```
<integers> ::= <digit> <integers> | <digit>
<digit> ::= 0|1|2|3|4|5|6|7|8|9
<real> ::= <integers> . <integers>
```

```
<var_assgn> ::= var <var_name> = < i_expr>
<bool_var_assgn> :: = var <var_name> = <boolean>
```

Variable names can be anything except for keywords:

```
<var_name> ::= !<keywords>
<keywords> ::= var | add | mult | div | mod | true | false | or | not | less | than | equal |
greater | to | not | then | end | if | from | to | output | outputs | text | cla | array
```

```
<i_expr> ::= <i_expr> add <mult_expr> | <i_expr> sub <mult_expr> | <mult_expr>
<mult_expr> ::= <mult_expr> mult <neg_expr> | <mult_expr> div <neg_expr> |
<mult_expr> mod <neg_expr> | <neg_expr>
<neg_expr> ::= <root> | -<root>
<root> ::= (<i_expr>) | <var_name> = <var_name> | <real> | <integers> | <var_assgn>
```

```
<boolean> ::= true | false
```

```
<b_expr> ::= or(<b_expr>,<and_expr>) | and( <and_expr>,<b_root>) | <b_root> |
not(<b_root>) | <comp> | <var_name>
<b_root> ::= <bool> | (<b_expr>) | <var_name>
<comp> ::= <i_expr> less than <i_expr> |<i_expr> greater than <i_expr> |<i_expr> less
than or equal to <i_expr> |<i_expr> greater than or equal to <i_expr> |<i_expr> equal to
<i_expr> |<i_expr> not equal to <i_expr>
```

```
<conditionals> ::= if <comp> then <statements> end if | or if <comp> then <statements>
end if | or <statement> end if
```

```
<break> ::= stop loop now
```

```
<loops> ::= <from_loop> | <for_each>
<from_loop> ::= from <var_assgn> to <comp> increment by <integers> <statements>
end | from <var_assgn> to <comp> decrement by <integers> <statements> end
<for_each> ::= for each <var_name> in <statements> do <statements> end for each
```

```
<statements> ::= <statement> | <statement><statements> | <empty>
<statement> ::= <i_expr> | <conditionals> | <while_loop> | <from_loop> | <for_each>
```

```
<print_to_output> ::= output(<root>) | outputs(<root>)

<CLA> ::= cmd(<int>) | cmd(<int>) <CLA>

<str_literals> ::= “<chars>”
<chars> ::= <ASCII Chars>

<arrays> ::= var <var_name> = array(<integers> <empty> | <integers>, <integers>)

<comments> ::= start comment | end comment | *<chars>
```