package practicapl;

import java\_cup.runtime.\*;

action code

{:

int tab=1;

:}

terminal String identifier;

terminal program, begin, end, punto, punto\_coma;

terminal integer, real, char\_, const\_, op\_igual, var, dos\_puntos, coma;

terminal String numeric\_integer\_const;

terminal String numeric\_integerHex\_const;

terminal String numeric\_real\_const;

terminal String numeric\_realHex\_const;

terminal String string\_const;

terminal procedure, function, abrir\_paren, cerrar\_paren, abrir\_corchete, cerrar\_corchete;

terminal op\_asignacion, op\_menor, op\_mayor, op\_menorIgual, op\_mayorIgual, op\_distinto;

terminal op\_suma, op\_resta, op\_mult, op\_divEntero, op\_mod, op\_or, op\_and, op\_not, op\_divReal;

terminal if\_, then, else\_, while\_, do\_, for\_, to, case\_, of, type, array, record;

non terminal Simbolo PRG;

non terminal Simbolo BLQ;

non terminal Simbolo DCLLIST;

non terminal Simbolo DCL;

non terminal Simbolo SENTLIST;

non terminal Simbolo SENT;

non terminal Simbolo ALLTYPES;

non terminal Simbolo DEFCTE;

non terminal Simbolo CTELIST;

non terminal Simbolo SIMPVALUE;

non terminal Simbolo DEFTYPE;

non terminal Simbolo TYPELIST;

non terminal Simbolo UDTYPE;

non terminal Simbolo DEFVAR;

non terminal Simbolo DEFVARLIST;

non terminal Simbolo VARLIST;

non terminal Simbolo TBAS;

non terminal Simbolo DEFPROC;

non terminal Simbolo FORMAL\_PARAMLIST;

non terminal Simbolo FORMAL\_PARAM;

non terminal Simbolo DEFFUN;

non terminal Simbolo ASIG;

non terminal Simbolo ID;

non terminal Simbolo EXP;

non terminal Simbolo OP;

non terminal Simbolo OPCOMP;

non terminal Simbolo OPARIT;

non terminal Simbolo OPLOG;

non terminal Simbolo FACTOR;

non terminal Simbolo SUBPPARAMLIST;

non terminal Simbolo EXPLIST;

non terminal Simbolo PROC\_CALL;

non terminal Simbolo EXEBLQ;

non terminal Simbolo DCLLIST\_BLQ;

non terminal Simbolo COND;

non terminal Simbolo ELSECOND;

non terminal Simbolo WLOOP;

non terminal Simbolo FLOOP;

non terminal Simbolo CASE;

non terminal Simbolo CASELIST;

non terminal Simbolo DCL\_BLQ;

precedence left op\_menor;

precedence left op\_mayor;

precedence left op\_mayorIgual;

precedence left op\_menorIgual;

precedence left op\_igual;

precedence left op\_distinto;

precedence left op\_suma;

precedence left op\_resta;

precedence left op\_mod;

precedence left op\_mult;

precedence left op\_divEntero;

precedence left op\_divReal;

precedence left op\_not;

precedence left op\_or;

precedence left op\_and;

precedence left else\_;

precedence left if\_;

start with PRG;

PRG ::= program identifier:iden punto\_coma {: System.out.println("<!DOCTYPE html><HTML><head><title>EjNotable.pl</title><style>"

+".cte {color:rgb(19,189,72);}"

+".ident {color:rgb(55,40,244);}"

+".palres {color:red;font-weight:bold;}"

+"</style></head><BODY>"

+"<A NAME=\"inicio\"><H1>Programa:"

+iden

+";</H1><H2>Funciones y procedimientos</H2><UL>"); :}BLQ:blq punto

{:

Simbolo prg = new Simbolo();

prg.valor1= (blq.valor1+"</BODY></HTML>");

System.out.println(prg.valor1);

:};

BLQ ::= DCLLIST:dcllist begin SENTLIST:sentlist end

{:

Simbolo blq = new Simbolo();

//En valor1 esta los subprogrmas y en valor2 los variables

blq.valor1 = "<LI><A HREF=\"#ProgPpal\">Programa Principal</A><BR/></UL>"

