```
(* * * * * * * * * * * * * * * * * * * * * * * * * * * * * *)
(* * * * * * * * * * * * * * * * * * * * * * * * * * * * * *)
(* *                                                       * *)
(* *  UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO              * *)
(* *                                                       * *)
(* *          FACULTAD DE INGENIERIA                       * *)
(* *                                                       * *)
(* *             CONTROL  DIGITAL                          * *)
(* *                                                       * *)
(* *                                                       * *)
(* *       DISE¥O:                                         * *)
(* *            & Rafael Ramos Aguilar.                    * *)
(* *            % Alejandro Florez Alvarez.                * *)
(* *            # Andres Gomez de Silva Garza.             * *)
(* *                                                       * *)
(* *                  (&) MEXICO D.F.SEPT/1988             * *)
(* *                  (%) Mexico D.F.Sept/1989             * *)
(* *                  (#) Mexico D.F.Oct /1989             * *)
(* * * * * * * * * * * * * * * * * * * * * * * * * * * * * *)
(* * * * * * * * * * * * * * * * * * * * * * * * * * * * * *)
```

```
  {                 Gram tica                          }
  {                                                    }
  {        R0: <POLIN'> --> <POLIN>                    }
  {        R1: <POLIN>  --> <POL>                      }
  {        R2: <POLIN>  --> -<POL>                     }
  {        R3: <POL>    --> <POL><ADDOP><TERM>         }
  {        R4: <POL>    --> <TERM>                     }
  {        R5: <TERM>   --> <CONST><ZETA>              }
  {        R6: <TERM>   --> <CONST>                    }
  {        R7: <TERM>   --> <ZETA>                     }
  {        R8: <ZETA>   --> Z                          }
  {        R9: <ZETA>   --> Z ^ Entero                 }
  {       R10: <CONST>  --> Real                       }
  {       R11: <CONST>  --> Entero                     }
  {       R12: <ADDOP>  --> +                          }
  {       R13: <ADDOP>  --> -                          }
```

```
 Program CDZ;

Const
   Numero_formas=3;  { Necesaria definicion para Turbo Formas }
   Pop = $70;
   Inv = $1F;
   Norm = 14;
   Blk  =$8E;
{$I TFor.Sys }       { Turbo Formas }

  Type

     POL_YARRAY=Array[0..1000] of Real;    { Tipo funci¢n a
```

```
graficar }

     POL_ARRAY=Array[0..100] of Real;      { Tipo polinomio }

     SEM_REG =Record                       { Registro para
an lisis-s¡ntesis  }
                                           {  de polinomios de
entrada        }
               N_ENTERO:Integer;
               N_FLOAT :Real;
               ESTADO  :Byte;
             End;

     LST80 = String[80];                   { Tipo string de
entrada }

     Arch  = Record
                 Desc : String[78];
                 X    : Pol_Array;
                 O    : Byte;
                 V    : Pol_Array;
                 R    : Byte;
                 Q    : Pol_Array;
                 M    : Byte;
                 P    : Pol_Array;
                 N    : Byte;
                 Y    : Pol_YArray;
                 Lims : Integer;
             End{Record};
     NombresD    = Array [1..12] of Char;
     Directorios = Array [1..50] of NombresD;


   {EndType}

  Var
     V,                                  { V(Z) }
     X,                                  { X(Z) }  { U(Z) = X(Z)/V
(Z) }
     P,                                  { P(Z) }
     Q     :POL_ARRAY;                   { Q(Z) }  { G(Z) = Q(Z)/P
(Z) }
     Y     :POL_YARRAY;                  { Y(k) }  { Y(Z) = G(Z)*U
(Z) }
     M,                                  { Orden de Q(Z) }
     N,                                  { Orden de P(Z) }
     O,                                  { Orden de X(Z) }
     R,                                  { Orden de V(Z) }
     SP    :Byte;                        { Apuntador de Stack }
     STACK :Array[0..10] of SEM_REG;     { Stack para an lisis-s
¡ntesis }
     LIMS  :Integer;                     { L¡mite superior de
```

```
graficaci¢n }
     MAX,                                { Valores m ximo y
}
     MIN   :Real;                        {  m¡nimo de Y(k)
}
     C     :Char;                        { Buffer de un caracter
}
     ERRROR:Boolean;                     { Error }
     Error1:Boolean;
     I,J   :INTEGER;
     Quit  :Boolean;
     Count      : Integer;
     Directorio : Directorios;
     NombreD    : NombresD;
     Opcion     : Char;
     DatosYa    : Boolean;
     Archivo    : File Of Arch;
     Registro   : Arch;


Procedure VidInv;
Begin
   TextColor(15);
   TextBackGround(1);
End;

Procedure VidBlink;
Begin
   TextColor(15+Blink);
   TextBackGround(4);
End;

Procedure VidNorm;
Begin
   TextColor(14);
   TextBackGround(0);
End;

Procedure VidPop;
Begin
   TextColor(0);
   TextBackGround(7);
End;

Procedure Beep;
Begin
   Sound(600);
   Delay(50);
   NoSound;
End;
```

```
  {$I LEE.CDZ }
                { Archivo que contiene la rutina }
                {    especial de lectura.        }


                { Lee siguiente caracter de teclado }

  Function GETCHAR(CADENA:LST80;Var NUM:Byte):Char;
    Begin
      If NUM <49 Then
        Begin
         C:=CADENA[NUM];
         NUM:=NUM+1;
         GETCHAR:=UpCase(C);
        End
      Else
         GETCHAR:='#'
      {Endif};
    End{GETCHAR};

                { Procedimiento general de errores }

  Procedure ERROR(I:Byte { Error a reportar });
    Begin
      GotoXY(1,24);
      VidBlink;
      ClrEol;
      Case I of
          1:Write('Error:S¡mbolo inv lido');
          2:Write('Error:Sintaxis incorrecta');
      End{Case};
      Sound(600);
      Delay(50);
      NoSound;
      VidInv;
    End{ERROR};

                { Analizador l,xico( Scanner ) }

  Procedure LEX_AN(Var TOKEN  :Byte     { Token encontrado  }  ;
