import java\_cup.runtime.\*;

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 INICIAL;

non terminal Simbolo PRG;

non terminal Simbolo BLQ;

non terminal Simbolo BLQ2;

non terminal Simbolo DCLLIST2;

non terminal Simbolo DCLLIST;

non terminal Simbolo DCL;

non terminal Simbolo DCL2;

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:id punto\_coma BLQ:blq punto

{: Simbolo prg = new Simbolo();

prg.valor = “<!DOCTYPE html>\n<HTML>\n<head>\n<title>EjNotable.pl</title>\n”

+”<style>\n”

+”.cte {color:rgb(19,189,72);}\n”

+”.ident {color:rgb(55,40,244);}\n”

+”.palres {color:rgb(0,0,0);font-weight:bold;}\n”

+”</style>\n</head>\n”

+”<BODY>\n”

+”<A NAME=\"inicio\">\n”

+”<H1>Programa:”+”id.valor”+”;</H1>\n”

+”<H2>Funciones y procedimientos</H2>\n”

+”</BODY>\n</HTML>”;

try {

FileOutputStream fout = new FileOutputStream("Salida.log");

PrintStream out = new PrintStream(fout);

System.setOut(out);

System.out.print(prg.valor);

} catch (FileNotFoundException ex) {

ex.printStackTrace();

}

:};

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

{: Simbolo blq = new Simbolo();

blq.valor = //En valor1 esta los subprogrmas y en valor2 los variables

dcllist.valor1+”<A NAME=\"ProgPpal\"><H2>Programa” +”Principal</H2>”+dcllist.valor2+”<BR/><SPAN CLASS=\"palres\">begin</SPAN>”

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

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

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

System.out.print(blq.valor);

:};

BLQ ::= begin SENTLIST:sentlist end

{: Simbolo blq = new Simbolo();

blq.valor = ”<A NAME=\"ProgPpal\">\n<H2>Programa Principal</H2>\n<BR/>\n”

+“<SPAN CLASS=\"palres\">begin</SPAN>\n”+sentlist.valor

+”<SPAN CLASS=\"palres\">end</SPAN>\n<BR/>\n<A HREF=\"#ProgPpal\">”

+”Inicio del programa principal</A>\n<BR/>\n<A HREF=\"#inicio\">”

+”Inicio de programa</A>\n<BR/>”;

System.out.print(blq.valor);

:};

//////////////////////////////////////////////////////////////////////////////////////////////////////////////////

BLQ2 ::= DCLLIST2 begin SENTLIST end

{<UL>

<LI><A HREF="#areaCuadrado">function areaCuadrado ( lado: REAL ) : REAL </A></LI>

<LI><A HREF="#intercambioEntero">procedure intercambio ( v1, v2: INTEGER ) </A></LI>

<LI><A HREF="#ProgPpal">Programa princial</A></LI>

</UL>

<HR/>

<A NAME="areaCuadrado">

<SPAN CLASS="palres">function</SPAN> <SPAN CLASS="ident">areaCuadrado</SPAN> ( <A NAME="lado"><SPAN CLASS="ident">lado</SPAN>: <SPAN CLASS="palres">REAL</SPAN> ) : <SPAN CLASS="palres">REAL</SPAN> ;

<BR/><SPAN CLASS="palres">type</SPAN> <A NAME="miArray"><SPAN CLASS="ident">miArray</SPAN> = <SPAN CLASS="palres">array</SPAN> [ <SPAN CLASS="cte">0</SPAN> .. <SPAN CLASS="cte">5</SPAN> ] <SPAN CLASS="palres">of</SPAN> <SPAN CLASS="palres">INTEGER</SPAN> ;

<BR/><SPAN CLASS="palres">var</SPAN>

<A NAME="resultado"><SPAN CLASS="ident">resultado</SPAN>: <SPAN CLASS="palres">REAL</SPAN>;

<BR/><SPAN CLASS="palres">begin</SPAN>

<DIV style="text-indent: 0.5cm"><A HREF="#resultado">resultado</A> := <SPAN CLASS="cte">+0.0</SPAN>;</DIV>

<DIV style="text-indent: 0.5cm"><A HREF="#miArray">miArray</A>[<SPAN CLASS="cte">1</SPAN>] := 0</SPAN>;</DIV>