+ dcllist.valor1

+"<HR><A NAME=\"ProgPpal\"><H2>Programa Principal</H2>"

+ dcllist.valor2

+"<SPAN CLASS=\"palres\">begin</SPAN>"

+"<DIV style=\"text-indent: "+tab+"cm\">"+sentlist.valor1+"</DIV>"

+"<SPAN CLASS=\"palres\">end</SPAN>.<BR/>"

+"<A HREF=\"#ProgPpal\">Inicio del programa principal</A><BR/>"

+"<A HREF=\"#inicio\">Inicio de programa</A><BR/>";

RESULT=blq;

:};

BLQ ::= begin SENTLIST:sentlist end

{:

Simbolo blq = new Simbolo();

//En valor1 esta los subprogrmas y en valor2 los variables

blq.valor1 = "<A NAME=\"ProgPpal\"><H2>Programa Principal</H2>"

+"<SPAN CLASS=\"palres\">begin</SPAN>"

+"<DIV style=\"text-indent: "+tab+"cm\">"+sentlist.valor1+"</DIV>"

+"<SPAN CLASS=\"palres\">end</SPAN>.<BR/>"

+"<A HREF=\"#ProgPpal\">Inicio del programa principal</A><BR/>"

+"<A HREF=\"#inicio\">Inicio de programa</A><BR/>";

RESULT=blq;

:};

DCLLIST ::= DCLLIST:dcllist1 DCL:dcl

{:

Simbolo dcllist = new Simbolo();

dcllist.valor1=dcllist1.valor1

+ dcl.valor1;

dcllist.valor2=dcllist1.valor2

+ dcl.valor2;

RESULT=dcllist;

:};

DCLLIST ::= DCL:dcl

{:

Simbolo dcllist = new Simbolo();

dcllist.valor1=dcl.valor1;

dcllist.valor2=dcl.valor2;

RESULT=dcllist;

:};

SENTLIST ::= SENTLIST:sentlist1 SENT:sent

{:

Simbolo sentlist = new Simbolo();

sentlist.valor1=sentlist1.valor1+sent.valor1;

RESULT=sentlist;

:};

SENTLIST ::= SENT:sent

{:

Simbolo sentlist = new Simbolo();

sentlist.valor1=sent.valor1;

RESULT=sentlist;

:};

DCL ::={:tab=1;:} DEFCTE:defcte

{:

Simbolo dcl = new Simbolo();

dcl.valor1 = "";

dcl.valor2 = defcte.valor2;

RESULT= dcl;

:};

DCL ::={:tab=1;:} DEFTYPE:deftype

{:

Simbolo dcl = new Simbolo();

dcl.valor1 = "";

dcl.valor2 = deftype.valor2;

RESULT= dcl;

:};

DCL ::={:tab=1;:} DEFVAR:defvar

{:

Simbolo dcl = new Simbolo();

dcl.valor1 = "";

dcl.valor2 = defvar.valor2;

RESULT= dcl;

:};

DCL ::= DEFPROC:defproc

{:

Simbolo dcl = new Simbolo();

dcl.valor1 = defproc.valor1;

dcl.valor2 = "";

RESULT= dcl;

:};

DCL ::= DEFFUN:deffun

{:

Simbolo dcl = new Simbolo();

dcl.valor1 = deffun.valor1;

dcl.valor2 = "";

RESULT= dcl;

:};

DEFCTE ::= const\_ CTELIST:ctelist

{:

Simbolo defcte = new Simbolo();

defcte.valor1 = "";

defcte.valor2="<SPAN CLASS=\"palres\">CONST &nbsp;</SPAN>" + "<DIV style=\"text-indent: "+tab+"cm\">" + ctelist.valor1 + "</DIV>";

RESULT=defcte;

:};

CTELIST ::= identifier:iden op\_igual SIMPVALUE:simpvalue punto\_coma

{:

Simbolo ctelist = new Simbolo();

ctelist.valor1 = "<A NAME=\""+iden+"\"><SPAN CLASS=\"ident\">"+iden+"</SPAN></A>="+simpvalue.valor1+";<BR>";

RESULT= ctelist;

:};

CTELIST ::= CTELIST:ctelist1 identifier:iden op\_igual SIMPVALUE:simpvalue punto\_coma

