7. A P E N D I C E S.

7.1. LISTADO DEL CODIGO EN LENGUAJE PASCUAL.

```
                    { ********************************************* }
      { ****  Modulo principal del Qenqo          *** }
      { ********************************************* }
      { ** Autor : JOSE LUIS SEGOVIA ** }
      { ** Lima-Peru 1991. ** }

{$R+,S+,I+,D+,T-,F-,V+,B+,N-,L+ } {$M 40000,0,150000 }

PROGRAM Interprete_para_dinamica_de_Sistemas__QENQO;
USES crt, dos, pantuni, lexanuni, parseuni, ejecuni, grafuni, printer;
VAR Snts:INTEGER;
PROCEDURE escoge_opcion; FORWARD;
PROCEDURE muestra_pantalla; FORWARD;
{$F+}
PROCEDURE misalida;        { ** Controla la salida por errores ** }
BEGIN
   exitproc:=puntero_salida;
   cierragrafico;
   gotoxy(1,lastrow); clreol;
   write('Pulse cualquier tecla para continuar.');
   REPEAT UNTIL keypressed;
   muestra_pantalla;
   escoge_opcion;
END;
{$F-}

PROCEDURE lee_archivo;
          { ********************************************** }
          { *** Selecciona archivos para su compilacion *** }
          { ********************************************** }
   VAR halla : BOOLEAN;
BEGIN halla:=FALSE;
   window(1, lastrow, lastcol, lastrow); CLRSCR;
   poncursor(12);
   REPEAT
      gotoxy(1,lastrow-1); clreol;
      write('Ingrese nombre de archivo :');
      readln(n_archivo);
      assign(archivo, n_archivo);
      gotoxy(1,lastrow-1); clreol;
      IF (NOT existe_arch(n_archivo)) OR (n_archivo <= ' ') THEN
      BEGIN
         write('Archivo <', n_archivo, '> no existe.');
         n_archivo := 'SIN_NOMBRE.Q';
      END
      ELSE halla:=TRUE;
   UNTIL halla OR (n_archivo = 'SIN_NOMBRE.Q');
   window(1,1,lastcol,lastrow); escribeinvln(10,lastrow-1,'                    ');
   escribeinvln(10,lastrow-1,n_archivo);
   poncursor($2000);
END;


PROCEDURE Lee_variable_S;
          { ************************************************ }
          { ** Lee la variable en Analisis de Sensibilidad ** }
          { ************************************************ }
```

3.2.6. ADMINISTRACION DE LA MEMORIA.

Se han establecido dos tablas principales: La Tabla de Símbolos y la Tabla de Código Intermedio.

3.2.6.1. LA TABLA DE SIMBOLOS.

La Tabla de Símbolos es el lugar donde se almacenan los tokens o símbolos y sus datos relacionados. Lo constituye un arreglo de determinado tamaño, conteniendo registros con los siguientes datos:

a) Nombre del Símbolo. Cadena de identificación.

b) Objeto. Entero para identificar el token o símbolo.

c) Atributo. Para diferenciar si es variable, función, subsistema o real absoluto.

d) Nivel. Para diferenciar su profundidad en la anidacion estructurada.

e) Valor. Determina el valor real en caso de ser variable.

La Tabla de Símbolos tiene dos partes, la primera contiene los tokens de las Palabras Reservadas y la segunda es utilizado por el sistema durante la compilación.

Ejemplo:

NUMERO	NOMBRE	OBJETO	ATRIBUTO	NIVEL	VALOR
...					
111					
112	'NOMBREVAR'	402	VARIABLE	'1'	12.00
113					
...					

3.2.6.2. LA TABLA DE CODIGO INTERMEDIO.

La Tabla de Código Intermedio es el lugar donde se almacena el código generado por el Analizador Sintáctico/Generador de Código. Es un arreglo de registros los siguiente datos:

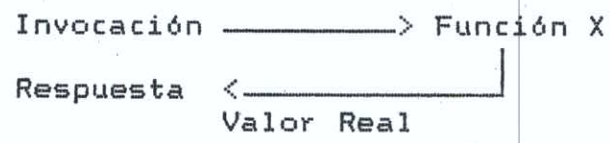
a) Operador.
b) Primer Operando.
c) Segundo Operando.
d) Tercer Operando.

3.2.6.3. OTRAS TABLAS.

Ha sido necesario utilizar otros arreglos pequeños para almacenar datos como los de la Tabla de errores, tabla de datos de la gráfica que se detallan en la descripción de cada módulo.