                   Var NUM_INT:Integer  { Signo o entero(?) }  ;
                   Var NUMERO :Real     { Constante(?)      }  ;
                   Var P_CHAR :Byte     { Apuntador de caracter
};
                       CADENA :LST80    { String a analizar } );
  {$I LEX.CDZ }
                { Archivo que contiene:
}
                { SCAN[0..8,1..7] = Matriz del aut$mata l,xico
}

  Var
```

```
      EDO,                    { Estado del aut¢mata }
      COL    :Byte;           { Columna en la matriz del aut¢mata
}
      F      :Real;           { Variables de "trabajo" }

                  { Obtiene la columna en el aut¢mata l,xico }
                  {   del caracter que tiene por argumento   }

  Function COLUMN(C:Char):Byte;

     Begin
       Case C of
            'Z' : COLUMN:=1;
            '.' : COLUMN:=3;
            '+' : COLUMN:=4;
            '-' : COLUMN:=5;
            '^' : COLUMN:=6;
            ' ' : COLUMN:=7;
            '#' : COLUMN:=8;
          Else
             If C in DIGITO then
                COLUMN:=2;
        End{Case};
     End{COLUMN};
                                { LEX_AN }
    Begin
      F:=10;
      NUMERO:=0;
      EDO:=0;
      COL:=COLUMN(C);
      While SCAN[EDO,COL] <> 10 Do
        Begin
          EDO:=SCAN[EDO,COL];
          If EDO= 2 Then                                  { Entero
}
              NUMERO:=NUMERO*10+Ord(C)-Ord('0')
          Else
            If EDO=3 Then                                 { Real
}
              Begin
                NUMERO:=NUMERO +(Ord(C)-Ord('0'))/F;
                F:=F*10;
              End
          {Endif};
          C:=GETCHAR(CADENA,P_CHAR);
          COL:=COLUMN(C);
        End
       {Endwhile};
       TOKEN:=EDO;
       Case EDO of
            0,8:TOKEN:=0;                                 { Token
inv lido }
```

```
             2:If NUMERO<=$7FFF then
                 NUM_INT:=Trunc(NUMERO)                  { Token
es entero }
               Else
                 TOKEN:=3                                { Integer
overflow, }
               {Endif};                                  { el
token es real  }
             4:NUM_INT:=1;
             5:NUM_INT:=-1;
       End{Case};
    End{LEX_AN};

                 { Analizador sem ntico }

  Procedure SEMANTICS(     REGLA  :Byte      { Regla a reducir
}  ;
                       Var VECTOR :POL_ARRAY { Polinomio
}  ;
                       Var MAX    :Byte      { Orden de vector
}  ;
                       Var N_INT  :Integer   { Campo entero de
STACK }  ;
                       Var N_REAL :Real      { Campo real de
STACK   }) ;

    Var
      INDICE:Byte;
    Begin
      INDICE:=STACK[SP].N_ENTERO;
      Case REGLA of
          1: VECTOR[INDICE]:=VECTOR[INDICE] + STACK[SP].N_FLOAT;
          2: VECTOR[INDICE]:=VECTOR[INDICE] - STACK[SP].N_FLOAT;
          3: Begin
               VECTOR[INDICE]:=VECTOR[INDICE] + STACK
[SP].N_FLOAT*
                                                 STACK[SP-
1].N_ENTERO;
               N_INT:=STACK[SP-2].N_ENTERO;
               N_REAL:=STACK[SP-2].N_FLOAT;
             End;
          5: Begin
               N_INT:=STACK[SP].N_ENTERO;
               N_REAL:=STACK[SP-1].N_FLOAT;
             End;
          6: Begin
               N_REAL:=STACK[SP].N_FLOAT;
               N_INT:=0;
             End;
          7: Begin
               N_REAL:=1;
               N_INT:=STACK[SP].N_ENTERO;
```

```
            End;
          8: Begin
               N_INT:=1;
               If MAX=0 then
                 MAX:=1;
            End;
          9: Begin
               N_INT:=STACK[SP].N_ENTERO;
               If N_INT> MAX then
                 MAX:=N_INT;
            End;
         12: N_INT:=1;
         13: N_INT:=-1;
       Else
         Begin
           N_INT:=STACK[SP].N_ENTERO;
           N_REAL:=STACK[SP].N_FLOAT;
         End
    End{case};
  End{SEMANTICS};



                  { Procedimiento de An lisis-S¡ntesis (Parser) }

  Procedure SYNTAX_AN(Var POL   :POL_ARRAY  { Polinomio
resultante } ;
                      Var ORDEN :Byte       { Orden del polinomio
} ;
                          CADENA:LST80      { String a analizar
} ;
                      Var ERRROR:Boolean    { Bandera de error
} );
     {$I SYX.CDZ }
                  { Archivo que contiene:
}
                  { PAR_MAT [0..17,1..13] tabla de an lisis
sint ctico        }
                  { VEC_PROD [1.. 2,1..13] tabla con las reglas
de producci¢n }

    Var
      TOKEN,
      I,
      REGLA,
      P_CHAR  :Byte;
      W       :Integer;
      N_INT   :Integer;
      N_REAL  :Real;
```

```
                { Quita N elementos del STACK }

    Procedure POP(N:Byte);
      Begin
        SP:=SP-N;
      End{POP};

                 { Agrega un registro al STACK }

    Procedure PUSH( EDO    :Byte    { Sintaxis };
                    N_INT  :Integer { Campo entero};
                    N_REAL :Real    { Campo real});
      Begin
        SP:=SP+1;
        With STACK[SP] do
          Begin
            ESTADO:=EDO;
            N_ENTERO:=N_INT;
            N_FLOAT:=N_REAL;
          End{With};
      End{PUSH};

               { SYNTAX_AN }
Begin
    ORDEN:=0;
    ERRROR:=False;
    For I:=0 to 100 do
      POL[I]:=0
    {Endfor};
    SP:=-1;
    PUSH(0,0,0);
    P_CHAR:=1;
    REGLA:=1;
    C:=GETCHAR(CADENA,P_CHAR);
    LEX_AN(TOKEN,N_INT,N_REAL,P_CHAR,CADENA);
    Repeat
      If TOKEN=0 then
        Begin
          ERROR(1);
          ERRROR:=True;
          REGLA:=0;
        End
      Else
        Begin
          W:=PAR_MAT[STACK[SP].ESTADO,TOKEN];
          Case W of

             1..17 :Begin                              { Shift }
                      PUSH(W,N_INT,N_REAL);
                      LEX_AN(TOKEN,N_INT,N_REAL,P_CHAR,CADENA);
                    End;
```

```
              -13..-1:Begin                                { Reduce }
                        REGLA:=-W;
                        SEMANTICS(REGLA,POL,ORDEN,N_INT,N_REAL);
