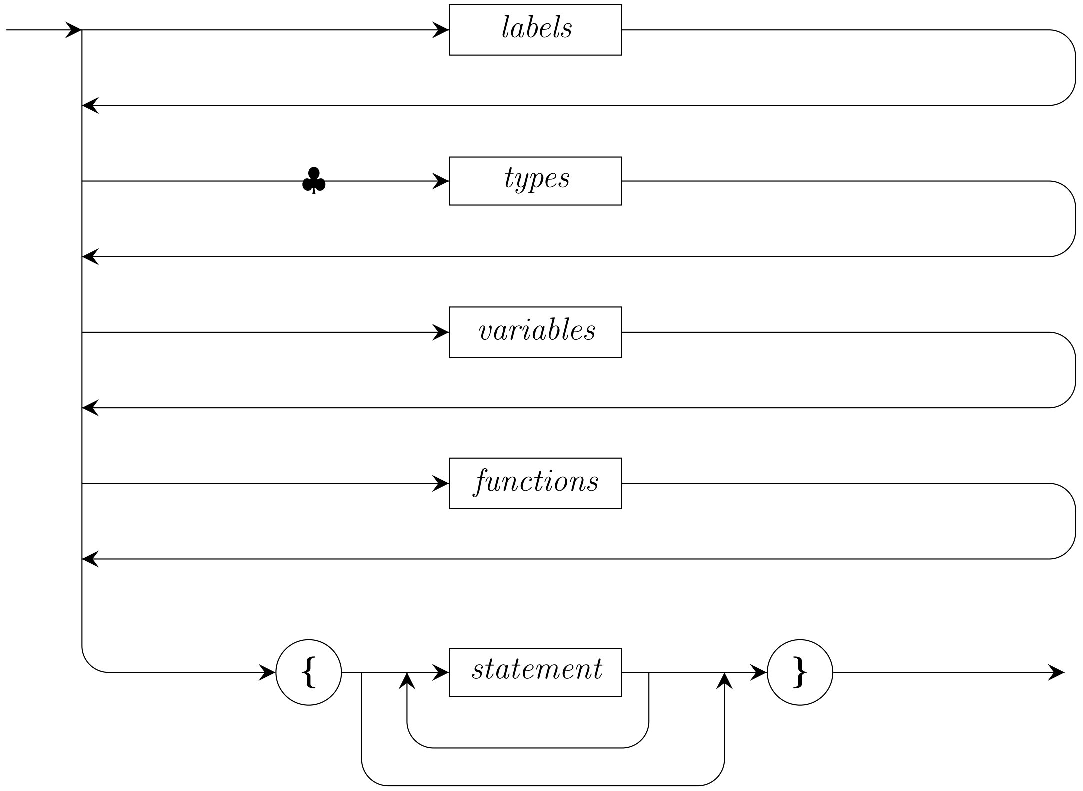
function



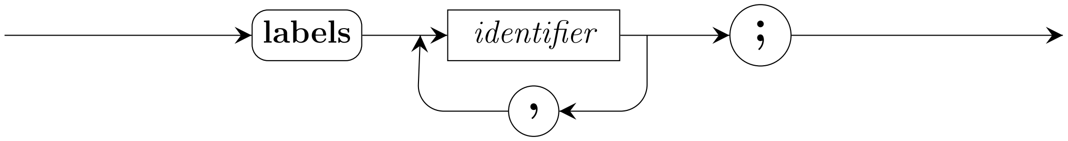
function_header



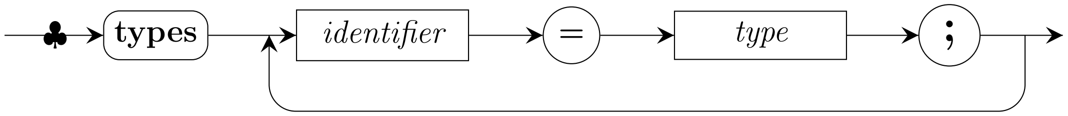
block



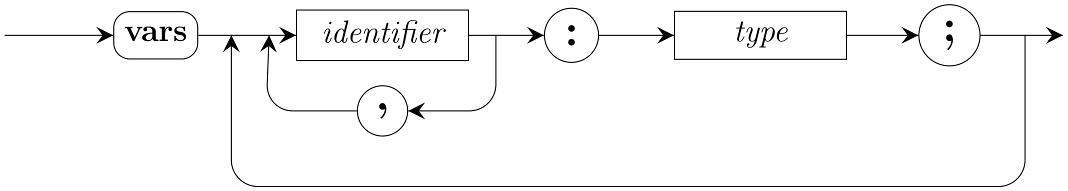
labels



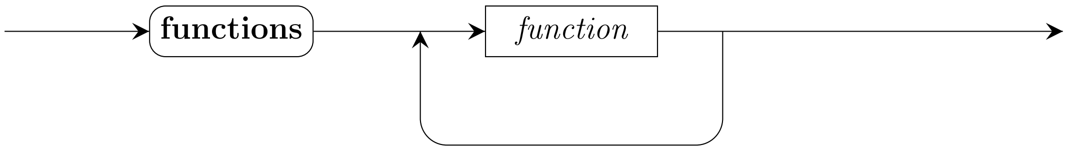