3.2.7. FUNCIONES.

Una función devuelve un valor real luego de haber realizado una o más órdenes secuenciales.



3.2.8. SUBSISTEMAS.

Un sistema complejo se divide en subsistemas donde se describen las relaciones entre cada una de sus propias variables.

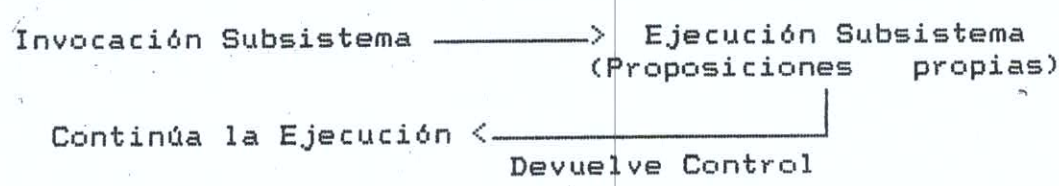
Un subsistema puede subdividirse en otro de menor jerarquía, hasta en 8 niveles de profundidad, cada uno de los cuales puede tener sus funciones propias.

Definición de los subsistemas:

```
Subsistema N1;
        Subsistema N1_1;            {Dentro de N1}
              Subsistema N1_1_1;    {Dentro de N1_1}
                   <Cuerpo del Bloque N1_1_1>
              Subsistema N1_1_2;    {dentro de N1_1}
                   <Cuerpo del Bloque N1_1_2>
        <Cuerpo del Bloque N1_1>
   <Cuerpo del Bloque N1>
```

Invocación de los subsistemas:

Se invoca a un subsistema para que ejecute sus proposiciones internas, luego de las cuales retorna el control.



Dentro del bloque N1_1 se puede invocar a cualquiera de sus subsistemas de menor jerarquía para que se ejecuten.

```
Subsistema N1_1;              {Declaración de N1_1}
   ..
   Subsistema N1_1_1;         {Declaración de N1_1_1}
      ..
   Subsistema N1_1_2;         {Declaración de N1_1_2}
      ..
   Inicio                     {Inicio de  N1_1}
  ..
     N1_1_2;                  {Invocación a N1_1_2;}
     N1_1_1;                  {Invocación a N1_1_1;}
  ..
   Fin;                       {Fin de declaración de N1_1}
```

3.2.9. AMBITO DE LAS VARIABLES.

Las variables declaradas dentro de un subsistema sólo pueden utilizarse dentro de éste. El paso de información de un subsistema a otro se realiza mediante la declaración y el paso de parámetros.

```
. . .
Subsistema N1;
var a,b,c : flujo;            {Declaración de variables}
  Subsistema N1_1;
  var a1, b1, c1:flujo;
  Inicio
     ..
  a = b1;              {Modificación del estado de la
     ..                   {variable a en el subsistema N1_1 }
  Fin;
Inicio
  ...
  b = ***
  b1 = a;                 {Error, b1 no es variable de N1}
  ***
Fin;
```

Para el sistema de anidamiento se establece un arreglo

especial donde se reconoce la profundidad de la declaración de las variables, hasta un máximo de 8 niveles de profundidad.

3.2.10. DEFINICION Y PASO DE PARAMETROS.

Las funciones o los subsistemas se pueden definir con o sin parámetros. Una función se puede definir de la forma siguiente :

```
FUNCION Nombre (v1,v2 : <tipo>) : <tipo>;
          <Cuerpo>;
```

donde <tipo> es cualquier tipo excepto Tabla.

Una función se invoca dentro de una expresión numérica como sigue:

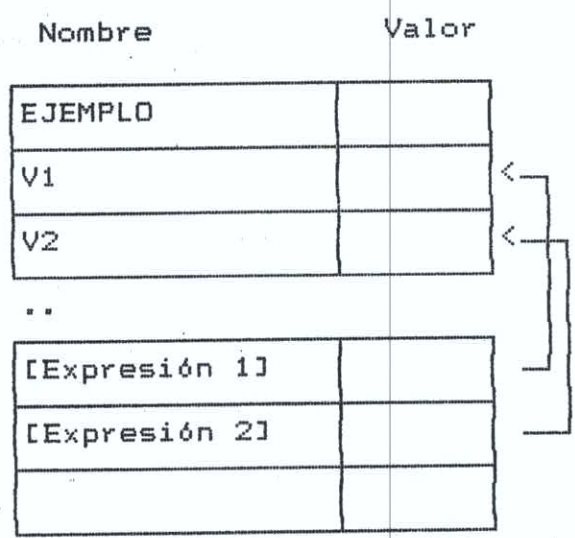
```
Variable = Ejemplo(<Expresión1>, <Expresión2>);
```

En el caso de invocación a subsistemas o funciones el paso de parámetros es por valor, de la siguiente manera:

El valor de <Expresión 1> pasa a -----> v1
El valor de <Expresión 2> pasa a -----> v2
siempre y cuando sean del mismo tipo.