                        POP(VEC_PROD[NO_SYM,REGLA]);
                        PUSH(PAR_MAT[STACK[SP].ESTADO,VEC_PROD
[COL,REGLA]],

N_INT,N_REAL);
                      End;
                                                             { Cadena
v lida }
                       0:Begin
                          Vidinv;
                          REGLA:=0;
                          GotoXY(1,24);ClrEol;

                        End;

                      99:Begin                               { Error
sint ctico }
                          ERROR(2);
                          ERRROR:=true;
                          REGLA:=0;
                        End;
                 End{Case};
            End{Else};
          Until
            REGLA=0;
       End{SYNTAX_AN};

                  { Lectura y an lisis de funciones de
transferencia }
                  {              en el dominio de Z
}

  Procedure LECTOR(Var Q:POL_ARRAY { Polinomio del numerador   }
;
                   Var M:Byte      { Orden de Q(Z)             }
;
                   Var P:POL_ARRAY { Polinomio del denominador }
;
                   Var N:Byte      { Orden de P(Z)             }
;
                   Var Error1:Boolean
);

    Var
       ERRROR     : Boolean;
       Salir      : Boolean;
       Ind        : Integer;
       D_I_STRING,
       N_I_STRING :LST80;
```

```
                { Verifica que: POL[I]<>0       }
                { Es s¢lo para usuarios ma¤osos }

    Procedure POL_CHECK(Var POL:POL_ARRAY { Polinomio a checar }
;
                        Var I:Byte        { Orden de POL(Z)    }
);

      Begin
        ERRROR:=False;
        While (POL[I]=0) and Not Errror do
          If I>0 then
            I:=I-1
          Else
              ERRROR:=True
          {End IF};
        {Endwhile};
      End {POL_CHECK};

                      { LECTOR }
    Begin
      PutStr
('ÚÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÂÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ
ÄÄÄÄÄÄÄÄÄÄÄÄÄÄ¿',4,0,Inv);
      PutStr('³Polinomio del numerador   :³
³',5,0,Inv);
      PutStr
('ÃÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÅÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ
ÄÄÄÄÄÄÄÄÄÄÄÄÄÄ´',6,0,Inv);
      PutStr('³Polinomio del denominador :³
³',7,0,Inv);
      PutStr
('ÀÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÁÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ
ÄÄÄÄÄÄÄÄÄÄÄÄÄÄÙ',8,0,Inv);

      ERRROR:=False;
      Salir:=False;
      N_I_STRING:='
';
      D_I_STRING:='
';
      Ind:=1;
      Repeat
         Case Ind Of
          1 :Begin
             Repeat
                N_I_STRING:=LSTR(N_I_STRING,6,30,48,Salir);
                IF Not Salir Then
                   Begin
                   SYNTAX_AN(Q,M,N_I_STRING,ERRROR);
                   POL_CHECK(Q,M);
```

```
                 End
              {endIf};
           Until (Not ERRROR) Or Salir;
           If Salir Then Ind:=4 else ind:=2;
           End;
        2 :Begin
           Repeat
              D_I_STRING:=LSTR(D_I_STRING,8,30,48,Salir);
              If Not Salir Then
                 Begin
                 SYNTAX_AN(P,N,D_I_STRING,ERRROR);
                 POL_CHECK(P,N);
                 End
              {end If};
           Until (ERRROR=False) Or salir;
           If Salir Then Ind:=1 else ind:=3;
           end;
        3 : Begin
            If N<M then                  { Checa realizabilidad
f¡sica }
              Begin
                Ind:=1;
                GOTOXY(1,24);
                VidBlink;
                Writeln('Esta funci¢n de transferencia no es
realizable. Int‚ntalo otra vez ');
                VidInv;
              End
            else
               ind:=4;
            {Endif};
            End;
       End{Case};
    Until Ind=4;
    Error1:=Salir;

  End{LECTOR};

                { Lectura de la se¤al de entrada }

 Procedure In_put;
   Var
     OPCION :Char;
     W,
     C      :Real;
     I,
     J      : Integer;

   Begin
     PutStr('ÚÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ¿',11,24,Inv);
     PutStr('³Se¤al de entrada :       ³',12,24,Inv);
     PutStr('ÃÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ´',13,24,Inv);
```

```
PutStr('³1.- Pulso       C ë(k)   ³',14,24,Inv);
PutStr('³2.- Escalon     C æ(k)   ³',15,24,Inv);
PutStr('³3.- Seno        C Sen(wk)³',16,24,Inv);
PutStr('³4.- Coseno      C Cos(wk)³',17,24,Inv);
PutStr('³5.- Otra        P(z)/Q(z)³',18,24,Inv);
PutStr('ÀÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÙ',19,24,Inv);
CurOff;
I:=1;
VidPop;
GotoXY(26,15);Write('1.- Pulso       C ë(k)   ');
Repeat
   J:=I;
   Read(Kbd,Opcion);
   Case Opcion of
      #27 : If KeyPressed Then
               Begin
               Read(Kbd,Opcion);
               Case Opcion of
                'H','K' : If I=1 Then
                             i:=5
                          Else
                             i:=i-1
                          {EndIf};


                'P','M' : If I=5 Then
                             i:=1
                          Else
                             i:=i+1
                          {EndIf};
                'G': i:=1;
                'O': i:=5;
               End{Case};
               VidInv;
               GotoXY(26,14+J);
               Case J of
                 1 : Write('1.- Pulso       C ë(k)   ');
                 2 : Write('2.- Escalon     C æ(k)   ');
                 3 : Write('3.- Seno        C Sen(wk)');
                 4 : Write('4.- Coseno      C Cos(wk)');
                 5 : Write('5.- Otra        P(z)/Q(z)');
               End;

               GotoXY(26,14+I);
               VidPop;
               Case I of
                 1 : Write('1.- Pulso       C ë(k)   ');
                 2 : Write('2.- Escalon     C æ(k)   ');
                 3 : Write('3.- Seno        C Sen(wk)');
                 4 : Write('4.- Coseno      C Cos(wk)');