<DIV style="text-indent: 0.5cm"><A HREF="#resultado">resultado</A> := <A HREF="#lado">lado</A> \* <A HREF="#lado">lado</A> + <A HREF="#miArray">miArray</A>[ <SPAN CLASS="cte">1</SPAN> + <SPAN CLASS="cte">2</SPAN> ] ;</DIV>

<DIV style="text-indent: 0.5cm"><A HREF="#areaCuadrado">areaCuadrado</A> := <A HREF="#resultado">resultado</A>; </DIV>

<SPAN CLASS="palres">end</SPAN>; <BR/>

<A HREF="#areaCuadrado">Inicio de rutina</A><BR/>

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

<HR/>

<A NAME="intercambioEntero">

<SPAN CLASS="palres">procedure</SPAN> <SPAN CLASS="ident">intercambio</SPAN> ( <A NAME="v1"><SPAN CLASS="ident">v1</SPAN>, <A NAME="v2"><SPAN CLASS="ident">v2</SPAN>: <SPAN CLASS="palres">INTEGER</SPAN> ) ;

<BR/><SPAN CLASS="palres">var</SPAN>

<A NAME="aux"><SPAN CLASS="ident">aux</SPAN>: <SPAN CLASS="palres">INTEGER</SPAN>;

<BR/><SPAN CLASS="palres">begin</SPAN>

<DIV style="text-indent: 0.5cm"><SPAN CLASS="palres">if</SPAN> ( <A HREF="#v1">v1</A> <> <A HREF="#v2">v2</A> ) <SPAN CLASS="palres">then</SPAN> </DIV>

<DIV style="text-indent: 0.5cm"><SPAN CLASS="palres">begin</SPAN></DIV>

<DIV style="text-indent: 1cm"><A HREF="#aux">aux</A> := <SPAN CLASS="cte">0</SPAN>;</DIV>

<DIV style="text-indent: 1cm"><A HREF="#aux">aux</A> := <A HREF="#v1">v1</A>;</DIV>

<DIV style="text-indent: 1cm"><A HREF="#v1">v1</A> := <A HREF="#v2">v2</A>;</DIV>

<DIV style="text-indent: 1cm"><A HREF="#v2">v2</A> := <A HREF="#aux">aux</A>;</DIV>

<DIV style="text-indent: 0.5cm"><SPAN CLASS="palres">end</SPAN></DIV>

<SPAN CLASS="palres">end</SPAN>;<BR/>

<A HREF="#intercambioEntero">Inicio de rutina</A><BR/>

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

<HR/>

:};

BLQ2 ::= begin SENTLIST end

{:

<UL>

<LI><A HREF="#areaCuadrado">function areaCuadrado ( lado: REAL ) : REAL </A></LI>

<LI><A HREF="#intercambioEntero">procedure intercambio ( v1, v2: INTEGER ) </A></LI>

<LI><A HREF="#ProgPpal">Programa princial</A></LI>

</UL>

<HR/>

<A NAME="areaCuadrado">

<SPAN CLASS="palres">function</SPAN> <SPAN CLASS="ident">areaCuadrado</SPAN> ( <A NAME="lado"><SPAN CLASS="ident">lado</SPAN>: <SPAN CLASS="palres">REAL</SPAN> ) : <SPAN CLASS="palres">REAL</SPAN> ;

<BR/><SPAN CLASS="palres">type</SPAN> <A NAME="miArray"><SPAN CLASS="ident">miArray</SPAN> = <SPAN CLASS="palres">array</SPAN> [ <SPAN CLASS="cte">0</SPAN> .. <SPAN CLASS="cte">5</SPAN> ] <SPAN CLASS="palres">of</SPAN> <SPAN CLASS="palres">INTEGER</SPAN> ;

<BR/><SPAN CLASS="palres">var</SPAN>

<A NAME="resultado"><SPAN CLASS="ident">resultado</SPAN>: <SPAN CLASS="palres">REAL</SPAN>;

<BR/><SPAN CLASS="palres">begin</SPAN>

<DIV style="text-indent: 0.5cm"><A HREF="#resultado">resultado</A> := <SPAN CLASS="cte">+0.0</SPAN>;</DIV>

<DIV style="text-indent: 0.5cm"><A HREF="#miArray">miArray</A>[<SPAN CLASS="cte">1</SPAN>] := 0</SPAN>;</DIV>