En la tabla de símbolos se aprecia :

Tabla de Símbolos



```
   VAR halla : BOOLEAN; j:INTEGER;
BEGIN halla:=FALSE;
   window(1, lastrow, lastcol, lastrow); CLRSCR;
   poncursor(12);
   REPEAT
      gotoxy(1,lastrow-1); clreol;
      write(' Ingrese Variable :');
      readln(SVar);
      gotoxy(1,lastrow-1); clreol;
      FOR j:=CONT_IDENT DOWNTO 1 DO
         IF SVar = ts[j].nombre THEN BEGIN halla:=TRUE;  SNts:=j; END;
      IF (NOT halla) AND (SVAR <> '') THEN write('Variable <', SVar, '> no existe.');
   UNTIL halla OR (SVAR = '');
   window(1,1,lastcol,lastrow);
   poncursor($2000);
END;

PROCEDURE Lee_ParSensibil(VAR Variable:REAL; f,c:INTEGER);
             { ******************************************************** }
             { ** Toma la variable para el analisis de sensibilidad ** }
             { ******************************************************** }
   VAR cr:STRING[9];
BEGIN
   Variable:=leerealxy(fila[c,f].x,fila[c,f].y);
   str(Variable:9,cr);
   escribeinvln(fila[c,f].x,fila[c,f].y,cr);
END;


PROCEDURE carga_ts_globales;
             { ******************************************************** }
             { * Carga los datos de la tabla de simbolos al trabajos * }
             { ******************************************************** }
   VAR i,j:INTEGER;
BEGIN
   j:=1;
   FOR i:=1 TO cont_ident DO
      IF (ts[i].atr = ID_NIVEL) OR (ts[i].atr = ID_CONTROL) THEN
      BEGIN
         prueba[1,j].numhis:=1;
         prueba[1,j].nomvar:=ts[i].nombre;
         prueba[1,j].nts   :=i;
         prueba[1,j].valor :=ts[i].valor;
         j:=j+1;
      END;
   FOR i:=2 TO npruebas DO prueba[i]:=prueba[1];
END;

PROCEDURE carga_globales_ts(numprueba:byte);
              { ***************************************** }
              { ** carga de globales a tabla de simbolos * }
              { ***************************************** }
   VAR i:INTEGER;
BEGIN
   FOR i:=1 TO nfilc6 DO
      IF prueba[numprueba,i].nts > 0 THEN
         ts[prueba[numprueba,i].nts].valor:=prueba[numprueba,i].valor;
END;
```

```
PROCEDURE graficasensib;
      { ********************************************************* }
       { *** Realiza los graficos de Analisis de Sensibilidad *** }
       { ********************************************************* }
  VAR r1 :REAL;
BEGIN
  r1:=SIni;
  WHILE r1 <= SFIn DO
  BEGIN
    EJECUTA(1);
    carga_globales_ts(npruebax);
    ts[Snts].valor:=R1;
    ejecuta(2);
    grafica(narchwork);
    R1:=R1+SInc;
  END;
END;

PROCEDURE muestra_pantalla;
BEGIN
  clrscr;
  escribeinvln(1,1, '*  QENQO *        INTERPRETE PARA MODELOS EN DINAMICA DE SISTEMAS
');
  escribenorln(1,2,'Archivos  Editar  coRrre  Compilar  Calculador  Globales  Vsensibil  Salir');
  escribeinvln(1,lastrow-1,'ARCHIVO:                                !(c) Jose  Segovia Juarez, 1991.!<-
-> Mover');
  escribeinvln(10,lastrow-1,n_archivo);
END;

PROCEDURE blanquea_arreglos;
           { ************************************* }
           { ** Inicializa los arreglos con ceros ** }
           { ************************************* }
BEGIN
  c:=' ';
  paso:=1;
  numlin:=1;
  numcol:=1;
  ininter:=1;
  ntabwrit:=1;
  numerror:=0;
  nvargraf:=0;
  lastrow := 25;
  lastcol := 79;
  flag:=FALSE;
  flagapos:=TRUE;
  flagdosp:=FALSE;
  flagdecvar:=FALSE;
  flagnivel:=FALSE;
  final:=FALSE;
  flagtok := FALSE; numero:=FALSE;
  asig_a_tabla:=FALSE;