                 5 : Write('5.- Otra        P(z)/Q(z)');
               End;
```

```
                  End
                Else
                  Opcion:='S'
                {EndIf};
           'S' : Opcion:='X';
           #13 : Opcion:=Chr(48+i);
        End{Case};

    Until Opcion In['1'..'5'];
    CurOn;
    VidInv;

    GotoXY(25,5);
    Case OPCION of
        '1': Begin                                  { Pulso  }
               X[0]:=1;
               repeat
                 GotoXY(1,24);
                 Write('Valor de la constante [Def=1]:');
                 Clreol;
                 {$I-} Read(X[0]); {$I+}
               Until
                 (IOResult=0) And (X[0]<>0);
               GotoXy(1,24);ClrEol;
               V[0]:=1;
               O:=0;
               R:=0;
             End;
        '2': Begin                                  { Escal¢n }
               X[1]:=1;
               repeat
                 GotoXY(1,24);
                 Write('Valor de la constante [Def=1]:');
                 Clreol;
                 {$I-} Read(X[1]); {$I+}
               Until
                 (IOResult=0) And (X[1]<>0);
               GotoXY(1,24);Clreol;
               X[0]:=0;
               O:=1;
               V[0]:=-1;
               V[1]:=1;
               R:=1;
             End;
    '3','4': Begin                                  { Seno y
coseno }
               repeat
                 GotoXY(1,24);
                 Write('Valor de la constante:');
                 Clreol;
                 {$I-} Read(C); {$I+}
```

```
                 Until
                   IOResult=0;
                 GotoXy(1,24);ClrEol;
                 repeat
                   GotoXY(1, 24);
                   Write('Velocidad angular (Radianes):');
                   Clreol;
                   {$I-} Read(W); {$I+}
                 Until
                   IOResult=0;
                 GotoXy(1,24);ClrEol;
                 If OPCION='3' then
                   Begin
                     X[0]:=0;
                     X[1]:=C*Sin(W);
                     O:=1;
                   End
                 Else
                   Begin
                     X[0]:=0;
                     X[1]:=-C*Cos(W);
                     X[2]:=C;
                     O:=2;
                   End
                 {Endif};
                 V[0]:=1;
                 V[1]:=-2*Cos(W);
                 V[2]:=1;
                 R:=2;
               End;
          '5': Begin                                     { Otra
}
               PutStr
('ÚÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ¿',0,14,Inv);
               PutStr('³Exprese la se¤al de entrada en el dominio
de Z ³',1,14,Inv);
               PutStr
('ÀÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÙ',2,14,Inv);
               LECTOR(X,O,V,R,Error1);
               End;
        End{Case};
      End{IN_PUT};
                        { Multiplica dos polinomios }

  Procedure MULTICS(Var H:POL_ARRAY { H(Z)<--H(Z) * U(Z) } ;
                    Var N:Byte      { Orden de H(Z)      } ;
                        U:POL_ARRAY { U(Z)               } ;
                        M:Byte      { Orden de U(Z)      };
                        W:POL_ARRAY { H(z) }              );
    Var
      I,J:Integer;
```

```
    Begin
      For I:=0 to M+N do                 { Inicializaci¢n }
         H[I]:=0
      {Endfor};
      For I:=0 to M do
        For J:=0 to N do
           H[I+J]:=H[I+J]+W[J]*U[I]
        {Endfor}
      {Endfor};
      N:=N+M;

    End{MULTICS};

                      { Divide dos polinomios }
                      {     Obtiene Y(k)      }
                      {       Y <-- Q/P       }

  Procedure DIVICS;{   Y:POL_ARRAYY( Polinomio a graficar )
                       J:Integer   ( Elementos de Y(k)    )
                       Q:POL_ARRAY ( Polinomio dividendo  )
                       M:Integer   ( Orden de Q           )
                       P:POL_ARRAY ( Polinomio divisor    )
                       N:Integer   ( Orden de P           )  }
    Var
     K,J,I:Integer;
     POL  :POL_YARRAY;
    Begin

      MAX:=0;
      MIN:=0;
      For k:=m+1 to 100 do
         Q[k]:=0
      {end if};
      For K:=0 to LIMS do        { Inicializaci¢n }
        Begin
         Y[k]:=0;
         POL[K]:=0;

        End
      {Endfor};
      For I:=0 to M div 2 do
        Begin
          POL[M-I]:=Q[I];        { Intercambia a los coeficientes
del }
          POL[I]:=Q[M-I];        { polinomio dividendo para
realizar  }
        End                      {    el algoritmo de la divisi¢n
}
      {Endfor};
      J:=0;
      For I:=N-M to LIMS do
        Begin
```

```
          Y[I]:=POL[J]/P[N];
          If Y[I]>MAX then
            MAX:=Y[I]
          Else
            If Y[I]<MIN then
              MIN:=Y[I]
            {Endif}
          {Endif};
          GOTOXY (25,15);Write(I:5);Write(' ',Y[I]:20);
          For K:=0 to N do
             POL[K+J]:=POL[K+J]-Y[I]*P[N-K];
          {Endfor};
          J:=J+1;
        End
    End{DIVICS};


{*****************************************}
{***  Procedimiento de Despliegue de los ***}
{***  datos de la grafica en pantalla.   ***}
{*****************************************}
Procedure Memor(Limite:Integer);
     VAR
          n,
          I,
          J,
          K,
          L          : INTEGER;
          TOR        : CHAR;
{   Sub procedimiento   }
PROCEDURE UP;
BEGIN
     IF I>0 THEN
          BEGIN
          IF J=1 THEN
                BEGIN
                GOTOXY(1,J);
                WRITE(i:7,'³',Y[i]:20);
                INSLINE;
                GOTOXY(1,J);
                I:=I-1;
                VidPop;
                WRITE(i:7,'³',Y[i]:20);
                VidInv;
                END
          ELSE
                BEGIN
                GOTOXY(1,J);
                WRITE(i:7,'³',Y[i]:20);
                J:=J-1;
                I:=I-1;
                GOTOXY(1,J);
```

```
                    VidPop;
