$\langle program \rangle \rightarrow \langle roots \rangle$

$\langle roots \rangle \rightarrow \langle roots \rangle$
  | $\langle root \rangle \langle roots \rangle$

$\langle root \rangle \rightarrow \langle function \rangle$
  | $\langle dcl \rangle$;

$\langle dcl \rangle \rightarrow \langle type \rangle \langle id \rangle \mid \langle type \rangle \langle assign \rangle$

$\langle stmts \rangle \rightarrow \langle stmt \rangle$
  | $\langle stmt \rangle \langle stmts \rangle$

$\langle stmt \rangle \rightarrow \langle assign \rangle$;
  | $\langle if \rangle$
  | $\langle while \rangle$
  | $\langle from \rangle$
  | $\langle empty \rangle$
  | $\langle dcl \rangle$;
  | $\langle functioncall \rangle$;
  | $\langle switch \rangle$

$\langle id \rangle \rightarrow \langle letter \rangle$
  | $\langle id \rangle \langle letter \rangle$
  | $\langle id \rangle \langle digit \rangle$

$\langle letter \rangle \rightarrow$ [a-z]
  | [A-Z]

$\langle digit \rangle \rightarrow$ [0-9]

$\langle assign \rangle \rightarrow \langle id \rangle$ <-- $\langle expr \rangle$

$\langle expr \rangle \rightarrow \langle expr \rangle + \langle term \rangle$
  | $\langle expr \rangle - \langle term \rangle$
  | $\langle term \rangle$

$\langle term \rangle \rightarrow \langle term \rangle * \langle factor \rangle$
  | $\langle term \rangle / \langle factor \rangle$
  | $\langle factor \rangle$

$\langle factor \rangle \rightarrow (\langle expr \rangle)$
  | $\langle id \rangle$

$\langle switch \rangle \rightarrow$ switch $(\langle expr \rangle)$ begin $\langle cases \rangle$ end

$\langle cases \rangle \rightarrow$ case $\langle expr \rangle$: $\langle stmts \rangle \langle endcase \rangle$

⟨*endcase*⟩ → ⟨*cases*⟩
   | break;
   | break; ⟨*cases*⟩
   | default: ⟨*stmts*⟩ break;
   | break; default: ⟨*stmts*⟩ break;

⟨*from*⟩ → from ⟨*expr*⟩ to ⟨*logexpr*⟩ step ⟨*expr*⟩ begin ⟨*stmts*⟩ end

⟨*while*⟩ → while(⟨*logexpr*⟩) begin ⟨*stmts*⟩ end

⟨*if*⟩ → if(⟨*logexpr*⟩) begin ⟨*stmts*⟩ ⟨*endif*⟩

⟨*endif*⟩ → end else ⟨*if*⟩
   | end else begin ⟨*stmts*⟩ end
   | end

⟨*logexpr*⟩ → ⟨*logexpr*⟩ OR ⟨*andcomp*⟩
   | ⟨*andcomp*⟩

⟨*andcomp*⟩ → ⟨*andcomp*⟩ AND ⟨*comp*⟩
   | ⟨*comp*⟩

⟨*comp*⟩ → ⟨*boolean-operand*⟩ ⟨*comparison-operator*⟩ ⟨*boolean-operand*⟩

⟨*boolean-operand*⟩ → true
   | false
   | ⟨*expr*⟩
   | ⟨*boolean*⟩

⟨*boolean*⟩ → !(⟨*logexpr*⟩)
   | (⟨*logexpr*⟩)

⟨*comparison-operator*⟩ → >
   | <
   | <=
   | >=
   | !=
   | =

⟨*type*⟩ → ⟨*primitive-type*⟩
   | ⟨*array-type*⟩

⟨*primitive-type*⟩ → ⟨*numeric-type*⟩
   | bool

⟨*numeric-type*⟩ → ⟨*integral-type*⟩
   | double

$\langle integral\text{-}type \rangle \rightarrow$ int
   | char

$\langle array\text{-}type \rangle \rightarrow \langle type \rangle$ [ ]

$\langle function \rangle \rightarrow$ function $\langle id \rangle$ returns $\langle type \rangle$ params $(\langle params \rangle)$ begin $\langle stmts \rangle$ return $\langle expr \rangle$ end
   | function $\rightarrow$ function $\langle id \rangle$ returns nothing params $(\langle params \rangle)$ begin $\langle stmts \rangle$ end

$\langle params \rangle \rightarrow \langle subparams \rangle$
   | $\langle empty \rangle$

$\langle subparams \rangle \rightarrow \langle type \rangle \langle id \rangle, \langle subparams \rangle$
   | $\langle type \rangle \langle id \rangle$

$\langle functioncall \rangle \rightarrow$ call $\langle id \rangle(\langle params \rangle)$