```
Run[[Program]] =
            Execute[[Block]]
            HALT

Execute[[declare Declarations do Statements od]] =
            JUMP s
            Elaborate[[Declarations]]
        s:  PUSH varsize                    (PUSH not generated if varsize = 0)
            Execute[[Statements]]

Elaborate[[Declaration*]] =
            Elaborate[[$Declaration_1$]]
            Elaborate[[$Declaration_2$]]
            …
            Elaborate[[$Declaration_n$]]

Elaborate[[var Identifier;]] =

Elaborate[[func Identifier( IdList ) Block return Expression;]]=
            Elaborate[[IdList]]
            Execute[[Block]]
            Evaluate[[Expression]]
            RETURN (1) paramsize

Elaborate[[IdList]] =

Execute[[Statement*]] =
            Execute[[$Statement_1$]]
            Execute[[$Statement_2$]]
            …
            Execute[[$Statement_n$]]

Execute[[Expression;]] =
            Evaluate[[Expression]]
            POP 1

Execute[[if Expression then $Statements_1$ else $Statements_2$ fi;]] =
            Evaluate[[Expression]]
            JUMPIF (0) e
            Execute[[Statements1]]
            JUMP d
       e:   Execute[[Statements2]]
       d:
```

```
Execute[[while Expression do Statements od;]] =
    r:  Evaluate[[Expression]]
        JUMPIF (0) d
        Execute[[Statements]]
        JUMP r
    d:

Execute[[say Expression;]] =
        Evaluate[[Expression]]
        CALL putint
        CALL puteol

Evaluate[[Identifier := Expression]] =
        Evaluate[[Expression]]
        STORE varoffset[varreg]
        LOAD varoffset[varreg]

Evaluate[[Expression$_1$ Operator Expression$_2$]] =
        Evaluate[[Expression$_1$]]
        Evaluate[[Expression$_2$]]
        CALL operator

Evaluate[[- Expression]] =
        Evaluate[[Expression]]
        CALL neg

Evaluate[[+ Expression]] =
        Evaluate[[Expression]]

Evaluate[[IntegerLiteral]] =
        LOADL literal

Evaluate[[Identifier]] =
        LOAD varoffset[varreg]

Evaluate[[Identifier( ExpressionList )]] =
        Evaluate[[ExpressionList]]
        CALL (funcreg) funcadr[CB]

Evaluate[[Expression ( , Expression )*]] =
        Evaluate[[Expression$_1$]]
        Evaluate[[Expression$_2$]]
        …
        Evaluate[[Expression$_n$]]
```