                    WRITE(i:7,'³',Y[i]:20);
                    VidInv;
                    END
              {END IF};
              END
        ELSE
              BEEP
        {END IF};

END;
{   Sub procedimiento   }
PROCEDURE DAWN;
BEGIN
        IF I<K THEN
              BEGIN
              IF J=10 THEN
                    BEGIN
                    GOTOXY(1,J);
                    WRITE(i:7,'³',Y[i]:20);
                    GOTOXY(1,1);
                    DELLINE;
                    GOTOXY(1,J);
                    I:=I+1;
                    VidPop;
                    WRITE(i:7,'³',Y[i]:20);
                    VidInv;
                    END
              ELSE
                    BEGIN
                    GOTOXY(1,J);
                    WRITE(i:7,'³',Y[i]:20);
                    J:=J+1;
                    I:=I+1;
                    GOTOXY(1,J);
                    VidPop;
                    WRITE(i:7,'³',Y[i]:20);
                    VidInv;
                    END
              {END IF};
              END
        ELSE
              BEEP
        {END IF};

END;
{**********************************************************}
BEGIN
        SaveScr;
        CurOff;
        k:=Limite;
        PutStr('ÚÄÄÄÄÄÄÄÂÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ¿', 8,24,Inv);
```

```
      PutStr('³   k   ³          Y(k)         ³', 9,24,Inv);
      PutStr('ÃÄÄÄÄÄÄÄÄÅÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ´',10,24,Inv);
      PutStr('³       ³                       ³',11,24,Inv);
      PutStr('³       ³                       ³',12,24,Inv);
      PutStr('³       ³                       ³',13,24,Inv);
      PutStr('³       ³                       ³',14,24,Inv);
      PutStr('³       ³                       ³',15,24,Inv);
      PutStr('³       ³                       ³',16,24,Inv);
      PutStr('³       ³                       ³',17,24,Inv);
      PutStr('³       ³                       ³',18,24,Inv);
      PutStr('³       ³                       ³',19,24,Inv);
      PutStr('³       ³                       ³',20,24,Inv);
      PutStr('ÀÄÄÄÄÄÄÄÄÁÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÙ',21,24,Inv);
      WINDOW(26,12,54,21);
      I:=-1;
      J:=1;
      REPEAT
            I:=I+1;
            GOTOXY(1,I+1);
            WRITE(i:7,'³',Y[i]:20);
      UNTIL (I=9) OR (I=K);
      I:=0;
      GOTOXY(1,J);
      VidPop;
      WRITE(i:7,'³',Y[i]:20);
      Vidinv;
      REPEAT
            READ(KBD,TOR);
            IF (TOR=#27) And KeyPressed THEN
                  BEGIN
                  READ(KBD,TOR);
                  CASE TOR OF
                        'H': UP;
                        'I': for n:=1 to 10 do UP;
                        'P': DAWN;
                        'Q': for n:=1 to 10 do DAWN;
                  ELSE
                        BEEP;
                  END{CASE};
                  END
            ELSE
                  BEEP
            {END IF};
      UNTIL (TOR=#13) Or (Tor=#27);
      WINDOW(1,1,80,25);
      CurOn;
      LoadScr;
END;


Procedure GraficaDatos(CoeficientesResultado : POL_YARRAY;
```

```
                        NumeroDeIteraciones   : Integer);
var
    Opcion,
    Ch                : Char;
    ValorMinimo,
    ValorMaximo       :Real;
    I,J,
    Code,
    TipoDeDesplegado,
    NivelHorizontal,
    DesplHorizontal,
    DesplVert,
    MemHor,
    MemVer,
    PuntosAGraficar,
    Aux               : Integer;
    RangoVertical,
    RelacionMaximoATotal,
    ValorMedio        : Real;
    NumEnCaracteres   : string[15];
begin
    ClrScr;
    Writeln('NOTA: El eje vertical aparecera ligeramente hacia la
izquierda del lugar sobre');
    Writeln('el eje horizontal correspondiente a k = 0.  El eje
horizontal aparecera donde');
    Writeln('la amplitud de la solucion valga cero.  Para cada
lectura de la grafica, el');
    Writeln('eje horizontal representa el valor de k (el numero
de la muestra) y el eje');
    Writeln('vertical representa el valor que toma la funcion en
cada k.');


      PutStr('ÚÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ¿',11,24,Inv);
      PutStr('³Tipo de Graficacion:     ³',12,24,Inv);
      PutStr('ÃÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ´',13,24,Inv);
      PutStr('³1.- Impulsos.            ³',14,24,Inv);
      PutStr('³2.- Puntos.              ³',15,24,Inv);
      PutStr('³3.- Puntos Unidos.       ³',16,24,Inv);
      PutStr('ÀÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÙ',17,24,Inv);
      CurOff;
      I:=1;
      VidPop;
      GotoXY(26,15);Write('1.- Impulsos.           ');
      Repeat
         J:=I;
         Read(Kbd,Opcion);
         Case Opcion of
            #27 : If KeyPressed Then
                     Begin
                     Read(Kbd,Opcion);
```

```
                  Case Opcion of
                   'H','K' : If I=1 Then
                                i:=3
                             Else
                                i:=i-1
                             {EndIf};


                   'P','M' : If I=3 Then
                                i:=1
                             Else
                                i:=i+1
                             {EndIf};
                   'G': i:=1;
                   'O': i:=3;
                  End{Case};
                  VidInv;
                  GotoXY(26,14+J);
                  Case J of
                    1 : Write('1.- Impulsos.             ');
                    2 : Write('2.- Puntos.               ');
                    3 : Write('3.- Puntos Unidos.        ');
                  End;

                  GotoXY(26,14+I);
                  VidPop;
                  Case I of
                    1 : Write('1.- Impulsos.             ');
                    2 : Write('2.- Puntos.               ');
                    3 : Write('3.- Puntos Unidos.        ');
                  End;

                  End
               Else
                  Opcion:='S'
               {EndIf};
