METODOLOGÍA DE LA PROGRAMACIÓN. Programación en Pascal

Autor: Víctor Sánchez2

El objetivo de este documento es proveer de una gran batería de ejercicios resueltos en Pascal que parten del nivel más básico hasta llegar a estructuras de datos más complejas.

Al final podreis ir encontrando las prácticas que voy realizando en mi universidad.

Gracias a todos por vuestra confianza.

Escribir un programa en Pascal que sume dos números:

a = 4 b = 3

```
PROGRAM EJER01;
        var a,b,c:INTEGER;

BEGIN

{Empezamos con lo básico, un programa que escribe la suma de 2 numeros en pantalla}

        a:=4;
        b:=3;

{Se asigna un valor cualquiera a las variables "a" y "b"}

        c:=a+b;

        WRITE (c); {Muestra en pantalla el valor de la suma}
END.


PROGRAM EJER1B;
  USES CRT; {Lo usamos para poder borrar la pantalla}
        VAR a,b,c:INTEGER;

BEGIN
        ClrScr; {Limpiamos la pantalla}

        WRITELN ('Este programa suma dos numeros:');
        WRITELN (' ');
        WRITE ('Introduzca un numero: ');       READLN (a);
        WRITE ('Introduzca otro numero: ' );    READLN (b);
        WRITELN (' ');
        c:=a+b;

        WRITE ('EL RESULTADO ES: ');
        WRITE (c);

END.
```

Escribir un programa en Pascal que sume, reste, multiplique y divida dos números:

x = 10 y = 2

```
PROGRAM EJER02;
        USES CRT; {Nos va a permitir limpiar la pantalla junto con ClrScr}
        VAR x,y:INTEGER;
        VAR suma,rest,mult,divi:INTEGER;


BEGIN
```

```
        x:=10;
        y:=2;

        suma:=x + y;
        rest:=x - y;
        mult:=x * y;
        divi:=x div y;

{Con estas 4 variables realizamos las cuatro operaciones aritméticas fundamentales: suma,
resta, multiplicación y división}

        ClrScr; {Limpia la pantalla}

        WRITE ('SUMA:');                    WRITELN (suma);
        WRITE ('RESTA:');                   WRITELN (rest);
        WRITE ('MULTIPLICACION:');          WRITELN (mult);
        WRITE ('DIVISION:');                WRITE (divi);

END.


PROGRAM EJER2B;
        USES CRT;
        VAR x,y:REAL;
        VAR suma,rest,mult:REAL;
        VAR divi:REAL;

        {suma, resta, multiplica y divide 2 numeros reales}

BEGIN
        WRITELN ('Este programa suma, resta, multiplica y divide:');
        WRITELN ('Escriba dos numeros reales');
        WRITELN (' ');


        Read(x);
        Read(y);

        suma:=x + y;
        rest:=x - y;
        mult:=x * y;
        divi:=x / y;

        ClrScr;

        WRITE ('SUMA:');                WRITELN (suma:3:0);
        WRITE ('RESTA:');               WRITELN (rest:3:0);
        WRITE ('MULTIPLICACION:');      WRITELN (mult:3:0);
        WRITE ('DIVISION:');            WRITE (divi:5:2);

END.


PROGRAM EJER02;
        USES CRT;
        VAR x,y:INTEGER;
        VAR suma,rest,mult,divi:INTEGER;
BEGIN
     x:=10;
     y:=2;

     suma:=x + y;
     rest:=x - y;
     mult:=x * y;
     divi:=x div y;

     ClrScr;

     WRITE('SUMA:');                        WRITELN(suma);
     WRITE('RESTA:');                       WRITELN(rest);
     WRITE('MULTIPLICACION:');              WRITELN(mult);
     WRITE('DIVISION:');                    WRITELN(divi);
END.
```

Escribir un programa en Pascal que calcule el área de un rectángulo:

lado1 = 3 lado2 = 4
área del rectángulo=lado1 * lado2

```
PROGRAM EJER03;
        USES CRT;
        VAR lado1,lado2:INTEGER;
        VAR area:INTEGER;

BEGIN

{Este programa nos va a servir para calcular el area de un rectángulo}

        {Damos valores para las variables}
        lado1:=3;
        lado2:=4;

        area:=lado1*lado2; {Calculamos el area}

        ClrScr;

        WRITE ('AREA DEL RECTANGULO: ');  WRITE (area); {Lo mostramos en pantalla}
END.
```

```
PROGRAM EJER3B;
        USES CRT;
        VAR lado1,lado2:REAL;
        VAR area:REAL;

BEGIN
        {Este programa calcula el area de un rectangulo}

        ClrScr;

        WRITELN ('Escriba los lados del rectangulo');

        Read(lado1);
        Read(lado2);
        WRITELN (' ');

        area:=lado1*lado2;

        WRITE ('AREA DEL RECTANGULO:');  WRITE (area:5:2);
END.
```