types



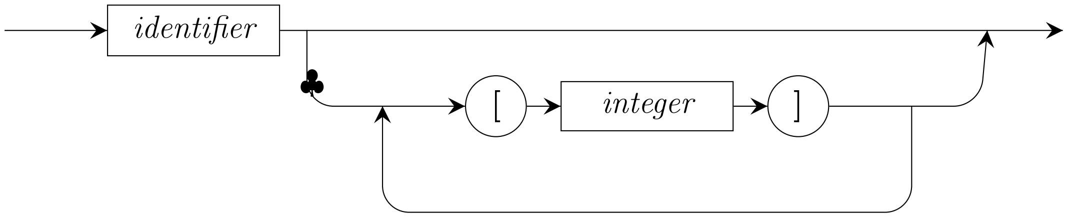
variables



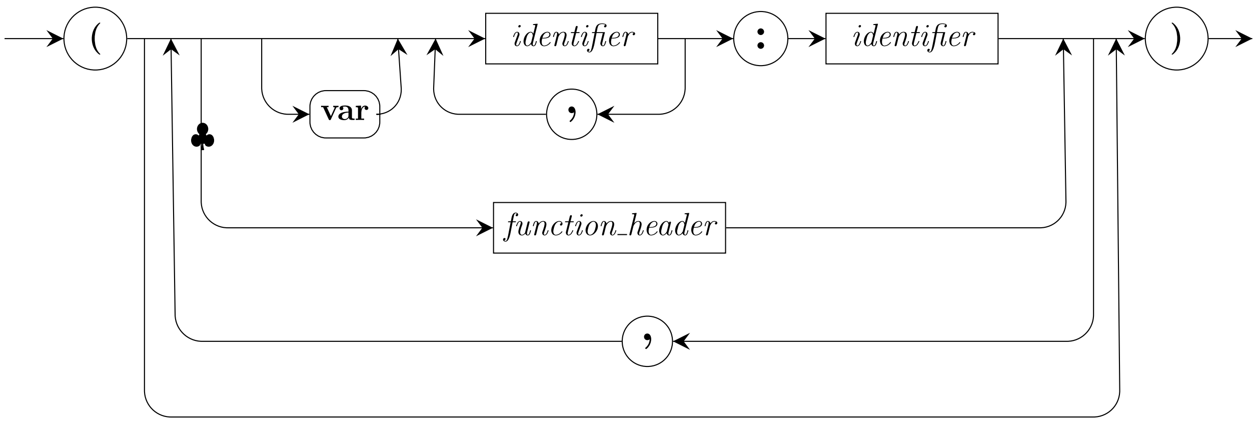
functions



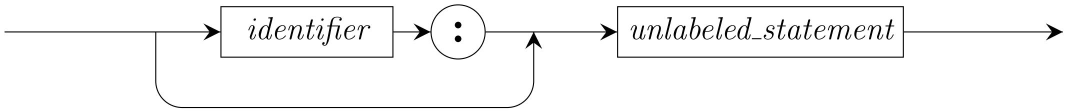
type



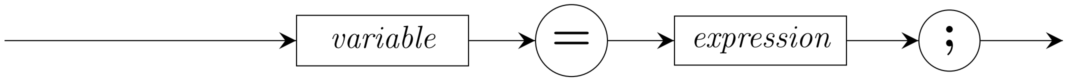
formal_parameters



