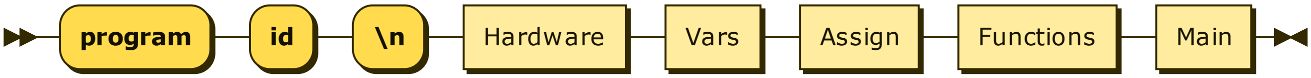
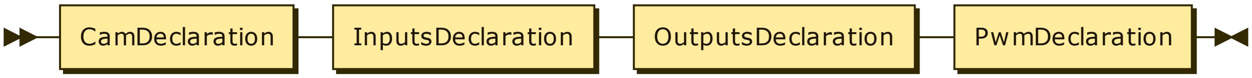
Program:



```
Program  ::= 'program' 'id' '\n' Hardware Vars Assign Functions Main
```

no references

Hardware:

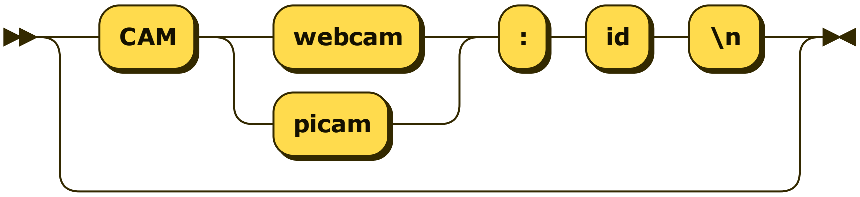


```
Hardware ::= CamDeclaration InputsDeclaration OutputsDeclaration PwmDeclaration
```

referenced by:

- Program

CamDeclaration:

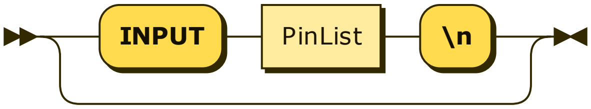


```
CamDeclaration
         ::= ( 'CAM' ( 'webcam' | 'picam' ) ':' 'id' '\n' )?
```

referenced by:

- Hardware

InputsDeclaration:

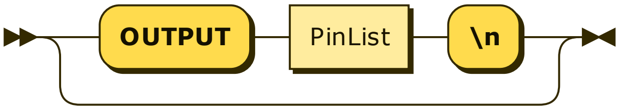


```
InputsDeclaration
         ::= ( 'INPUT' PinList '\n' )?
```

referenced by:

- Hardware

OutputsDeclaration:

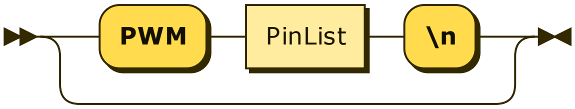


```
OutputsDeclaration
         ::= ( 'OUTPUT' PinList '\n' )?
```

referenced by:

- Hardware

PwmDeclaration:

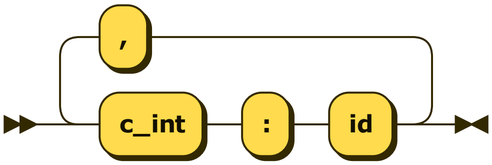


```
PwmDeclaration
         ::= ( 'PWM' PinList '\n' )?
```

referenced by:

- Hardware

PinList:

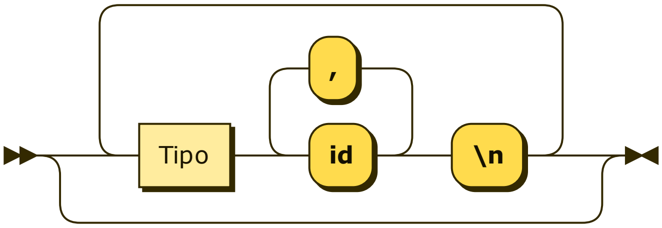


```
PinList  ::= 'c_int' ':' 'id' ( ',' 'c_int' ':' 'id' )*
```

referenced by:

- InputsDeclaration
- OutputsDeclaration
- PwmDeclaration

Vars:

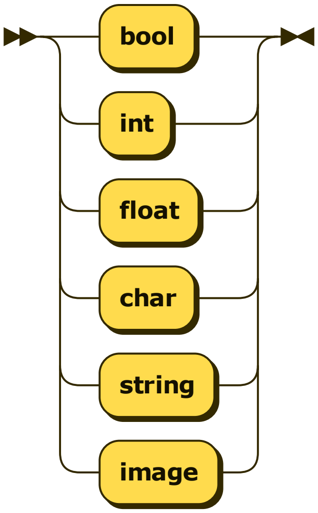


```
Vars     ::= ( Tipo 'id' ( ',' 'id' )* '\n' )*
```

referenced by:

- Program

Tipo:

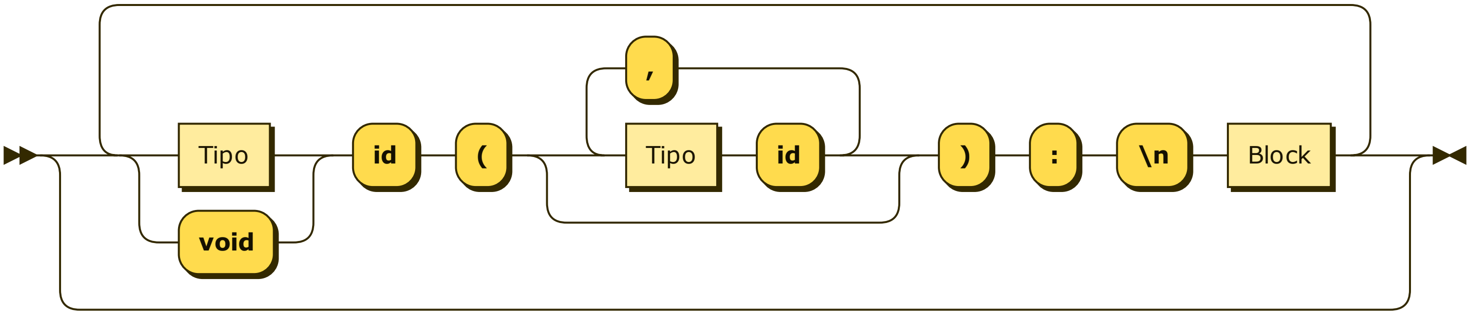


```
Tipo     ::= 'bool'
           | 'int'
           | 'float'
           | 'char'
           | 'string'
           | 'image'
```

referenced by:

- Functions
- Main
- Vars

Functions:

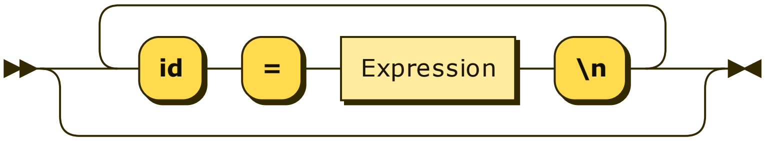


```
Functions
         ::= ( ( Tipo | 'void' ) 'id' '(' ( Tipo 'id' ( ',' Tipo 'id' )* )? ')' ':' '\n' Block )*
```

referenced by:

- Program

Assign:

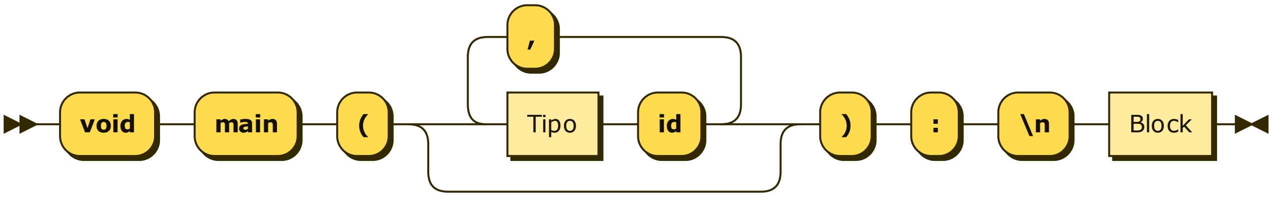


```
Assign   ::= ( 'id' '=' Expression '\n' )*
```

referenced by:

- Program
- Statement

Main:

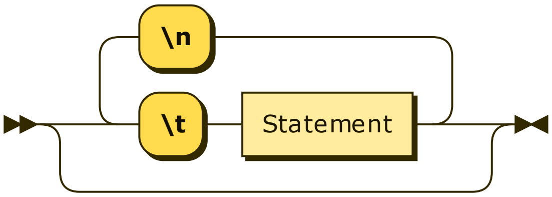


```
Main     ::= 'void' 'main' '(' ( Tipo 'id' ( ',' Tipo 'id' )* )? ')' ':' '\n' Block
```

referenced by:

- Program

Block:

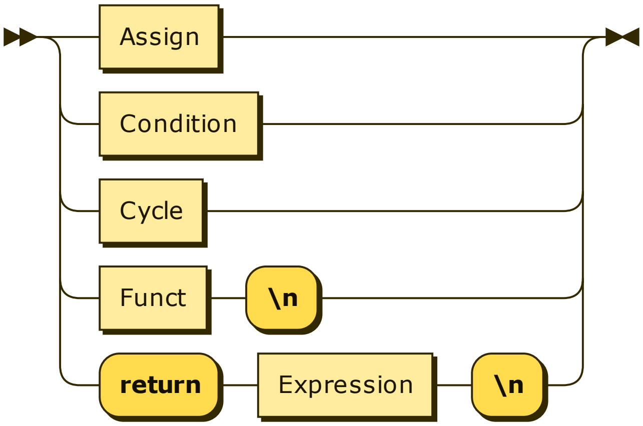


```
Block    ::= ( '\t' Statement ( '\n' '\t' Statement )* )?
```

referenced by:

- Condition
- Cycle
- Functions
- Main

Statement:

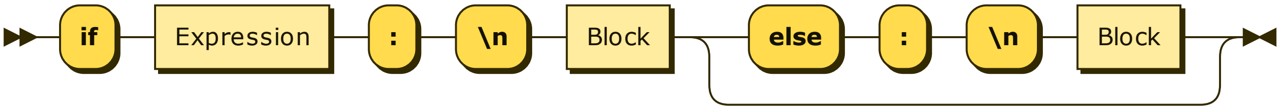


```
Statement
         ::= Assign
           | Condition
           | Cycle
           | Funct '\n'
           | 'return' Expression '\n'
```

referenced by:

- Block

Condition:

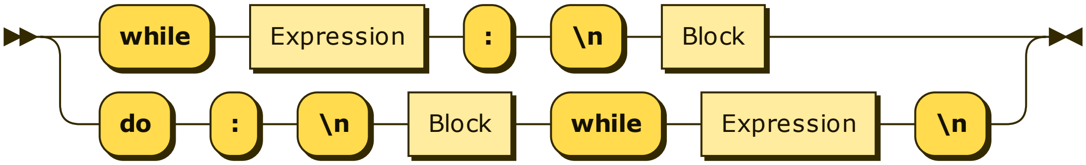


```
Condition
         ::= 'if' Expression ':' '\n' Block ( 'else' ':' '\n' Block )?
```

referenced by:

- Statement

Cycle:

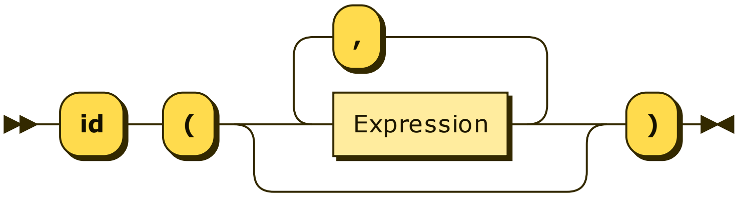


```
Cycle    ::= 'while' Expression ':' '\n' Block
           | 'do' ':' '\n' Block 'while' Expression '\n'
```

referenced by:

- Statement

Funct:

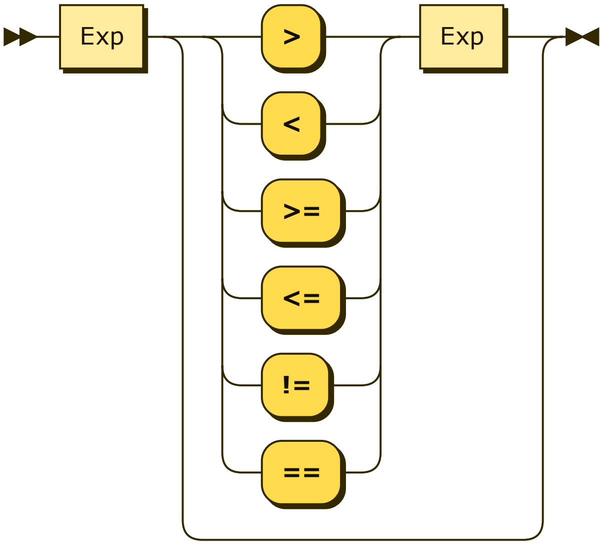


```
Funct    ::= 'id' '(' ( Expression ( ',' Expression )* )? ')'
```

referenced by:

- Factor
- Statement

Expression:

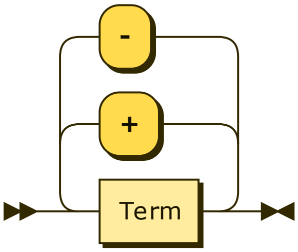


```
Expression
         ::= Exp ( ( '>' | '<' | '>=' | '<=' | '!=' | '==' ) Exp )?
```

referenced by:

- Assign
- Condition
- Cycle
- Factor
- Funct
- Statement

Exp:

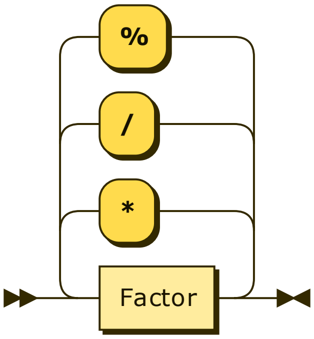


```
Exp      ::= Term ( ( '+' | '-' ) Term )*
```

referenced by:

- Expression

Term:

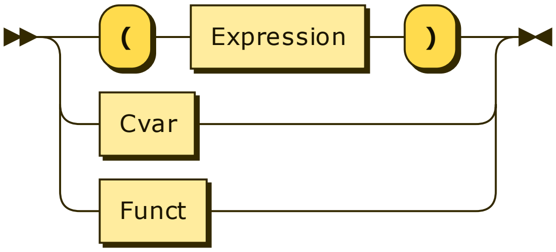


```
Term     ::= Factor ( ( '*' | '/' | '%' ) Factor )*
```

referenced by:

- Exp

Factor:

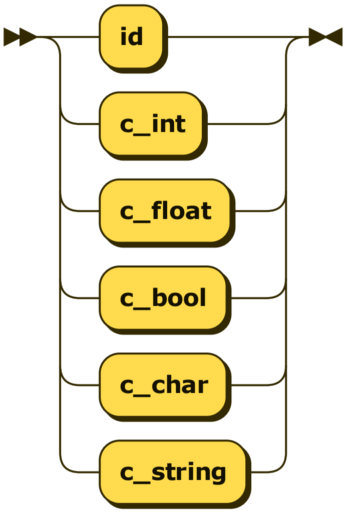


```
Factor   ::= '(' Expression ')'
           | Cvar
           | Funct
```

referenced by:

- Term

Cvar:



```
Cvar     ::= 'id'
           | 'c_int'
           | 'c_float'
           | 'c_bool'
           | 'c_char'
           | 'c_string'
```

referenced by:

- Factor

... generated by Railroad Diagram Generator