On factorise la grammaire et on numérote les règles

$r_1 :< fichier > \rightarrow$ with Ada.Text_IO; use Ada.Text_IO; procedure $< ident >$ is $< fichier >_2$

$r_2 :< fichier >_2 \rightarrow$ begin $< instr >^+$ end $< fichier >_3$

$r_3 : \rightarrow < decl >^+$ begin $< instr >^+$ end $< fichier >_3$

$r_4 :< fichier >_3 \rightarrow$; EOF

$r_5 : \rightarrow < ident >$; EOF

$r_6 :< decl > \rightarrow$ type $< decl >_{11}$

$r_7 : \rightarrow$ procedure $< ident > < decl >_{21}$

$r_8 : \rightarrow$ function $< ident > < decl >_{31}$

$r_9 :< decl >_{11} \rightarrow < ident > < decl >_{12}$

$r_{10} : \rightarrow < ident >_{,}^+ : < type > < decl >_{13}$

$r_{11} :< decl >_{12} \rightarrow$;

$r_{12} : \rightarrow$ is $< decl >_{14}$

$r_{13} :< decl >_{13} \rightarrow$;

$r_{14} : \rightarrow (:=< expr >)$;

$r_{15} :< decl >_{14} \rightarrow$ access $< ident >$;

$r_{16} : \rightarrow$ record $< champs >^+$ end record;

$r_{17} :< decl >_{21} \rightarrow$ is $< decl >_{22}$

$r_{18} : \rightarrow < param >$ is $< decl >_{22}$

$r_{19} :< decl >_{22} \rightarrow$ begin $< instr >^+$ end $< decl >_{23}$

$r_{20} : \rightarrow < decl >^+$ begin $< instr >^+$ end $< decl >_{23}$

$r_{21} :< decl >_{23} \rightarrow$;

$r_{22} : \rightarrow < ident >$;

$r_{23} :< decl >_{31} \rightarrow$ return $< type >$ is $< decl >_{22}$

$r_{24} : \rightarrow < param >$ return $< type >$ is $< decl >_{22}$

$r_{25} :< champs > \rightarrow < ident >_{,}^+ : < type >$;

$r_{26} :< type > \rightarrow < ident >$

$r_{27} : \rightarrow$ access $< ident >$

$r_{28} :< params > \rightarrow (< param >_{;}^+)$

$r_{29} :< param > \rightarrow < ident >_{,}^+ : < param >_2$

$r_{30} :< param >_2 \rightarrow < type >$

$r_{31} : \rightarrow < mode > < type >$

$r_{32} :< mode > \rightarrow$ in $< mode >_1$

$r_{33} :< mode >_1 \rightarrow$ out

$r_{34} : \rightarrow \wedge$

r_{35} :$< expr >$ $\rightarrow < entier > < expr >_{recur}$

r_{36} : $\rightarrow < caractère > < expr >_{recur}$

r_{37} : $\rightarrow$ true $< expr >_{recur}$

r_{38} : $\rightarrow$ false $< expr >_{recur}$

r_{39} : $\rightarrow$ null $< expr >_{recur}$

r_{40} : $\rightarrow (< expr >) < expr >_{recur}$

r_{41} : $\rightarrow < acces > < expr >_{recur}$

r_{42} : $\rightarrow$ not $< expr > < expr >_{recur}$

r_{43} : $\rightarrow$ - $< expr > < expr >_{recur}$

r_{44} : $\rightarrow$ new $< ident > < expr >_{recur}$

r_{45} : $\rightarrow < ident > (< expr >^{+}_{,}) < expr >_{recur}$

r_{46} : $\rightarrow$ character ' val $(< expr >) < expr >_{recur}$

r_{47} :$< expr >_{recur}$ $\rightarrow < operateur > < expr > < expr >_{recur}$

r_{48} : $\rightarrow \wedge$

r_{49} :$< instr >$ $\rightarrow < acces > := < expr >;$

r_{50} : $\rightarrow < ident > < instr >_1$

r_{51} : $\rightarrow$ return $< instr >_2$

r_{52} : $\rightarrow$ begin $< instr >^{+}$ end;

r_{53} : $\rightarrow$ if $< expr >$ then $< instr >^{+} < instr >_3$

r_{54} : $\rightarrow$ for $< ident >$ in reverse? $< expr >$ .. $< expr >$ loop $< instr >^{+}$ end loop;

r_{55} : $\rightarrow$ while $< expr >$ loop $< instr >^{+}$ end loop;

r_{56} :$< instr_? >$ $\rightarrow$;

r_{57} : $\rightarrow (< expr >^{+}_{,});$

r_{58} :$< instr_? >$ $\rightarrow$;