<DIV style="text-indent: 0.5cm"><A HREF="#resultado">resultado</A> := <A HREF="#lado">lado</A> \* <A HREF="#lado">lado</A> + <A HREF="#miArray">miArray</A>[ <SPAN CLASS="cte">1</SPAN> + <SPAN CLASS="cte">2</SPAN> ] ;</DIV>

<DIV style="text-indent: 0.5cm"><A HREF="#areaCuadrado">areaCuadrado</A> := <A HREF="#resultado">resultado</A>; </DIV>

<SPAN CLASS="palres">end</SPAN>; <BR/>

<A HREF="#areaCuadrado">Inicio de rutina</A><BR/>

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

<HR/>

<A NAME="intercambioEntero">

<SPAN CLASS="palres">procedure</SPAN> <SPAN CLASS="ident">intercambio</SPAN> ( <A NAME="v1"><SPAN CLASS="ident">v1</SPAN>, <A NAME="v2"><SPAN CLASS="ident">v2</SPAN>: <SPAN CLASS="palres">INTEGER</SPAN> ) ;

<BR/><SPAN CLASS="palres">var</SPAN>

<A NAME="aux"><SPAN CLASS="ident">aux</SPAN>: <SPAN CLASS="palres">INTEGER</SPAN>;

<BR/><SPAN CLASS="palres">begin</SPAN>

<DIV style="text-indent: 0.5cm"><SPAN CLASS="palres">if</SPAN> ( <A HREF="#v1">v1</A> <> <A HREF="#v2">v2</A> ) <SPAN CLASS="palres">then</SPAN> </DIV>

<DIV style="text-indent: 0.5cm"><SPAN CLASS="palres">begin</SPAN></DIV>

<DIV style="text-indent: 1cm"><A HREF="#aux">aux</A> := <SPAN CLASS="cte">0</SPAN>;</DIV>

<DIV style="text-indent: 1cm"><A HREF="#aux">aux</A> := <A HREF="#v1">v1</A>;</DIV>

<DIV style="text-indent: 1cm"><A HREF="#v1">v1</A> := <A HREF="#v2">v2</A>;</DIV>

<DIV style="text-indent: 1cm"><A HREF="#v2">v2</A> := <A HREF="#aux">aux</A>;</DIV>

<DIV style="text-indent: 0.5cm"><SPAN CLASS="palres">end</SPAN></DIV>

<SPAN CLASS="palres">end</SPAN>;<BR/>

<A HREF="#intercambioEntero">Inicio de rutina</A><BR/>

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

<HR/>”;

:};

/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

DCLLIST ::= DCLLIST:dcllist DCL:dcl

{:

<A NAME="ProgPpal">

<H2>Programa Principal</H2>

<SPAN CLASS="palres">var</SPAN>

<A NAME="medida"><SPAN CLASS="ident">medida</SPAN>: <SPAN CLASS="palres">REAL</SPAN>;

<A NAME="valor1"><SPAN CLASS="ident">valor1</SPAN>, <A NAME="valor2"><SPAN CLASS="ident">valor2</SPAN>: <SPAN CLASS="palres">INTEGER</SPAN>;

:};

DCLLIST ::= DCL

{:

<A NAME="ProgPpal">

<H2>Programa Principal</H2>

<SPAN CLASS="palres">var</SPAN>

<A NAME="medida"><SPAN CLASS="ident">medida</SPAN>: <SPAN CLASS="palres">REAL</SPAN>;

<A NAME="valor1"><SPAN CLASS="ident">valor1</SPAN>, <A NAME="valor2"><SPAN CLASS="ident">valor2</SPAN>: <SPAN CLASS="palres">INTEGER</SPAN>;

:};

DCLLIST2 ::= DCLLIST2 DCL2

{: :};

DCLLIST2 ::= DCL2

{: :};

SENTLIST ::= SENT

{:

<DIV style="text-indent: 0.5cm"><A HREF="#medida">medida</A> := <SPAN CLASS="cte">$4.A</SPAN>;</DIV>

<DIV style="text-indent: 0.5cm"><A HREF="#valor1">valor1</A> := <SPAN CLASS="cte">-3</SPAN>;</DIV>

<DIV style="text-indent: 0.5cm"><A HREF="#valor2">valor2</A> := <SPAN CLASS="cte">$F6</SPAN>;</DIV>

<DIV style="text-indent: 0.5cm">

<SPAN CLASS="palres">while</SPAN> <A HREF="#medida">medida</A> < <A HREF="#valor1">valor1</A> <SPAN CLASS="palres">do</SPAN>

