

```

1
2  ANNULLABILI: statlistp , whenlistp , exprlistp
3
4  (prog)      ::= <statlist> EOF
5                statlist.next=newlabel() } <statlist> { emitlabel(statlist.next) }
6  (statlist)  ::= <stat> <statlistp>
7  (statlistp) ::= ; <stat> <statlistp> | ε
8
9  (stat)      ::= = ID <expr>
10                | print(<exprlist>)
11                | read(ID)
12                | cond<whenlist> else <stat>
13                | while(<bexpr>) <stat>
14                | { statlist }
15  (whenlist)  ::= <whenitem> <whenlistp>
16  (whenlistp) ::= <whenitem> <whenlistp> | ε
17  (whenitem)  ::= when(<bexpr>) do <stat>
18  (bexpr)     ::= RELOP <expr> <expr>
19  (expr)      ::= +(<exprlist>) | -<expr> <expr>
20                | *(<exprlist>) | /<expr> <expr>
21                | NUM | ID
22  (exprlist)  ::= <expr> <exprlistp>
23  (exprlistp) ::= <expr> <exprlistp> | ε
24  -----
25  FIRST:
26  ANNULLABILI: statlistp , whenlistp , exprlistp
27
28  FIRST(prog) = FIRST(statlist) = {print, read, cond, while, {, =}
29  FIRST(statlist) = FIRST(stat) (statlistp non conta perche stat non e
annulabile) = {print, read, cond, while, {, =}
30  FIRST(statlistp) = FIRST ; (stat) (statlistp) | ε = { ; }
31  FIRST(stat) = FIRST( = ID(expr))
32                (print(<exprlist>))
33                (read(ID))
34                (cond(whenlist)) (else(<stat>)) = (mi tengo solo cond per la
2condizione dell algoritmo )
35                (while(bexpr))
36                ({<statlist>}) = {print, read, cond, while, {, =}
37
38  FIRST(whenlist) = FIRST(whenitem) (whenlistp) = {when}
39  FIRST(whenlistp) = FIRST(whenitem) (whenlistp) | ε = {when}
40  FIRST(whenitem) = FIRST(when(bexpr)) (do(stat)) = {when}
41  FIRST(bexpr) = FIRST(RELOP(expr) (expr)) = {RELOP}
42  FIRST(expr) = FIRST(+(<exprlist>))
43                (*(<exprlist>))
44                (-(<expr>) (<expr>))
45                (/(<expr>) (<expr>))
46                (NUM) (ID)
= {NUM, ID, +, -, *, /}
47  FIRST(exprlist) = FIRST(expr) (exprlistp) = {NUM, ID, +, -, *, /}
48  FIRST(exprlistp) = FIRST(expr) (exprlistp) = {NUM, ID, +, -, *, /}
49  -----
50  FOLLOW:
51  FIRST(EOF) <= FOLLOW(STATLIST)
52
53  FOLLOW(STATLIST) <= FOLLOW(STAT)
54  FIRST(STATLISTP) <= FOLLOW(STAT)
55  FOLLOW(STATLIST) <= FOLLOW(STATLISTP)
56
57  FOLLOW(STATLISTP) <= FOLLOW(STAT)
58  FIRST(STATLISTP) <= FOLLOW(STAT)
59  // FOLLOW(STATLISTP) <= FOLLOW(STATLISTP)
60
61  FOLLOW(STAT) <= FOLLOW(EXPR)
62  FIRST(" ") <= FOLLOW(EXPRLIST)
63  FIRST("else") <= FOLLOW(WHENLIST)
64  FIRST(" ") <= FOLLOW(BEXPR)
65  FIRST("}") <= FOLLOW(STATLIST)
66
67  FOLLOW(WHENLIST) <= FOLLOW(WHENITEM)
68  FIRST(WHENLISTP) <= FOLLOW(WHENITEM)