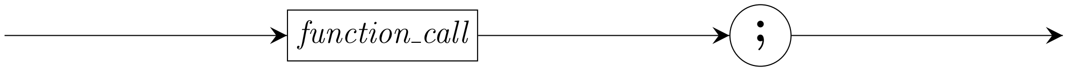
statement



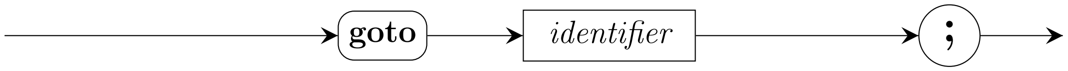
assignment



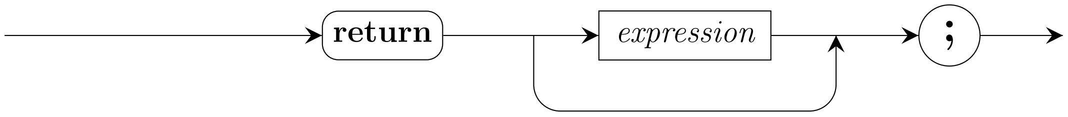
function_call_statement



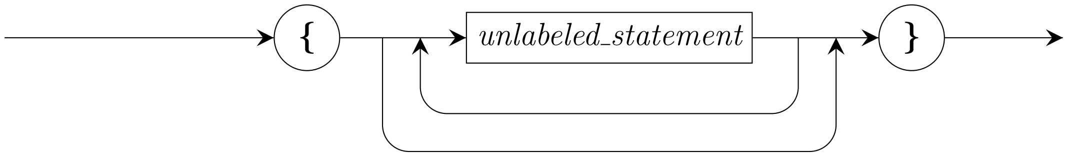
goto