  fillchar(ts,sizeof(ts),0);
  fillchar(ptc,sizeof(ptc),#32);
  fillchar(TE,sizeof(TE),' ');
  fillchar(tcode,sizeof(tcode),0);
  fillchar(dat_vargraf,sizeof(dat_vargraf),0);
```

```
  fillchar(prueba,sizeof(prueba),0);
  fillchar(tab_interp, sizeof(tab_interp),0);
  randomize;
  ninterp := 0;
  narchwork:=' ';
  mtoken:=0;
  pbloque:=1;
  idbloqueG:='1';
  VALOR_real:=0;
  prx:=0; cx:=0; vtn:=0; varx:=0;
  iniciog:=10; finalg:=11;
  snts:=0; N_MAXX:=40; N_MAXY:=20;
  SVAR:=' '; sINI:=0; sFIN:=0; SINC:=0;
  carga;
  FOR i := 1 TO 16 DO
     area_id_bloque[i]:=idbloqueG; {copia estado de idbloqueg}
  cont_ident:=maxpalres;
END;

PROCEDURE escoge_opcion;
                    { ************************************ }
                    { ** Selecciona las opciones del menu ** }
                    { ************************************ }
  VAR  val_ESC,flag_interior:BOOLEAN; col_ant:byte; v1:INTEGER; ccr:STRING[9];
  PROCEDURE corre_opcion(opcion:STRING);     {** libreria de procedimientos **}
     VAR n1 :INTEGER; rc:INTEGER; cr:STRING[9];
  BEGIN
     IF opcion = 'LL' THEN lee_archivo;
     IF opcion = 'CP' THEN
     BEGIN
        blanquea_arreglos;
        parser; gotoxy(1, 12); clreol;
        ejecuta(1);
        carga_ts_globales;
        gotoxy((lastcol DIV 2)-15, 12); clreol; writeln('Fin de Compilacion.  ');
     END;
     IF opcion = 'RN' THEN
     BEGIN
        IF cx > 1 THEN
        BEGIN
           gotoxy((lastcol DIV 2)-15, lastrow DIV 2); clreol; write('INTEGRANDO a T = ');
           carga_globales_ts(npruebax);
           ejecuta(2);
           gotoxy((lastcol DIV 2 - 15), lastrow DIV 2);
           write('Pulse una tecla para Graficar   ');
           REPEAT UNTIL keypressed;
           IF nvargraf > 1 THEN
           BEGIN
              gotoxy(1, 12); clreol;
              ini_grafico;
              grafico_general;
              IF (SInc > 0) AND (SVAr > ' ') THEN graficasensib
              ELSE grafica(narchwork);
              cr:='X'; readln(cr);
              cierragrafico;
           END;
           muestra_pantalla; escribeinv(columx);
        END;
```

```
      END;
      IF opcion = 'HI' THEN
      BEGIN
         npruebax:=round(leerealxy(fila[columx,1].x,fila[columx,1].y));
         IF (npruebax <= 0) OR (npruebax > npruebas) THEN npruebax := 1;
         IF (npruebax > 1) AND (prueba[npruebax,1].valor = 0)
            THEN prueba[npruebax]:=prueba[1];
         str(npruebax:9,cr);
         fila[6,1].texto:=cr;
         escribeinvln(fila[6,1].x,fila[6,1].y,fila[6,1].texto);
      END;
      IF (((opcion >= '1') AND (opcion <= '9')) OR
         ((opcion >= '10') AND (opcion <= '15'))) AND (npruebax > 1) THEN
      BEGIN
         val(opcion,n1,rc);
         prueba[npruebax,n1].valor :=
         leerealxy(fila[columx,filx].x,fila[columx,filx].y);
         str(prueba[npruebax,n1].valor:9,cr);
         escribeinvln(fila[columx,filx].x,fila[columx,filx].y,cr);
      END;
      IF opcion ='V1' THEN BEGIN
         lee_variable_S;
      END;
      cad := '  ';
      IF opcion ='V2' THEN BEGIN lee_ParSensibil(SIni,filx,columx);
         str(Sini:9:2,cad);
      END;
      IF opcion ='V3' THEN BEGIN lee_ParSensibil(Sfin,filx,columx);
         str(Sfin:9:2,cad);
      END;
      IF opcion ='V4' THEN BEGIN lee_Parsensibil(SInc,filx,columx);
         str(Sfin:9:2,cad);
      END;