          'S' : Opcion:='X';
          #13 : Opcion:=Chr(48+i);
       End{Case};

     Until Opcion In['1'..'3'];

     CurOn;
     VidInv;
     TipoDeDesplegado:=Ord(Opcion)-48;

    ClrScr;

    GotoXY(1,22);
    Write('Nota : Minimo 5 , Maximo ',NumeroDeIteraciones);
    PutStr
('ÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ
```

```
ÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ',22,0,Inv);
    PutStr('
',23,0,Inv);
    PutStr
('ÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ
ÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ',24,0,Inv);

    Repeat
        GotoXy(1,24);ClrEoL;
        Write('Cuantos datos se Grafican
<Def=',NumeroDeIteraciones,'>:  ');
        PuntosAGraficar:=NumeroDeIteraciones;
        {$I-} Read(PuntosAGraficar); {$I+}
    Until (IOResult=0) And
          (PuntosAGraficar>4) And
          (PuntosAGraficar<=NumeroDeIteraciones);

    ValorMaximo:=0;
    ValorMinimo:=0;
    For i:=1 to PuntosAGraficar-1 do
    begin
        If (CoeficientesResultado[i]>ValorMaximo) Then
            ValorMaximo:=CoeficientesResultado[i];
        If (CoeficientesResultado[i]<ValorMinimo) Then
            ValorMinimo:=CoeficientesResultado[i];
    end;
    RangoVertical:=ValorMaximo-ValorMinimo;
    RelacionMaximoATotal:=ValorMaximo/RangoVertical;
    NivelHorizontal:=trunc((175*RelacionMaximoATotal)+5);
    ValorMedio:=(RangoVertical/2)+ValorMinimo;
    HiRes;
    Draw(47,5,47,180,White);
    Draw(47,NivelHorizontal,630,NivelHorizontal,White);
    For i:=1 to PuntosAGraficar do
    begin
        DesplHorizontal:=trunc(i*(583/PuntosAGraficar))+49;
        DesplVert:=NivelHorizontal-trunc(CoeficientesResultado[i]
*(175/RangoVertical));
        Case TipoDeDesplegado Of
           1 : Draw
(DesplHorizontal,DesplVert,DesplHorizontal,NivelHorizontal,White)
;
           2 : Plot(DesplHorizontal,DesplVert,White);
           3 : if I>1 Then Draw
(MemHor,MemVer,DesplHorizontal,DesplVert,White);
        End{Case};
        MemHor:=DesplHorizontal;
        MemVer:=DesplVert;
    end;
    Window(1,1,5,23);
    GotoXY(1,1);
    Write(ValorMaximo:9);
```

```
    GotoXY(1,6);
    Write(ValorMedio+(RangoVertical/4):9);
    GotoXY(1,11);
    Write(ValorMedio:9);
    GotoXY(1,17);
    Write(ValorMedio-(RangoVertical/4):9);
    GotoXY(1,22);
    Write(ValorMinimo:9);
    Window(7,24,80,25);
    GotoXY(1,1);
    Write('0');
    GotoXY(16,1);
    If (((trunc(PuntosAGraficar/4)) div 100) = 0) Then
        Write(' ');
    If (((trunc(PuntosAGraficar/4)) div 10) = 0) Then
        Write(' ');
    Write(trunc(PuntosAGraficar/4));
    GotoXY(35,1);
    If (((trunc(PuntosAGraficar/2)) div 100) = 0) Then
        Write(' ');
    If (((trunc(PuntosAGraficar/2)) div 10) = 0) Then
        Write(' ');
    Write(trunc(PuntosAGraficar/2));
    GotoXY(53,1);
    If (((trunc(PuntosAGraficar*3/4)) div 100) = 0) Then
        Write(' ');
    If (((trunc(PuntosAGraficar*3/4)) div 10) = 0) Then
        Write(' ');
    Write(trunc(PuntosAGraficar*3/4));
    GotoXY(71,1);
    If (((PuntosAGraficar) div 100) = 0) Then
        Write(' ');
    If (((PuntosAGraficar) div 10) =0) Then
        Write(' ');
    Write(PuntosAGraficar);
    GotoXY(16,2);
    Write('Apriete cualquier tecla para continuar.');
    Repeat
    Until Keypressed;
    Window(1,1,80,25);
    TextMode(C80);
end;



Procedure Graphics;
Var
   I,
   J : Byte;
   Opcion : Char;
Begin
```

```
ClrScr;
Repeat
   PutStr('ÚÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ¿',11,24,Inv);
   PutStr('³An lisis de Datos :       ³',12,24,Inv);
   PutStr('ÃÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ´',13,24,Inv);
   PutStr('³1.- Graficaci¢n            ³',14,24,Inv);
   PutStr('³2.- Tabla de Datos         ³',15,24,Inv);
   PutStr('ÀÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÙ',16,24,Inv);
   CurOff;
   I:=1;
   VidPop;
   GotoXY(26,15);Write('1.- Graficaci¢n          ');
   Repeat
      J:=I;
      Read(Kbd,Opcion);
      Case Opcion of
         #27 : If KeyPressed Then
                  Begin
                  Read(Kbd,Opcion);
                  Case Opcion of
                   'H','K' : If I=1 Then
                                i:=2
                             Else
                                i:=i-1
                             {EndIf};


                   'P','M' : If I=2 Then
                                i:=1
                             Else
                                i:=i+1
                             {EndIf};
                   'G': i:=1;
                   'O': i:=2;
                  End{Case};
                  VidInv;
                  GotoXY(26,14+J);
                  Case J of
                    1 : Write('1.- Graficaci¢n          ');
                    2 : Write('2.- Tabla de Datos       ');
                  End;

                  GotoXY(26,14+I);
                  VidPop;
                  Case I of
                    1 : Write('1.- Graficaci¢n          ');
                    2 : Write('2.- Tabla de Datos       ');
                  End;

                  End
               Else
                  Opcion:='S'
```

```
                {EndIf};
            'S' : Opcion:='X';
            #13 : Opcion:=Chr(48+i);
         End{Case};

      Until Opcion In['1'..'2','S'];

      CurOn;
      VidInv;
      Case Opcion of
        '1' : Begin
              SaveScr;
              ClrScr;
              GraficaDatos(Y,LIMS);
              Repeat Until KeyPressed;
              LoadScr;
              End;
        '2' : MEMOR(lims);
      End{case};
   Until Opcion='S';
End;


Procedure Dir(Var Directorio : Directorios;
              Var Count      : Integer);
Const
   Attribute = $01;
   Blank     = ' ';
   SetDTA    = $1a;
   FindFirst = $4e;
   FindNext  = $4f;

Type
   Registers =
      RECORD Case Boolean of
       True  : ( al,ah,bl,bh,cl,ch,dl,dh : Byte);
       False : ( AX   ,BX   ,CX   ,DX   ,
                 BP,SI,DI,DS,ES,Flags    : Integer);