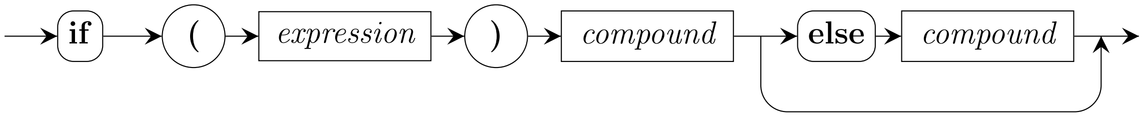
return



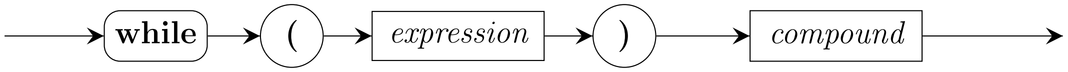
compound



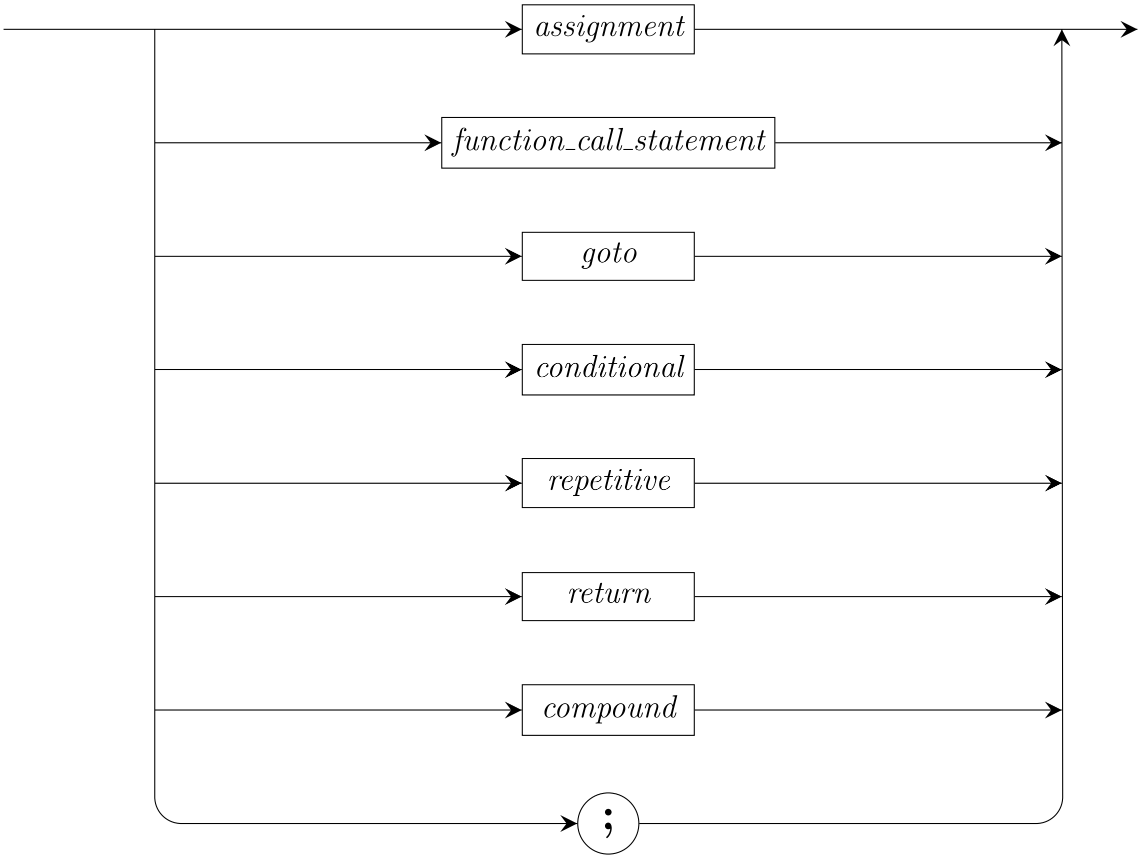
conditional



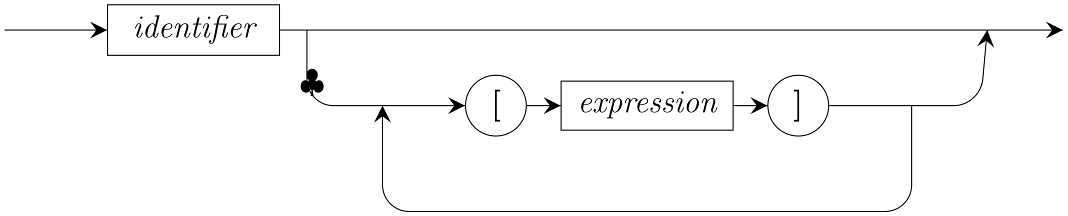
repetitive



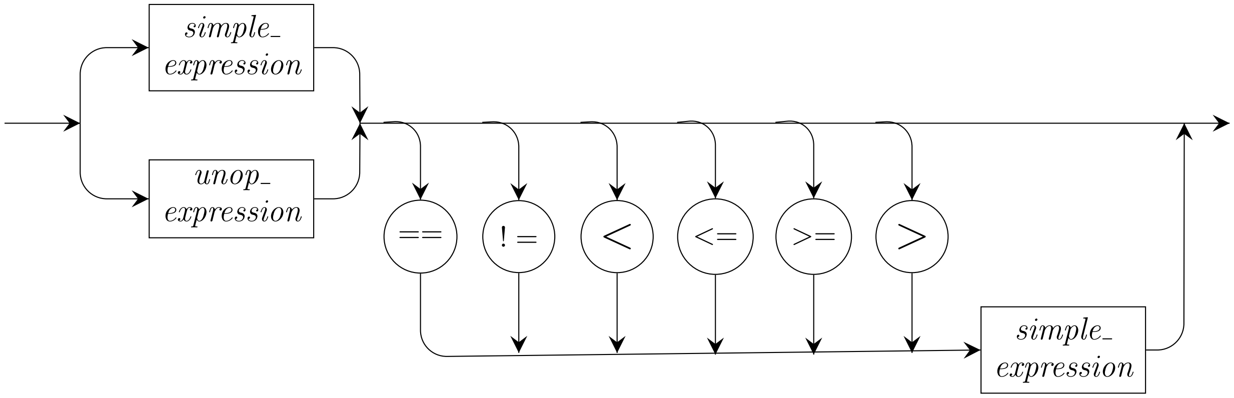