<A HREF="#medida">medida</A> := <A HREF="#areaCuadrado">areaCuadrado</A>( <A HREF="#medida">medida</A> );</DIV>

<DIV style="text-indent: 0.5cm"><A HREF="#intercambioEntero">intercambio</A>(<A HREF="#valor1">valor1</A>, <A HREF="#valor2">valor2</A>);</DIV>

:};

SENTLIST ::= SENTLIST SENT

{:

<DIV style="text-indent: 0.5cm"><A HREF="#medida">medida</A> := <SPAN CLASS="cte">$4.A</SPAN>;</DIV>

<DIV style="text-indent: 0.5cm"><A HREF="#valor1">valor1</A> := <SPAN CLASS="cte">-3</SPAN>;</DIV>

<DIV style="text-indent: 0.5cm"><A HREF="#valor2">valor2</A> := <SPAN CLASS="cte">$F6</SPAN>;</DIV>

<DIV style="text-indent: 0.5cm">

<SPAN CLASS="palres">while</SPAN> <A HREF="#medida">medida</A> < <A HREF="#valor1">valor1</A> <SPAN CLASS="palres">do</SPAN>

<A HREF="#medida">medida</A> := <A HREF="#areaCuadrado">areaCuadrado</A>( <A HREF="#medida">medida</A> );</DIV>

<DIV style="text-indent: 0.5cm"><A HREF="#intercambioEntero">intercambio</A>(<A HREF="#valor1">valor1</A>, <A HREF="#valor2">valor2</A>);</DIV>

:};

DCL ::= DEFCTE:defcte

{:

Simbolo dcl = new Simbolo();

dcl.tipo = String;

dcl.valor = defcte.valor;

RESULT= dcl;

:} ;

DCL ::= DEFTYPE:deftype

{:

Simbolo dcl = new Simbolo();

dcl.tipo = String;

dcl.valor = deftype.valor;

RESULT= dcl;

:} ;

DCL ::= DEFVAR:defvar

{:

Simbolo dcl = new Simbolo();

dcl.tipo = String;

dcl.valor = defvar.valor;

RESULT= dcl;

:} ;

DCL ::= DEFPROC:defproc

{:

Simbolo dcl = new Simbolo();

dcl.tipo = String;

dcl.valor = defproc.valor;

RESULT= dcl;

:} ;

DCL ::= DEFFUN:deffun

{:

Simbolo dcl = new Simbolo();

dcl.tipo = String;

dcl.valor = deffun.valor;

RESULT= dcl;

:} ;

DCL2 ::= DEFCTE

{:

:} ;

DCL2 ::= DEFTYPE

{:

:} ;

DCL2 ::= DEFVAR

{:

:} ;

DEFCTE ::= const\_ CTELIST

{:

:} ;

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

{:

:} ;

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

{:

:} ;

DEFTYPE ::= type TYPELIST

{:

:} ;

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

{:

:} ;

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

{:

:} ;

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

{:

:} ;

UDTYPE ::= record DEFVARLIST punto\_coma end

{:

:} ;

SIMPVALUE ::= numeric\_integer\_const

{:

:} ;

SIMPVALUE ::= numeric\_integerHex\_const

{: :};

SIMPVALUE ::= numeric\_real\_const

{: :};

SIMPVALUE ::= numeric\_realHex\_const

{: :};

SIMPVALUE ::= string\_const

{: :};

DEFVAR ::= var DEFVARLIST punto\_coma

{: :};

DEFVARLIST ::= VARLIST dos\_puntos ALLTYPES

{: :};

DEFVARLIST ::= DEFVARLIST punto\_coma VARLIST dos\_puntos ALLTYPES

{: :};

VARLIST ::= identifier

{: :};

VARLIST ::= identifier coma VARLIST

{: :};

DEFPROC ::= procedure identifier FORMAL\_PARAMLIST punto\_coma BLQ punto\_coma

{: :};

DEFPROC ::= procedure identifier punto\_coma BLQ punto\_coma

{: :};

DEFFUN ::= function identifier FORMAL\_PARAMLIST dos\_puntos ALLTYPES punto\_coma BLQ punto\_coma

{: :};

DEFFUN ::= function identifier dos\_puntos ALLTYPES punto\_coma BLQ punto\_coma

{: :};

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

{: :};