      End;

Var
   DTA    : Array [0..127] of Byte;
   Reg    : Registers;
   AsciiZ : String[65];

Procedure OneEntry;
Var
   i, j : Integer;
Begin
   MsDos(reg);
   If Reg.al=0 then
      Begin
```

```
      Count:=Count+1;
      j:=30;
      For i:=1 to 12 do
         If DTA[j]=0 Then
            NombreD[I]:=Blank
         Else
            Begin
            NombreD[I]:=Chr(DTA[j]);
            j:=j+1
            End;
      End
   {end If};
   Directorio[Count]:=NombreD;
End;

Procedure GetArg;
Begin
   BufLen:=64;
   AsciiZ:='';
   If (Length(AsciiZ) = 0) or
      (AsciiZ[Length(AsciiZ)]='\')
      Then AsciiZ:='A:\*.ATZ';
   AsciiZ:=AsciiZ+Chr(0);
End;

Procedure Initialize;
Begin
   Count:=0;
   GetArg;
{   Writeln('Directorio :',AsciiZ);   }
   With reg do
      Begin
      ah:=SetDTA;
      ds:=Seg(DTA);
      dx:=Ofs(DTA);
      MsDos(Reg);
      ah:=FindFirst;
      al:=0;
      cx:=Attribute;
      ds:=Seg(AsciiZ);
      dx:=Ofs(AsciiZ[1]);
      end
End;

Begin
   Initialize;
   OneEntry;
   With reg do While al=0 do
      Begin
      ah:=FindNext;
      OneEntry;
      End;
```

```
End;


Procedure DatosNuevos;
Begin
     DatosYa:=True;
     VidNorm;
     ClrScr;
     VidInv;
     PutStr('ÚÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ
¿',0,14,Inv);
     PutStr('³Funci¢n de transferencia del sistema discreto
³',1,14,Inv);
     PutStr
('ÀÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÙ',2,14,Inv);
     PutStr
('ÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ
ÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ',22,0,Inv);
     PutStr('
',23,0,Inv);
     PutStr
('ÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ
ÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ',24,0,Inv);
     LECTOR(Q,M,P,N,Error1);
     If Not Error1 Then
        Begin
        SaveScr;
        IN_PUT;
        LoadScr;
        MULTICS(Q,M,X,O,Q);         { Q(Z)<--Q(Z)*X(Z) }
        MULTICS(P,N,V,R,p);         { P(Z)<--P(Z)*V(Z) }
        Repeat
             ERRROR:=False;
             Repeat
                 GotoXY(1,24);
                 Write('L¡mite superior de graficaci¢n [0..1000]
<Def=',lims,'>:');
                 Clreol;

                 {$I-} Read(LIMS); {$I+}
             Until
                      IOResult=0;
             WriteLn;
             If (LIMS<0)or(LIMS>1000) then
                 ERRROR:=True;
        Until ERRROR=False;
        DIVICS;
        End
     Else
        DatosYa:=False;
     {end If};
```

```
End{DatosNuevos};


Procedure GrabaDatos;
Var
   ArchNom : String[8];
   Ch      : Char;
Begin
   GotoXY(1,24);
   ClrEol;
   Write('Nombre del archivo de Salida :[        ].ATZ');
   GotoXY(32,24);
   BufLen:=8;
   Read(ArchNom);
   BufLen:=78;
   If ArchNom='' Then
      Begin
      GotoXY(1,24);
      ClrEol;
      Exit;
      End
   {EndIf};
   If Exist(ArchNom+'.ATZ') Then
      Begin
      GotoXY(1,24);
      ClrEol;
      Write('Desea reescribir el archivo ',ArchNom,' S/N?');
      Repeat Read(Kbd,Ch); Ch:=Upcase(Ch); Until Ch in['S','N'];
      If Ch='N' Then
         Begin
         GotoXY(1,24);
         ClrEol;
         Exit;
         End
      {EndIf};
      End
   {EndIf};
   GotoXY(1,22);ClrEol;
   Write('Describa sus datos :');
   GotoXY(1,24);ClrEol;
   Read(Registro.Desc);
   Registro.X:=X;
   Registro.O:=O;
   Registro.V:=V;
   Registro.R:=R;
   Registro.Q:=Q;
   Registro.M:=M;
   Registro.P:=P;
   Registro.N:=N;
   Registro.Y:=Y;
   Registro.Lims:=Lims;
```

```
   Assign(Archivo,ArchNom+'.ATZ');
   Rewrite(Archivo);
   Write(Archivo,Registro);
   Close(Archivo);
   GotoXY(1,22);ClrEol;
   GotoXY(1,24);ClrEol;
End;

Procedure CargaDatos;
Type
   CDZStr78 = String[78];
Var
   ArchNom    : String[12];
   Descripstr : Array [1..50] Of CDZStr78;
   FileDesc   : File Of CDZStr78;
{*****************************************}
{***  Procedimiento de Despliegue de los ***}
{***  datos de la grafica en pantalla.   ***}
{*****************************************}
Procedure Directo;
     VAR
           n,
           I,
           J,
           K,
           L          : INTEGER;
           TOR        : CHAR;
{   Sub procedimiento   }
PROCEDURE UP;
BEGIN
     IF I>1 THEN
           BEGIN
           IF J=1 THEN
                 BEGIN
                 GOTOXY(1,J);
                 WRITE(Directorio[i]);
                 INSLINE;
                 GOTOXY(1,J);
                 I:=I-1;
                 VidPop;
                 WRITE(Directorio[i]);
                 VidInv;
                 Window(1,1,80,25);
                 GotoXY(1,24);ClrEol;
                 Write(DescripStr[I]);
                 WINDOW(2,4,14,13);
                 END
           ELSE
                 BEGIN
                 GOTOXY(1,J);
                 WRITE(Directorio[i]);
                 J:=J-1;
```

```
                I:=I-1;
                GOTOXY(1,J);
                VidPop;
                WRITE(Directorio[i]);
                VidInv;
                Window(1,1,80,25);
                GotoXY(1,24);ClrEol;
                Write(DescripStr[I]);
                WINDOW(2,4,14,13);
                END
          {END IF};
          END
     ELSE
          BEEP
     {END IF};

END;
{   Sub procedimiento   }
PROCEDURE DAWN;
BEGIN
     IF I<K THEN
          BEGIN
          IF J=10 THEN
                BEGIN
                GOTOXY(1,J);
                WRITE(Directorio[i]);
                GOTOXY(1,1);
                DELLINE;
                GOTOXY(1,J);
                I:=I+1;
                VidPop;
                WRITE(Directorio[i]);
                VidInv;
                Window(1,1,80,25);
                GotoXY(1,24);ClrEol;
                Write(DescripStr[I]);
                WINDOW(2,4,14,13);
                END
          ELSE
                BEGIN
                GOTOXY(1,J);
                WRITE(Directorio[i]);
                J:=J+1;
                I:=I+1;
                GOTOXY(1,J);
                VidPop;