unlabeled_statement



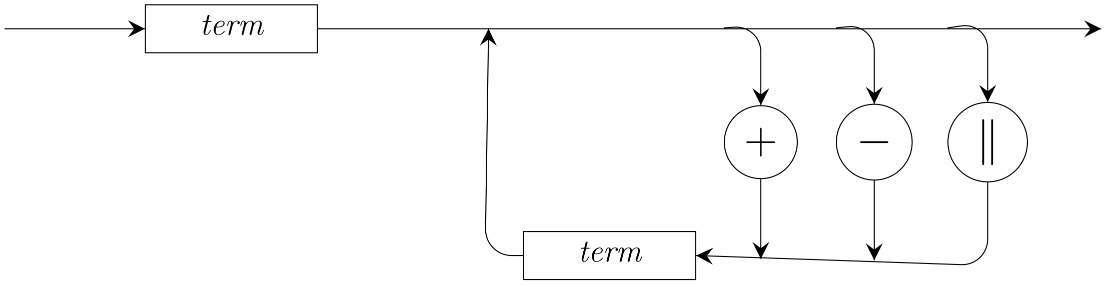
variable



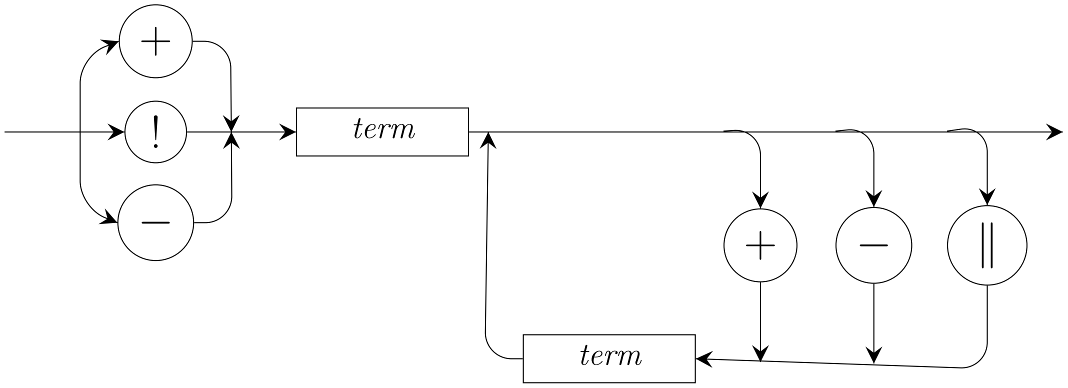
expression



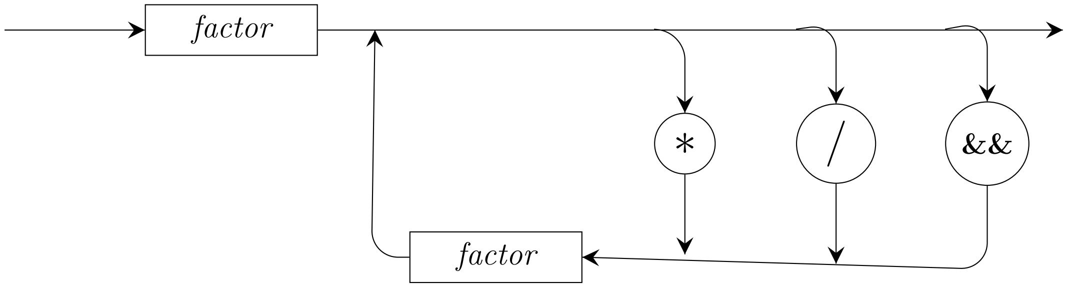
simple_expression



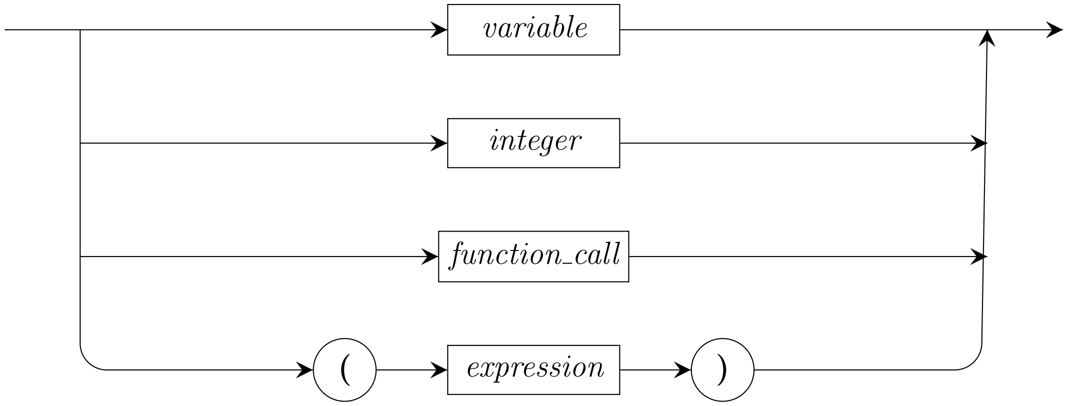