FORMAL\_PARAM ::= VARLIST dos\_puntos ALLTYPES

{: :};

FORMAL\_PARAM ::= VARLIST dos\_puntos ALLTYPES punto\_coma FORMAL\_PARAM

{: :};

TBAS ::= integer

{: :};

TBAS ::= real

{: :};

TBAS ::= char\_

{: :};

ALLTYPES ::= TBAS

{: :};

ALLTYPES ::= identifier

{: :};

SENT ::= ASIG:asig punto\_coma

{:

Simbolo sent = new Simbolo();

sent.tipo = String;

sent.valor=asig.valor+”;<BR/>”;

RESULT=sentlist;

:};

SENT ::= PROC\_CALL punto\_coma

{: :};

SENT ::= EXEBLQ

{: :};

SENT ::= COND

{: :};

SENT ::= WLOOP

{: :};

SENT ::= FLOOP

{: :};

SENT ::= CASE

{: :};

COND ::= if\_ EXP then SENT

{: :};

COND ::= if\_ EXP then SENT ELSECOND

{: :};

ELSECOND ::= else\_ SENT

{: :};

WLOOP ::= while\_ EXP do\_ SENT

{: :};

FLOOP ::= for\_ identifier op\_asignacion EXP to EXP do\_ SENT

{: :};

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

{: :};

CASELIST ::= EXP dos\_puntos SENT

{: :};

CASELIST ::= EXP dos\_puntos SENT CASELIST

{: :};

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

{:Simbolo sent = new Simbolo();

sent.tipo = String;

sent.valor=asig.valor+”;<BR/>”;

RESULT=sentlist;

:};

ID ::= identifier

{: :};

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

{: :};

ID ::= identifier punto identifier

{: :};

EXP ::= EXP OP EXP

{: :};

EXP ::= FACTOR

{: :};

OP ::= OPCOMP

{: :};

OP ::= OPLOG

{: :};

OP ::= OPARIT

{: :};

OPCOMP ::= op\_menor

{: :};

OPCOMP ::= op\_mayor

{: :};

OPCOMP ::= op\_menorIgual

{: :};

OPCOMP ::= op\_mayorIgual

{: :};

OPCOMP ::= op\_igual

{: :};

OPCOMP ::= op\_distinto

{: :};

OPARIT ::= op\_suma

{: :};

OPARIT ::= op\_resta

{: :};

OPARIT ::= op\_mult

{: :};

OPARIT ::= op\_divEntero

{: :};

OPARIT ::= op\_mod

{: :};

OPARIT ::= op\_divReal

{: :};

OPLOG ::= op\_or

{: :};

OPLOG ::= op\_and

{: :};

FACTOR ::= SIMPVALUE

{: :};

FACTOR ::= op\_not FACTOR

{: :};

FACTOR ::= abrir\_paren EXP cerrar\_paren

{: :};

FACTOR ::= identifier SUBPPARAMLIST

{: :};

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

{: :};

FACTOR ::= identifier | identifier punto identifier

{: :};

SUBPPARAMLIST ::= abrir\_paren EXPLIST cerrar\_paren

{: :};

EXPLIST ::= EXP

{: :};

EXPLIST ::= EXP coma EXPLIST

{: :};

PROC\_CALL ::= identifier SUBPPARAMLIST

{: :};

PROC\_CALL ::= identifier

{: :};

EXEBLQ ::= DCLLIST\_BLQ begin SENTLIST end punto\_coma

{:EXEBLQ.valor=(DCLLIST\_BLQ.valor++”begin”++SENTLIST.valor++”end;”);:};

EXEBLQ ::= begin SENTLIST end punto\_coma

{: :};

DCLLIST\_BLQ ::= DCLLIST\_BLQ punto\_coma DCL\_BLQ

{:DCLLIST\_BLQ.valor=( DCLLIST\_BLQ1.valor++”;”++DCL\_BLQ.valor);:};

DCLLIST\_BLQ ::= punto\_coma DCL\_BLQ

{:DCLLIST\_BLQ.valor=(“;”++DCL\_BLQ.valor);:};

DCL\_BLQ ::= DEFCTE

{:DCL.valor=DEFCTE.valor;:};

DCL\_BLQ ::= DEFTYPE

{:DCL.valor=DEFTYPE.valor;:};

DCL\_BLQ ::= DEFVAR

{:DCL.valor=DEFVAR.valor;:};