                WRITE(Directorio[i]);
                VidInv;
                Window(1,1,80,25);
                GotoXY(1,24);ClrEol;
                Write(DescripStr[I]);
                WINDOW(2,4,14,13);
```

```
                END
          {END IF};
          END
     ELSE
          BEEP
     {END IF};

END;
{*********************************************************}
BEGIN
     CurOff;
     k:=Count;
     PutStr('ÚÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ¿', 0, 0,Inv);
     PutStr('³Archivos :   ³', 1, 0,Inv);
     PutStr('ÃÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ´', 2, 0,Inv);
     PutStr('³             ³', 3, 0,Inv);
     PutStr('³             ³', 4, 0,Inv);
     PutStr('³             ³', 5, 0,Inv);
     PutStr('³             ³', 6, 0,Inv);
     PutStr('³             ³', 7, 0,Inv);
     PutStr('³             ³', 8, 0,Inv);
     PutStr('³             ³', 9, 0,Inv);
     PutStr('³             ³',10, 0,Inv);
     PutStr('³             ³',11, 0,Inv);
     PutStr('³             ³',12, 0,Inv);
     PutStr('ÀÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÙ',13, 0,Inv);
     GotoXY(1,24);ClrEol;
     Write(DescripStr[1]);
     WINDOW(2,4,14,13);
     I:=0;
     J:=1;
     REPEAT
          I:=I+1;
          GOTOXY(1,I);
          WRITE(Directorio[i]);
     UNTIL (I=9) OR (I=K);
     I:=1;
     GOTOXY(1,J);
     VidPop;
     WRITE(Directorio[i]);
     Vidinv;
     REPEAT
          READ(KBD,TOR);
          IF (TOR=#27) And KeyPressed THEN
               BEGIN
               READ(KBD,TOR);
               CASE TOR OF
                    'H': UP;
                    'P': DAWN;
               ELSE
                    BEEP;
               END{CASE};
```

```
                END
          ELSE
                BEEP
          {END IF};
     UNTIL (TOR=#13);
     WINDOW(1,1,80,25);
     CurOn;
     ArchNom:=Directorio[I];
END;




begin
   Dir(Directorio,Count);
   If (Count=0) Then
      Begin
      GotoXY(1,24);ClrEol;
      Write('No existen archivos de datos en el disco presente.
<Return>');
      Repeat Until KeyPressed;
      GotoXY(1,24);ClrEol;
      Exit;
      End
   {EndIf};

   For i:=1 to Count Do
      Begin
      Assign(FileDesc,Directorio[i]);
      Reset(FileDesc);
      Read(FileDesc,DescripStr[i]);
      Close(FileDesc);
      End
   {EndFor};

   Directo;

   Assign(Archivo,ArchNom);
   ReSet(Archivo);
   Read(Archivo,Registro);
   Close(Archivo);
   X:=Registro.X;
   O:=Registro.O;
   V:=Registro.V;
   R:=Registro.R;
   Q:=Registro.Q;
   M:=Registro.M;
   P:=Registro.P;
   N:=Registro.N;
   Y:=Registro.Y;
   Lims:=Registro.Lims;
```

```
   GotoXY(1,24);ClrEol;
   DatosYa:=True;
End;



             (*** Programa Principal ***)
Begin
   DatosYa:=False;
   Lims:=40;
   GetScr(1,'CDZ');
   CurOff;
   PutScr(1);
   Repeat Until KeyPressed;
   CurOn;
   Repeat
      ClrScr;
      PutStr('ÚÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ¿',11,24,Inv);
      PutStr('³Menu Principal :         ³',12,24,Inv);
      PutStr('ÃÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ´',13,24,Inv);
      PutStr('³1.- Datos Nuevos.        ³',14,24,Inv);
      PutStr('³2.- Cargar Datos (Disco).³',15,24,Inv);
      PutStr('³3.- Analizar Datos.      ³',16,24,Inv);
      PutStr('³4.- Grabar Datos (Disco).³',17,24,Inv);
      PutStr('³5.- Salir.               ³',18,24,Inv);
      PutStr('ÀÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÙ',19,24,Inv);
      PutStr
('ÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ
ÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ',22,0,Inv);
      PutStr('
',23,0,Inv);
      PutStr
('ÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ
ÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ',24,0,Inv);

      CurOff;
      I:=1;
      VidPop;
      GotoXY(26,15);Write('1.- Datos Nuevos.        ');
      Repeat
         J:=I;
         Read(Kbd,Opcion);
         Case Opcion of
            #27 : If KeyPressed Then
                     Begin
                     Read(Kbd,Opcion);
                     Case Opcion of
                      'H','K' : If I=1 Then
                                   i:=5
                                Else
                                   i:=i-1
                                {EndIf};
```

```
                    'P','M' : If I=5 Then
                                 i:=1
                              Else
                                 i:=i+1
                              {EndIf};
                    'G': i:=1;
                    'O': i:=5;
                   End{Case};
                   VidInv;
                   GotoXY(26,14+J);
                   Case J of
                     1 : Write('1.- Datos Nuevos.        ');
                     2 : Write('2.- Cargar Datos (Disco).');
                     3 : Write('3.- Analizar Datos.      ');
                     4 : Write('4.- Grabar Datos (Disco).');
                     5 : Write('5.- Salir.               ');
                   End{Case};

                   GotoXY(26,14+I);
                   VidPop;
                   Case I of
                     1 : Write('1.- Datos Nuevos.        ');
                     2 : Write('2.- Cargar Datos (Disco).');
                     3 : Write('3.- Analizar Datos.      ');
                     4 : Write('4.- Grabar Datos (Disco).');
                     5 : Write('5.- Salir.               ');
                   End{case};
                   End
                Else
                   Opcion:='S'
                {EndIf};
          'S' : Opcion:='X';
          #13 : Opcion:=Chr(48+i);
       End{Case};

   Until Opcion In['1'..'5','S'];
   VidNorm;

   CurOn;
   Case Opcion Of
     '1': DatosNuevos;
     '2': CargaDatos;
     '3': If DatosYa Then Graphics;
     '4': If DatosYa Then GrabaDatos;
     '5': ClrScr;
   End{Case};
Until Opcion in['5','S'];
```

```
   CurOff;
   PutScr(1);
   Repeat Until KeyPressed;
   CurOn;
   ClrScr;

End.     (** P.PRINCIPAL **)
```

→