```
ti- program ( <define> Cexecute> )
sprograms
cdefine>
                      ::= define | <variable=sel> | | #
<variable=sel>
                      ::= <not-variable> <variable> | <variable> | <not-variable> | <variable> <not-variable
::- execute ( <command-list> )
<execute>
snot-variable>
                      iz= not variable <not-variable+sel>
snot-variable-selp
                      ::= <type> <ie> <identifier-list-value> <not-variable-se
<variable>
                      ::= variable <variable-set>
<variable=sel>
                      ti= <type> <is> <idestifier-list> , <variable-se
                      ::= <set> (
*command-list>
                          sget .
                          <put>
                          <verify>
                          <Inop> .
                          cwhile:
                      ::- natural | real | char | boolean
<identifier-list>
                       :Xset <aspression> to <identifier-lint>
                          X<identifler=list> |
Kgeta
<put>
                            <identifier-list> |
                        se verify <expression> <false-true-sel>
                       t:= <is-true> <is-false> | <is-false> <is-true> | <is-false> //
r:= loop | <rommand-list> | while <expression> <is> true
«false-true-zel»
<loops
Swh1140
                       in while <expression> <la> true do | <command-list>
                       ::= _ (identifier=sel=u> | (letter) (identifier=sel>)
<identifier>
<!destifier-sel-up
                      t:= <letter> <identifier-sel> | <digit> <letter> <identifi
<identifier-sel>
                       :: <letter> <identifie:-sel> | <diqit> <letter> <identifie:
                                                                                         <identifier-sel-u> | ε
Kis-true>
                      ::= <ls> true ( <command-list> )
                      :: <is> false { <command-list> }
wis-falsox
                       :1- 18
<walues
                       : r= Chatural> | Creal> | Cohar> | Choolean>
sexpression?
                      :: <simple-expression> <expression-sel>
<expression-sel>
                      <pimple-expression>
                       ::= {term> <mimple-expression-sel>
<simple-expression-sel> ::= <additive-op> <tern> <simple-expression-sel> | :
«term»
                      :rw <factor> <term-sel>
sterm-sel-
                       :: coultiplicative-op> <factor> <term-sel> | e
<factor>
                       <iduntifier>
                          <sub-expression>
                          Congry>
<sub-expression>
                       it= ( <expression> )
                       it= + <unary-sel> | - <unary-sel> | 1 <unary-sel>
<unary-sel>
                       i:= Kexpression> Kunary-sel> | g
<relational-op>
                       110 < | > | == | != | <= | >=
<additive-op>
                       11=+ | - | |
                       110 * | ** | / | % | 3%
<multiplicative-op>
                       ::= <natural>
teral>
                          creal>
                           scharp.
                          shoolean>
<natural>
                       is= <digit>
steml:
                                    didita
                       ire edigits .
(digit)
                       1:-0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 9 | 9
                       1:= A | B | ... | Y | Z | A | b | ... | y | g
(char)
                       it= other-sel>
4char-sels
                       ti= <ascil> <char=sel> | £
<ascil>
                       11 = #0 | #1 | ... | #254 | #255
<boolean>
                       si= true | false
```

Vt = (program define not variable is natural real char boolean execute set gat to put loop while do true false () [] . + - * / ** % %% -- ! + < > <- 0- % () + |