```

```

69 FOLLOW(WHENLIST) <= FOLLOW(WHENLISTP)
70
71 FIRST(" ") <= FOLLOW(BEXPR)
72 FOLLOW(WHENITEM) <= (STAT)
73
74 FOLLOW(BEXPR) <= FOLLOW(EXPR)
75 // FIRST(EXPR) <= FOLLOW(EXPR)
76
77 FOLLOW(" ") <= FOLLOW(EXPRLIST)
78
79 FIRST(EXPRLISTP) <= FOLLOW(EXPR)
80 FOLLOW(EXPRLIST) <= FOLLOW(EXPR)
81 FOLLOW(EXPRLIST) <= FOLLOW(EXPRLISTP)
82
83 ANNULLABILI: statlistp , whenlistp , exprlistp
84 //-----
-----
85 TAB:          X:
86
87 PROG          $
88 STATLIST      (" " "EOF")
89 STATLISTP     (" " "EOF")
90 STAT          FOLLOW(STATLIST) FIRST(STATLISTP) FOLLOW(STATLISTP) FOLLOW(WHENITEM) =
91              (" " ";" "EOF" "ELSE" "WHEN")
92 WHENITEM      FOLLOW(WHENLIST) FIRST(WHENLISTP) =
93              ("ELSE" "WHEN" ";")
94 WHENLIST      ("ELSE" "WHEN")
95 WHENLISTP     ("ELSE")
96 BEXPR         (" ")
97 EXPR          FOLLOW(BEXPR) FOLLOW(STAT) = (" " " " ";" "EOF" "ELSE" "WHEN")
98 EXPRLIST      (" ")
99 EXPRLISTP     (" ")
100
101 GUIDA:
102 $FOLLOW(PROG)
103 PROG          = FIRST(STATLIST EOF)                                = {print, read, cond, while, {, =}
104 STATLIST      = FIRST(<STAT><STATLISTP>)                            = {print, read, cond, while, {, =}
105 STATLISTP     = FIRST(";" <STAT><STATLISTP>)                        = {";"}
106              FIRST(ε) U FOLLOW(STATLISTP)                          = {" " EOF } //
107 STAT          = FIRST("=" ID <EXPR>)                                = {"="}
108              FIRST(PRINT (<EXPRLIST>))                             = {"PRINT"}
109              FIRST(READ (ID))                                       = {"READ"}
110              FIRST(COND <WHENLIST> ELSE <STAT>)                     = {"COND"}
111              FIRST(WHILE (<BEXPR>) <STAT>)                           = {"WHILE"}
112              FIRST({<STATLIST>})                                    = {"{"}
113 WHENLIST      = FIRST(<WHENITEM><WHENLISTP>)                        = {"WHEN"}
114 WHENLISTP     = FIRST(<WHENITEM><WHENLISTP>)                        = {"WHEN"}
115              FIRST(ε) U FOLLOW(WHENLISTP)                          = {"ELSE"}
116 WHENITEM      = FIRST(when(<BEXPR>) do <STAT>)                     = {"WHEN"}
117 BEXPR         = FIRST(RELOP <EXPR><EXPR>)                          = {RELOP}
118 EXPR          = FIRST(+(<EXPRLIST>))                                = {"+"}
119              FIRST(-(<EXPRLIST>))                                    = {"-"}
120              FIRST(* <EXPR><EXPR>)                                    = {"*"}
121              FIRST(/ <EXPR><EXPR>)                                    = {"/"}
122              FIRST(NUM)                                             = {"NUM"}
123              FIRST(ID)                                              = {"ID"}
124 EXPRLIST      = FIRST(<EXPR><EXPRLISTP>)                            = {NUM, ID, +, -, *, /}
125 EXPRLISTP     = FIRST(<EXPR><EXPRLISTP>)                            = {NUM, ID, +, -, *, /}
126 EXPRLISTP     = FIRST(ε) U FOLLOW(EXPRLISTP)                      = {" " " "}
-----
-----
127 SDT OnTheFly
128
129 Prog          -> StatList      { emit(label) }
130              $
131
132 StatList      -> Stat
133              StatListP
134
135 StatListP     -> ;
136              Stat
137              StatListP

```

```

138
139 Stat -> =
140         ID
141         Expr          { emit(istore) }
142
143 Stat ->      print
144             (
145             ExperList
146             )          { emit(invokestatic) }
147
148 Stat-> read
149         (
150         ID
151         )          { emit(invokestatic) }
152                   { emit(store) }
153
154 Stat ->      cond
155             WhenList
156             else
157             Stat      { emit(label) }
158
159 Stat -> while          { emit(label) }
160         (
161         Bexpr
162         )          { emit(label) }
163         Stat      { emit(GOTO) }
164                   { emit(label) }
165
166 Stat ->      {
167             StatList
168             }
169
170 WhenList -> Whenitem
171             WhenListP
172
173 WhenListP-> Whenitem
174             WhenListP
175
176 WhenListP-> EPS
177
178 When ->      when
179             (
180             Bexpr      { emit(label) }
181             )
182             do
183
184 Bexpr ->      relop
185             Expr
186             Expr      { emit(relop) }
187                       { emit(GOTO) }
188
189 Expr ->      +
190             (
191             EL
192             )          { emit(iadd) }
193
194 Expr ->      *
195             (
196             ExperList
197             )          { emit(imul) }
198
199 Expr ->      -
200             Expr
201             Expr      { emit(isub) }
202
203 Expr ->      /
204             Expr
205             Expr      { emit(idiv) }
206
207 Expr ->      NUM      { emit(ldc) }
208
209 Expr ->      ID      { emit(iloadd) }
210

```

```
211 ExprList -> Expr
212         ExprListP
213
214 ExprListP -> Expr
215         ExprListP      { emit(iadd) } / { emit(imul) }
216
217 ExprListP -> EPS
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
```