r_{59} : $\rightarrow < expr >;$

r_{60} :$< instr_? >$ $\rightarrow$ end if;

r_{61} : $\rightarrow$ (else $< instr >^{+}$) end if;

r_{62} : $\rightarrow < elsif >^{+} < instr_? >$

r_{63} :$< instr_? >$ $\rightarrow$ end if;

r_{64} : $\rightarrow$ (else $< instr >^{+}$) end if;

r_{65} :$< opérateur >$ $\rightarrow =$

r_{66} : $\rightarrow /=$

r_{67} : $\rightarrow <$

r_{68} : $\rightarrow <=$

r_{69} : $\rightarrow >$

r_{70} : $\rightarrow >=$

r_{71} : $\rightarrow +$

r_{72} : $\rightarrow$ -

r_{73} : $\rightarrow$ *

r_{74} : $\rightarrow /$

r_{75} : $\rightarrow$ rem

r_{76} :$< acces >$ $\rightarrow < ident > < acces >_{recur}$

r_{77} : $\rightarrow < entier > < expr >_{recur} . < ident > < acces >_{recur}$

r_{78} : $\rightarrow < caractère > < expr >_{recur} . < ident > < acces >_{recur}$

r_{79} : $\rightarrow$ true $< expr >_{recur} . < ident > < acces >_{recur}$

r_{80} : $\rightarrow$ false $< expr >_{recur} . < ident > < acces >_{recur}$

r_{81} : $\rightarrow$ null $< expr >_{recur} . < ident > < acces >_{recur}$

r_{82} : $\rightarrow (< expr >) < expr >_{recur} . < ident > < acces >_{recur}$

r_{83} : $\rightarrow$ not $< expr > < expr >_{recur} . < ident > < acces >_{recur}$

r_{84} : $\rightarrow$ - $< expr > < expr >_{recur} . < ident > < acces >_{recur}$

r_{85} : $\rightarrow$ new $< ident > < expr >_{recur} . < ident > < acces >_{recur}$

r_{86} : $\rightarrow < ident > (< expr >^{+}_{,}) < expr >_{recur} . < ident > < acces >_{recur}$

r_{87} : $\rightarrow$ character ' val $(< expr >) < expr >_{recur} . < ident > < acces >_{recur}$

r_{88} :$< acces >_{recur}$ $\rightarrow$ $. < ident > < acces >_{recur}$

r_{89} : $\rightarrow \wedge$

r_{90} :$< instr >^{+}$ $\rightarrow < instr > < instr >^{+}_{1}$

r_{91} :$< instr >^{+}_{1}$ $\rightarrow < instr >^{+}$

r_{92} : $\rightarrow \wedge$

r_{93} :$< decl >^{+}$ $\rightarrow < decl > < decl >^{+}_{1}$

r_{94} :$< decl >^{+}_{1}$ $\rightarrow < decl >^{+}$

r_{95} : $\rightarrow \wedge$

r_{96} :$< champs >^{+}$ $\rightarrow < champs > < champs >^{+}_{1}$

r_{97} :$< champs >^{+}_{1}$ $\rightarrow < champs >^{+}$

r_{98} : $\rightarrow \wedge$

r_{99} :$< ident >^{+}_{,}$ $\rightarrow < ident > < ident >^{+}_{,1}$

r_{100} :$< ident >^{+}_{,1}$ $\rightarrow , < ident >^{+}_{,}$

r_{101} : $\rightarrow \wedge$

r_{102} :$< param >^{+}_{;}$ $\rightarrow < param > < param >^{+}_{;1}$

r_{103} :$< param >^{+}_{;1}$ $\rightarrow ; < param >^{+}_{;}$

r_{104} : $\rightarrow \wedge$

r_{105} :$< expr >^{+}_{,}$ $\rightarrow < expr > < expr >_1$ 2

r_{106} : $\rightarrow \wedge$

r_{107} :$< expr >_{12}$ $\rightarrow , < expr >^{+}_{,}$

r_{108} : $\rightarrow \wedge$

r_{109} :$< elsif >^{+}$ $\rightarrow$ (elsif $< expr >$ then $< instr >^{+}) < elsif >^{+}_{1}$

r_{110} :$< elsif >^{+}_{1}$ $\rightarrow < elsif >^{+}$

r_{111} : $\rightarrow \wedge$

$$P_{\wedge}(G) = \{< mode >_1, < expr >_{recur}, < acces >_{recur}, < instr >^{+}_{1}, < decl >^{+}_{1}, < champs >^{+}_{1}, < ident >^{+}_{,1}, < param >^{+}_{;1}, < expr >^{+}_{,}, < expr >_{12}, < elif >^{+}_{1}\}$$