{:

Simbolo ctelist = new Simbolo();

ctelist.valor1 = ctelist1.valor1

+ "<A NAME=\""+iden+"\"><SPAN CLASS=\"ident\">"+iden+"</SPAN></A>="+simpvalue.valor1+";<BR>";

RESULT= ctelist;

:};

DEFTYPE ::= type TYPELIST:typelist

{:

Simbolo deftype = new Simbolo();

deftype.valor1 = "";

deftype.valor2="<SPAN CLASS=\"palres\">TYPE &nbsp;</SPAN>" + "<DIV style=\"text-indent:"+tab+"cm\">" + typelist.valor1 + "</DIV><BR>";

RESULT= deftype;

:};

TYPELIST ::= identifier:iden op\_igual UDTYPE:udtype punto\_coma

{:

Simbolo typelist = new Simbolo();

typelist.valor1 = "</A><A NAME=\""+iden+"\"><SPAN CLASS=\"ident\">"+iden+"</SPAN></A>="+udtype.valor1+";";

RESULT= typelist;

:};

TYPELIST ::= identifier:iden op\_igual UDTYPE:udtype punto\_coma TYPELIST:typelist1

{:

Simbolo typelist = new Simbolo();

typelist.valor1 = "</A><A NAME=\""+iden+"\"><SPAN CLASS=\"ident\">"+iden+"</SPAN></A>="+udtype.valor1+";" + typelist1.valor1;

RESULT= typelist;

:};

UDTYPE ::= array abrir\_corchete SIMPVALUE:simpvalue1 punto punto SIMPVALUE:simpvalue2 cerrar\_corchete of ALLTYPES:alltypes

{:

Simbolo udtype = new Simbolo();

udtype.valor1 = "<SPAN CLASS=\"palres\">array</SPAN>[" + simpvalue1.valor1 + ".." + simpvalue2.valor1 + "] of " + alltypes.valor1;

RESULT= udtype;

:};

UDTYPE ::= record DEFVARLIST:defvarlist punto\_coma end

{:

Simbolo udtype = new Simbolo();

udtype.valor1 = "<SPAN CLASS=\"palres\">RECORD</SPAN>" + defvarlist.valor1 + "; end";

RESULT= udtype;

:};

SIMPVALUE ::= numeric\_integer\_const:nic

{:

Simbolo simpvalue = new Simbolo();

simpvalue.valor1="<SPAN CLASS=\"cte\">"+nic+"</SPAN>";

RESULT=simpvalue;

:} ;

SIMPVALUE ::= numeric\_integerHex\_const:nihc

{:

Simbolo simpvalue = new Simbolo();

simpvalue.valor1="<SPAN CLASS=\"cte\">"+nihc+"</SPAN>";

RESULT=simpvalue;

:};

SIMPVALUE ::= numeric\_real\_const:nrc

{:

Simbolo simpvalue = new Simbolo();

simpvalue.valor1="<SPAN CLASS=\"cte\">"+nrc+"</SPAN>";

RESULT=simpvalue;

:};

SIMPVALUE ::= numeric\_realHex\_const:nrhc

{:

Simbolo simpvalue = new Simbolo();

simpvalue.valor1="<SPAN CLASS=\"cte\">"+nrhc+"</SPAN>";

RESULT=simpvalue;

:};

SIMPVALUE ::= string\_const:sc

{:

Simbolo simpvalue = new Simbolo();

simpvalue.valor1="<SPAN CLASS=\"cte\">"+sc+"</SPAN>";

RESULT=simpvalue;

:};

DEFVAR ::= var DEFVARLIST:defvarlist punto\_coma

{:

Simbolo defvar = new Simbolo();

defvar.valor1= "";

defvar.valor2="<SPAN CLASS=\"palres\">VAR &nbsp;</SPAN>" + defvarlist.valor1;

RESULT=defvar;

:};

DEFVARLIST ::= VARLIST:varlist dos\_puntos ALLTYPES:alltypes

{:

Simbolo defvarlist = new Simbolo();

defvarlist.valor1="<DIV style=\"text-indent:"+tab+"cm\">"+varlist.valor1+":"+alltypes.valor1+";</DIV>";