     {if (opcion >='V2') and (opcion <= 'V4') then escribenorln(filx,columx,Cad);}
      IF opcion ='ED' THEN BEGIN gotoxy(1, 12); clreol;
         exec('NE.COM',N_archivo);
         cx:=0;
         muestra_pantalla; escribeinv(columx);
      END;
   END;

BEGIN
   val_esc:=FALSE;
   puntero_salida:=exitproc;
   exitproc:=@misalida;
   col_ant:=8;
   lt:=#31; opcion:=' ';
   v1:=1; filx:=1;
   REPEAT
      poncursor($2000);
      lt:=leetecla;
      window(1,1,lastcol,lastrow);
      IF (lt <> upkey) AND (lt <> downkey) THEN escribenor(columx);
      IF lt = rightkey THEN columx:=columna[columx].der;
      IF lt = leftkey THEN columx:=columna[columx].izq;
      IF val_esc OR (columx = 2) OR (columx = 3) OR (columx = 4) OR (columx = 5) THEN BEGIN
         IF lt = HOMEkey THEN columx:=1;
         IF lt = ENDkey THEN columx:=ncolm;
```

```
      END;
      IF lt = Downkey THEN BEGIN val_esc:=FALSE;
         filx:=columna[columx].numfil;
      END;
      IF lt = CR THEN BEGIN val_esc:=FALSE;
         opcion:=columna[columx].proceso;
      END;
      IF lt = ESC THEN val_esc:=TRUE;
      IF (lt <> upkey) AND (lt <> downkey) THEN escribeinv(columx);
      IF (val_esc) OR (col_ant <> columx) THEN limpiaventana(col_ant);
      IF (NOT val_esc) THEN haceventana(columx);
                     {** dentro de fila **, escoger solo opciones }
      IF (NOT val_esc) AND NOT ((columx = 2) OR (columx = 3) OR (columx = 4) OR (columx=5)) THEN
      BEGIN
         IF lt = HOMEkey THEN columna[columx].numfil:=1;
         IF lt = ENDkey THEN columna[columx].numfil:=columna[columx].tot_filas;
         v1:=columna[columx].numfil; filx:=v1;
         IF lt = upkey   THEN filx:=fila[columx,v1].arriba;
         IF lt = downkey THEN filx:=fila[columx,v1].abajo;
         v1:=filx;
         escribeinvln(fila[columx,v1].x,fila[columx,v1].y,fila[columx,v1].texto);
         columna[columx].numfil:=filx;
         columna[columx].proceso:=fila[columx,v1].proceso;
      END;
      col_ant:=columx;
      IF lt = CR THEN corre_opcion(opcion);
   UNTIL opcion='Ss';
   exitproc:=NIL;
   poncursor(1); {defecto}
   textmode(origmode);
   writeln('Gracias por utilizar Qenqo.');
END;
                { ****************************************** }
                {          Programa principal                }
                { ****************************************** }
BEGIN
clrscr;
n_archivo:='SIN_NOMB.Q';
IF paramstr(1) > ' ' THEN
BEGIN n_archivo:=paramstr(1);
assign(archivo, n_archivo); END;
IF NOT existe_arch(n_archivo) THEN
BEGIN
   n_archivo:='SIN_NOMB.Q';
   assign(archivo, n_archivo);
   rewrite(archivo); write(archivo,'Archivo Ficticio.');
   close(archivo);
END;
blanquea_arreglos;
inicia_pantalla;
carga_datos_pantalla;
muestra_pantalla;
npruebax:=1;
escribeinv(columx);
escoge_opcion;
END.
```

```
    { ************************************************* }
    { *** Unidad de administracion de pantalla inicial *** }
    { ************************************************* }
    { * Autor : JOSE LUIS SEGOVIA JUAREZ. * }
    { * Lima-Peru *. 1991 * }

UNIT pantuni;
INTERFACE
USES crt, dos;
    { ****************************** }
    { ** Definicion de Constantes ** }
    { ****************************** }
CONST
   ERRORCOLOR = 140;
   NULL  = #0;
   BS = #8;
   FORMFEED = #12;
   CR = #13;
   ESC = #27;
   HOMEKEY = #199;
   ENDKEY  = #207;
   UPKEY   = #200;
   DOWNKEY = #208;
   PGUPKEY = #201;
   PGDNKEY = #209;