unop_expression



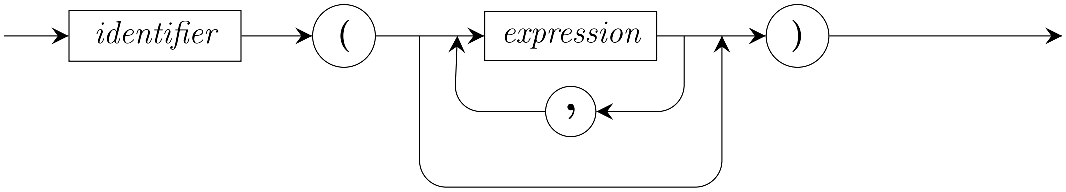
term



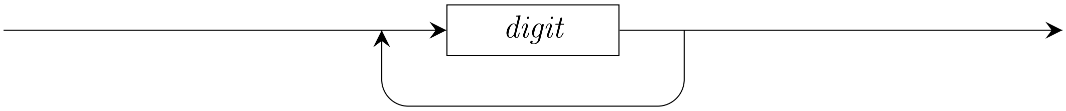
factor



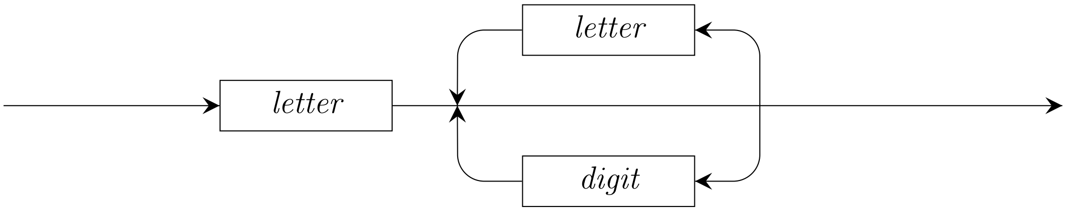
function_call



integer



identifier



Processed on June 18, 2020