RESULT=defvarlist;

:};

DEFVARLIST ::= DEFVARLIST:defvarlist1 punto\_coma VARLIST:varlist dos\_puntos ALLTYPES:alltypes

{:

Simbolo defvarlist = new Simbolo();

defvarlist.valor1=defvarlist1.valor1

+ "<DIV style=\"text-indent:"+tab+"cm\">" + varlist.valor1+":"+alltypes.valor1+";</DIV>";

RESULT=defvarlist;

:};

VARLIST ::= identifier:iden

{:

Simbolo varlist = new Simbolo();

varlist.valor1="<A NAME=\""+iden+"\"><SPAN CLASS=\"ident\">"+iden+"</SPAN></A>";

RESULT=varlist;

:};

VARLIST ::= identifier:iden coma VARLIST:varlist1

{:

Simbolo varlist = new Simbolo();

varlist.valor1="<A NAME=\""+iden+"\"><SPAN CLASS=\"ident\">"+iden+"</SPAN></A>, " + varlist1.valor1;

RESULT=varlist;

:};

DEFPROC ::= procedure identifier:iden FORMAL\_PARAMLIST:formalparamlist {:tab=0;:} punto\_coma {: System.out.println("<LI><A HREF=\"#"+iden+"\">PROCEDURE "+iden+"</A>"+formalparamlist.valor1+";</LI>"); :} EXEBLQ:exeblq punto\_coma

{:

Simbolo defproc = new Simbolo();

defproc.valor1="<HR><A NAME=\""+iden+"\"><SPAN CLASS=\"palres\">PROCEDURE</SPAN><SPAN CLASS=\"ident\">"+iden+"</SPAN>"+formalparamlist.valor1+";"+exeblq.valor1+"</A><BR>"

+"<A HREF=\"#"+iden+"\">Inicio de la rutina</A><BR/>"

+"<A HREF=\"#inicio\">Inicio de programa</A><BR/>";

RESULT=defproc;

:};

DEFPROC ::= procedure identifier:iden {:tab=0;:} punto\_coma {: System.out.println("<LI><A HREF=\"#"+iden+"\">PROCEDURE "+iden+";</A></LI>"); :}EXEBLQ:exeblq punto\_coma

{:

Simbolo defproc = new Simbolo();

defproc.valor1="<HR><A NAME=\""+iden+"\"><SPAN CLASS=\"palres\">PROCEDURE</SPAN><SPAN CLASS=\"ident\">"+iden+"</SPAN></A>;"+exeblq.valor1+"<BR>"

+"<A HREF=\"#"+iden+"\">Inicio de la rutina</A><BR/>"

+"<A HREF=\"#inicio\">Inicio de programa</A><BR/>";;

RESULT=defproc;

:};

DEFFUN ::=function identifier:iden FORMAL\_PARAMLIST:formalparamlist dos\_puntos ALLTYPES:alltypes {:tab=0;:} punto\_coma {: System.out.println("<LI><A HREF=\"#"+iden+"\">FUNCTION "+iden+"</A>"+formalparamlist.valor1+":"+alltypes.valor1+";</LI>"); :} EXEBLQ:exeblq punto\_coma

{:

Simbolo deffun = new Simbolo();

deffun.valor1="<HR><A NAME=\""+iden+"\"></A><SPAN CLASS=\"palres\">FUNCTION</SPAN><SPAN CLASS=\"ident\">"+iden+"</SPAN>"+formalparamlist.valor1+":"+alltypes.valor1+";"+exeblq.valor1+"<BR>"

+"<A HREF=\"#"+iden+"\">Inicio de la rutina</A><BR/>"

+"<A HREF=\"#inicio\">Inicio de programa</A><BR/>";

RESULT=deffun;

:};

DEFFUN ::= function identifier:iden dos\_puntos ALLTYPES:alltypes {:tab=0;:} punto\_coma {: System.out.println("<LI><A HREF=\"#"+iden+"\">FUNCTION "+iden+"():"+alltypes.valor1+";</A></LI>"); :} EXEBLQ:exeblq punto\_coma

{:

Simbolo deffun = new Simbolo();

deffun.valor1="<HR><A NAME=\""+iden+"\"></A><SPAN CLASS=\"palres\">FUNCTION</SPAN><SPAN CLASS=\"ident\">"+iden+"</SPAN>:"+alltypes.valor1+";"+exeblq.valor1+"<BR>"

+"<A HREF=\"#"+iden+"\">Inicio de la rutina</A><BR/>"

+"<A HREF=\"#inicio\">Inicio de programa</A><BR/>";;

RESULT=deffun;

:};

FORMAL\_PARAMLIST ::= abrir\_paren FORMAL\_PARAM:formalparam cerrar\_paren

{:

Simbolo formalparamlist = new Simbolo();

formalparamlist.valor1="("+formalparam.valor1+")";

RESULT=formalparamlist;

:};

FORMAL\_PARAM ::= VARLIST:varlist dos\_puntos ALLTYPES:alltypes

{:

Simbolo formalparam = new Simbolo();

formalparam.valor1=varlist.valor1+":"+alltypes.valor1;

RESULT=formalparam;

:};

FORMAL\_PARAM ::= VARLIST:varlist dos\_puntos ALLTYPES:alltypes punto\_coma FORMAL\_PARAM:formalparam1

{:

Simbolo formalparam = new Simbolo();

formalparam.valor1=varlist.valor1+":"+alltypes.valor1+";"+formalparam1.valor1;

RESULT=formalparam;

:};

TBAS ::= integer

{:

Simbolo tbas = new Simbolo();

tbas.valor1="<SPAN CLASS=\"palres\">INTEGER</SPAN>";

RESULT=tbas;

:};

TBAS ::= real:r

{:

Simbolo tbas = new Simbolo();

tbas.valor1="<SPAN CLASS=\"palres\">REAL</SPAN>";

RESULT=tbas;

:};

TBAS ::= char\_:c

{:

Simbolo tbas = new Simbolo();

tbas.valor1="<SPAN CLASS=\"palres\">CHAR</SPAN>";

RESULT=tbas;

:};

ALLTYPES ::= TBAS:tbas

{:

Simbolo alltypes = new Simbolo();

alltypes.valor1=tbas.valor1;

RESULT=alltypes;

:};

ALLTYPES ::= identifier:iden

{:

Simbolo alltypes = new Simbolo();

alltypes.valor1="<A HREF=\"#"+iden+"\"></A><SPAN CLASS=\"ident\">"+iden+"</SPAN>";

RESULT=alltypes;

:};

SENT ::= ASIG:asig punto\_coma

{:

Simbolo sent = new Simbolo();

sent.valor1="<DIV style=\"text-indent:"+tab+"cm\">"+asig.valor1+";</DIV>";

RESULT=sent;

:};

SENT ::= PROC\_CALL:proc punto\_coma

{:

Simbolo sent = new Simbolo();

sent.valor1="<DIV style=\"text-indent:"+tab+"cm\">"+proc.valor1+";</DIV>";

RESULT=sent;

:};

SENT ::= EXEBLQ:exeblq

{:

Simbolo sent = new Simbolo();

sent.valor1="<DIV style=\"text-indent:"+tab+"cm\">"+exeblq.valor1+"</DIV>";

RESULT=sent;

:};

SENT ::= COND:cond

{:

Simbolo sent = new Simbolo();

sent.valor1="<DIV style=\"text-indent:"+tab+"cm\">"+cond.valor1+"</DIV>";

RESULT=sent;

:};

SENT ::= WLOOP:wloop

{:

Simbolo sent = new Simbolo();

sent.valor1="<DIV style=\"text-indent:"+tab+"cm\">"+wloop.valor1+"</DIV>";

RESULT=sent;

:};

SENT ::= FLOOP:floop

{:

Simbolo sent = new Simbolo();

sent.valor1="<DIV style=\"text-indent: "+tab+"cm\">"+floop.valor1+"</DIV>";

RESULT=sent;

:};

SENT ::= CASE:case\_

{:

Simbolo sent = new Simbolo();

sent.valor1="<DIV style=\"text-indent:"+tab+"cm\">"+case\_.valor1+"</DIV>";

RESULT=sent;

:};

COND ::= if\_ EXP:exp then SENT:sent punto\_coma

{:

Simbolo cond = new Simbolo();

cond.valor1="<SPAN CLASS=\"palres\">IF</SPAN>"+exp.valor1+"<SPAN CLASS=\"palres\">THEN</SPAN>"+sent.valor1;

RESULT=cond;

:};

COND ::= if\_ EXP:exp then SENT:sent punto\_coma ELSECOND:elsecond punto\_coma

{:

Simbolo cond = new Simbolo();

cond.valor1="<SPAN CLASS=\"palres\">IF</SPAN>"+exp.valor1+" <SPAN CLASS=\"palres\">THEN</SPAN> "+sent.valor1+elsecond.valor1;

RESULT=cond;

:};

ELSECOND ::= else\_ SENT:sent

{:

Simbolo elsecond = new Simbolo();

elsecond.valor1="<SPAN CLASS=\"palres\">ELSE</SPAN>"+sent.valor1;

RESULT=elsecond;

:};

WLOOP ::= while\_ EXP:exp do\_ SENT:sent punto\_coma

{:

Simbolo wloop = new Simbolo();

wloop.valor1="<SPAN CLASS=\"palres\">WHILE</SPAN>"+exp.valor1+"<SPAN CLASS=\"palres\">DO</SPAN>"+sent.valor1;

RESULT=wloop;

:};

FLOOP ::= for\_ identifier:iden op\_asignacion EXP:exp1 to EXP:exp2 do\_ SENT:sent punto\_coma

{:

Simbolo floop = new Simbolo();

floop.valor1="<SPAN CLASS=\"palres\">FOR</SPAN> <SPAN CLASS=\"ident\">"+iden+"</SPAN> := "+exp1.valor1+"<SPAN CLASS=\"palres\">TO</SPAN>"+exp2.valor1+"<SPAN CLASS=\"palres\">DO</SPAN>"+sent.valor1;

RESULT=floop;

:};

CASE ::= case\_ EXP:exp of CASELIST:caselist end punto\_coma

{:

Simbolo case\_ = new Simbolo();

case\_.valor1="<SPAN CLASS=\"palres\">CASE</SPAN>"+exp.valor1+"<SPAN CLASS=\"palres\">OF</SPAN><BR>"+caselist.valor1+"<DIV style=\"text-indent: "+tab+"cm\"><SPAN CLASS=\"palres\">END</SPAN>;</DIV>";

RESULT=case\_;

:};

CASELIST ::= EXP:exp dos\_puntos SENT:sent

{:

Simbolo caselist = new Simbolo();

caselist.valor1="<DIV style=\"text-indent:"+(tab+1)+"cm\">"+exp.valor1+":</DIV> "+sent.valor1;

RESULT=caselist;

:};

CASELIST ::= EXP:exp dos\_puntos SENT:sent CASELIST:caselist1

{:

Simbolo caselist = new Simbolo();

caselist.valor1="<DIV style=\"text-indent: "+(tab+1)+"cm\">"+exp.valor1+": </DIV>"+sent.valor1+caselist1.valor1;

RESULT=caselist;

:};

ASIG ::= ID:id op\_asignacion EXP:exp

{:

Simbolo asig = new Simbolo();

asig.valor1=id.valor1+" := "+exp.valor1+"</SPAN>";

RESULT=asig;

:};

ID ::= identifier:iden

{:

Simbolo id = new Simbolo();

id.valor1="<A HREF=\"#"+iden+"\">"+iden+"</A>";

RESULT=id;

:};

ID ::= identifier:iden abrir\_corchete EXP:exp cerrar\_corchete

{:

Simbolo id = new Simbolo();

id.valor1="<A HREF=\"#"+iden+"\">"+iden+"</A>[<SPAN CLASS=\"cte\">"+exp.valor1+"</SPAN>]";

RESULT=id;

:};

ID ::= identifier:iden1 punto identifier:iden2

{:

Simbolo id = new Simbolo();

id.valor1="<A HREF=\"#"+iden1+"\">"+iden1+"</A>.<SPAN CLASS=\"ident\">"+iden2+"</SPAN>";

RESULT=id;

:};

EXP ::= EXP:exp1 OP:op EXP:exp2

{:

Simbolo exp = new Simbolo();

exp.valor1=exp1.valor1+" "+op.valor1+" "+exp2.valor1;

RESULT=exp;

:};

EXP ::= FACTOR:factor

{:

Simbolo exp = new Simbolo();

exp.valor1=factor.valor1;

RESULT=exp;

:};

OP ::= OPCOMP:opcomp

{:

Simbolo op = new Simbolo();

op.valor1=opcomp.valor1;

RESULT=op;

:};

OP ::= OPLOG:oplog

{:

Simbolo op = new Simbolo();

op.valor1=oplog.valor1;

RESULT=op;

:};

OP ::= OPARIT:oparit

{:

Simbolo op = new Simbolo();

op.valor1=oparit.valor1;

RESULT=op;

:};

OPCOMP ::= op\_menor

{:

Simbolo op = new Simbolo();

op.valor1=" < ";

RESULT=op;

:};

OPCOMP ::= op\_mayor

{:

Simbolo op = new Simbolo();

op.valor1=" > ";

RESULT=op;

:};

OPCOMP ::= op\_menorIgual

{:

Simbolo op = new Simbolo();

op.valor1=" <= ";

RESULT=op;

:};

OPCOMP ::= op\_mayorIgual

{:

Simbolo op = new Simbolo();

op.valor1=" >= ";

RESULT=op;

:};

OPCOMP ::= op\_igual

{:

Simbolo op = new Simbolo();

op.valor1= " = ";

RESULT=op;

:};

OPCOMP ::= op\_distinto

{:

Simbolo op = new Simbolo();

op.valor1=" <> ";

RESULT=op;

:};

OPARIT ::= op\_suma

{:

Simbolo op = new Simbolo();

op.valor1=" + ";

RESULT=op;

:};

OPARIT ::= op\_resta

{:

Simbolo op = new Simbolo();

op.valor1=" - ";

RESULT=op;

:};

OPARIT ::= op\_mult

{:

Simbolo op = new Simbolo();

op.valor1=" \* ";

RESULT=op;

:};

OPARIT ::= op\_divEntero

{:

Simbolo op = new Simbolo();

op.valor1="<SPAN CLASS=\"palres\">div</SPAN>";

RESULT=op;

:};

OPARIT ::= op\_mod

{:

Simbolo op = new Simbolo();

op.valor1="<SPAN CLASS=\"palres\">mod</SPAN>";

RESULT=op;

:};

OPARIT ::= op\_divReal

{:

Simbolo op = new Simbolo();

op.valor1= " / ";

RESULT=op;

:};

OPLOG ::= op\_or

{:

Simbolo op = new Simbolo();

op.valor1="<SPAN CLASS=\"palres\">or</SPAN>";

RESULT=op;

:};

OPLOG ::= op\_and

{:

Simbolo op = new Simbolo();

op.valor1="<SPAN CLASS=\"palres\">and</SPAN>";

RESULT=op;

:};

FACTOR ::= SIMPVALUE:simpvalue

{:

Simbolo factor = new Simbolo();

factor.valor1=simpvalue.valor1;

RESULT=factor;

:};

FACTOR ::= op\_not FACTOR:factor1

{:

Simbolo factor = new Simbolo();

factor.valor1="<SPAN CLASS=\"palres\">not &nbsp;"+factor1.valor1;

RESULT=factor;

:};

FACTOR ::= abrir\_paren EXP:exp cerrar\_paren

{:

Simbolo factor = new Simbolo();

factor.valor1="("+exp.valor1+")";

RESULT=factor;

:};

FACTOR ::= identifier:iden SUBPPARAMLIST:subpparamlist

{:

Simbolo factor = new Simbolo();

factor.valor1="<A HREF=\"#"+iden+"\">"+iden+"</A>"+subpparamlist.valor1;

RESULT=factor;

:};

FACTOR ::= identifier:iden abrir\_corchete EXP:exp cerrar\_corchete

{:

Simbolo factor = new Simbolo();

factor.valor1="<A HREF=\"#"+iden+"\">"+iden+"</A>[<SPAN CLASS=\"cte\">"+exp.valor1+"</SPAN>]";

RESULT=factor;

:};

FACTOR ::= identifier:iden

{:

Simbolo factor = new Simbolo();

factor.valor1="<A HREF=\"#"+iden+"\">"+iden+"</A>";

RESULT=factor;

:};

FACTOR ::= identifier:iden1 punto identifier:iden2

{:

Simbolo factor = new Simbolo();

factor.valor1="<A HREF=\"#"+iden1+"\">"+iden1+"</A>.<SPAN CLASS=\"ident\">"+iden2+"</SPAN>";

RESULT=factor;

:};

SUBPPARAMLIST ::= abrir\_paren EXPLIST:explist cerrar\_paren

{:

Simbolo subpparamlist = new Simbolo();

subpparamlist.valor1="("+explist.valor1+")";

RESULT=subpparamlist;

:};

EXPLIST ::= EXP:exp

{:

Simbolo explist = new Simbolo();

explist.valor1=exp.valor1;

RESULT=explist;

:};

EXPLIST ::= EXP:exp coma EXPLIST:explist1

{:

Simbolo explist = new Simbolo();

explist.valor1=exp.valor1+","+explist1.valor1;

RESULT=explist;

:};

PROC\_CALL ::= identifier:iden SUBPPARAMLIST:subpparamlist

{:

Simbolo proc = new Simbolo();

proc.valor1="<A HREF=\"#"+iden+"\">"+iden+"</A>"+subpparamlist.valor1;

RESULT=proc;

:};

PROC\_CALL ::= identifier:iden abrir\_paren cerrar\_paren

{:

Simbolo proc = new Simbolo();

proc.valor1="<A HREF=\"#"+iden+"\">"+iden+"</A>()";

RESULT=proc;

:};

EXEBLQ ::={:tab=tab+1;:} DCLLIST\_BLQ:dcllistblq begin SENTLIST:sentlist end

{:

Simbolo exeblq = new Simbolo();

exeblq.valor1 = "<BR>"+dcllistblq.valor2

+"<SPAN CLASS=\"palres\">begin</SPAN>"

+"<DIV style=\"text-indent: "+tab+"cm\">"+sentlist.valor1+"</DIV>"

+"<DIV style=\"text-indent: "+(tab-1)+"cm\"><SPAN CLASS=\"palres\">end</SPAN>;<DIV>";

RESULT=exeblq;

tab=tab-1;

:};

EXEBLQ ::={:tab=tab+1;:} begin SENTLIST:sentlist end

{:

Simbolo exeblq = new Simbolo();

exeblq.valor1 ="<SPAN CLASS=\"palres\">begin</SPAN>"

+"<DIV style=\"text-indent: "+tab+"cm\">"+sentlist.valor1+"</DIV>"

+"<DIV style=\"text-indent: "+(tab-1)+"cm\"><SPAN CLASS=\"palres\">end</SPAN>;</DIV>";

RESULT=exeblq;

tab=tab-1;

:};

DCLLIST\_BLQ ::= DCLLIST\_BLQ:dcllistblq1 DCL\_BLQ:dclblq

{:

Simbolo dcllistblq = new Simbolo();

dcllistblq.valor1=dcllistblq1.valor1

+ dclblq.valor1;

dcllistblq.valor2=dcllistblq1.valor2

+ dclblq.valor2;

RESULT=dcllistblq;

:};

DCLLIST\_BLQ ::= DCL\_BLQ:dclblq

{:

Simbolo dcllistblq = new Simbolo();

dcllistblq.valor1=dclblq.valor1;

dcllistblq.valor2=dclblq.valor2;

RESULT=dcllistblq;

:};

DCL\_BLQ ::= DEFCTE:defcte

{:

Simbolo dclblq = new Simbolo();

dclblq.valor1 = "";

dclblq.valor2 = defcte.valor2;

RESULT= dclblq;

:};

DCL\_BLQ ::= DEFTYPE:deftype

{:

Simbolo dclblq = new Simbolo();

dclblq.valor1 = "";

dclblq.valor2 = deftype.valor2;

RESULT= dclblq;

:};

DCL\_BLQ ::= DEFVAR:defvar

{:

Simbolo dclblq = new Simbolo();

dclblq.valor1 = "";

dclblq.valor2 = defvar.valor2;

RESULT= dclblq;

:};