   LEFTKEY = #203;
   INSKEY  = #210;
   RIGHTKEY = #205;
   DELKEY   = #211;
   CTRLLEFTKEY = #243;
   CTRLRIGHTKEY = #244;
   F1 = #187;
   F2 = #188;
   F3 = #189;
   F4 = #190;
   F5 = #191;
   F6 = #192;
   F7 = #193;
   F8 = #194;
   F9 = #195;
   F10 = #196;
   ncolm  = 8;          { **  Numero de columnas que muestra  ** }
   nfilas = 16;         { **  Numero de filas en cada columna ** }
   nfilc1 = 2;          { **  Num. filas en columna 1 ** }
   nfilc6 = 16;         { **  Num. filas en columna 6 ** }
   nfilc7 = 4;          { **  Num. filas en columna 7 ** }
   nfilc8 = 2;          { **  Num. filas en columna 8 ** }
   npruebas = 10;       { **  Num. de registros historicos de modelo ** }

 TYPE
       { **** Datos de contenido de columna **** }
   tipo_columna = RECORD
      numcol     : byte;       { * Num. de columna * }
      contenido  : STRING[10]; { * Texto * }
      numfil,                  { * Numero de fila actual * }
      tot_filas,               { * Num. total de filas * }
```

```
    izq,der    : byte;        { * Columna de izq. y der * }
    x,y        : byte;        { * ubicacion en pantalla * }
    vx1,vy1,vx2,vy2 : byte;   { * Ventana en pantall * }
    proceso    : STRING[2];   { * Codigo de Proceso que invoca * }
  END;
  tipo_fila = RECORD                { * Datos de filas * }
    numfil     : byte;        { * Numero de fila * }
    texto      : STRING[12];  { * Texto que muestra * }
    arriba,abajo : byte;      { * punteros hacia arriba y abajo * }
    x,y        : byte;        { * ubicacion en pantalla * }
    proceso    : STRING[2];   { * Codigo de Proceso que invoca* }
  END;

  tipo_historia = RECORD            { * Datos de Trabajos * }
    numhis :byte;             { * Numero de trabajo * }
    nomvar :STRING[12];       { * Nombre de variable * }
    nts    :INTEGER;          { * numero en tabla de simbolos * }
    valor  :REAL;             { * valor actual de variable *}
  END;

VAR
  columna : ARRAY [1..ncolm ] OF tipo_columna;              { * Define columnas * }
  fila    : ARRAY [1..ncolm, 1..nfilas] OF tipo_fila;       { * Define filas * }
  prueba  : ARRAY [1..npruebas,1..nfilas] OF tipo_historia; { * Define pruebas * }
  filx,columx:byte;                     { * fila y columna actual * }
  lt:CHAR;                              { * Para lectura de teclado * }
  npruebax   :INTEGER;                  { * Registra el numero de historia * }
  opcion:STRING[2];                     { * Opcion seleccionada en codigo * }
  lastrow, lastcol,origmode :word;      { * auxiliares * }
  cad:STRING[2];
  sini, sfin,sinc:REAL; svar:STRING[12];  { ** variables de sensibilidad ** }

procedure PonCursor(NuevoCursor:integer);
function leetecla:char;
procedure carga_datos_pantalla;
procedure escribeinvln(x,y:byte;texto:string);
procedure escribenorln(x,y:byte;texto:string);
function leerealxy(x,y:byte):real;
procedure escribeinv(r:byte);
procedure escribenor(r:byte);
Procedure haceventana(r:byte);
Procedure limpiaventana(r:byte);
procedure inicia_pantalla;

implementation
PROCEDURE PonCursor(NuevoCursor:INTEGER);
   { *** Pone el Cursor a NuevoCursor ************** }
   { ** Pone $2000-1=nocursor, $2000=ok, 15=lleno ** }
   { ********************************************** }
  VAR
    Reg : Registers;
BEGIN
  WITH Reg DO
  BEGIN
    AH := 1;
    BH := 0;
    CX := NuevoCursor;
    Intr($10, Reg);
```