Escribir un programa en Pascal que calcule el área de un triángulo:

base = 7 altura = 4 área del triángulo = (base * altura)/2

```
PROGRAM EJER04;
        USES CRT;
        VAR base,altura:REAL;
        VAR area:REAL;
BEGIN
        base:=7;
        altura:=4;

        area:=(base * altura) / 2;

        ClrScr;

        WRITE ('AREA DEL TRIANGULO: ');  WRITE (area:5:2);
      {:5:2 sirve para dar el formato de salida al numero, 5 posiciones y 2 decimales}
END.
```

```
PROGRAM EJER4B;
        USES CRT;
        VAR base,altura:REAL;
        VAR area:REAL;
BEGIN
        {Este programa sirve para calcular el area de un triangulo}

        ClrScr;
        WRITELN ('PARA CALCULAR EL AREA DE UN TRIANGULO:');
        WRITELN (' ');
        WRITE ('ESCRIBE LA BASE: ');         READLN (base);
        WRITE ('ESCRIBE LA ALTURA: ');       READLN (altura);
        WRITELN (' ');

        area:=(base * altura) / 2;

        WRITE ('EL AREA DEL TRIANGULO ES: ');  WRITE (area:5:2);
END.
```

Escribir un programa que calcule la longitud y el área de una circunferencia:

radio = 4 longitud de la circunferencia = 2 * PI * radio
área de la circunferencia = PI * radio^2

```
PROGRAM EJER05;
        USES CRT;
        VAR radio:REAL;
        VAR longitud,area:REAL;

BEGIN
        radio:=4;
        longitud:=2*3.1416*radio;

        area:=3.1416*radio*radio;

        ClrScr;

        WRITE ('LONGITUD DE LA CIRCUNFERENCIA:');       WRITELN (longitud:5:2);
        WRITE ('AREA DE LA CIRCUNFERENCIA:');           WRITE (area:5:2);
END.
```

Escribir un programa en Pascal que calcule la velocidad de un proyectil que recorre 2 Km en 5 minutos. Expresar el resultado en metros/segundo.

Velocidad = espacio/tiempo

```
PROGRAM EJER06;
        USES CRT;
        VAR espacio,tiempo:REAL;
        VAR velocidad:REAL;

BEGIN
        espacio:=2;
        tiempo:=5;

        velocidad:=(espacio*1000)/(tiempo*60);

        ClrScr;

        WRITE ('VELOCIDAD DEL PROYECTIL:');
        WRITE (velocidad:5:2); WRITE ('m/s');
END.


PROGRAM EJER6B;
        USES CRT;
        VAR espacio,tiempo,espacio2,tiempo2:REAL;
        VAR velocidad,velocidad2:REAL;
BEGIN
```

```
        {Este programa calcula la velocidad de un cuerpo}

        ClrScr;

        WRITE ('Para calcular la velocidad debe escribirlo en unidades ');
        WRITE ('del sistema internacional');
        WRITELN (' ');
        WRITE ('Escriba el espacio recorrido: ');      READLN (espacio);
        WRITE ('Escriba el tiempo transcurrido: ');    READLN (tiempo);
        WRITELN (' ');

        velocidad:=(espacio)/(tiempo);

        WRITE ('VELOCIDAD DEL PROYECTIL: ');
        WRITE (velocidad:5:2); WRITELN (' m/s');

        WRITELN (' ');
        WRITELN ('Si lo desea en Km/h introduzca los datos: ');
        WRITELN (' ');
        WRITE ('Escriba el espacio recorrido: ');     READLN (espacio2);
        WRITE ('Escriba el tiempo transcurrido: ');   READLN (tiempo2);
        WRITELN (' ');

        velocidad2:=(espacio2)/(tiempo2);

        WRITE (velocidad2:5:2); WRITE (' Km/h ');
END.


PROGRAM EJER06;
        USES CRT;
        VAR espacio,tiempo:REAL;
        VAR velocidad:REAL;

BEGIN
     espacio:=2;
     tiempo:=5;

     velocidad:=(espacio*1000)/(tiempo*60);

     ClrScr;

     WRITE('VELOCIDAD DEL PROYECTIL:');
     WRITE(velocidad:5:2);             WRITE(' m/s');
END.
```

Escribir un programa en Pascal que calcule el volumen de una esfera:

radio = 3 volumen de la esfera = 4/3 * PI * radio^3

```
PROGRAM EJER07;
        USES CRT;
        VAR radio:REAL;
        VAR volumen:REAL;
BEGIN
        radio:=3;
        volumen:=(4/3)*3.1416*(radio*radio*radio);

        ClrScr;

        WRITE ('VOLUMEN DE LA ESFERA:');         WRITE(volumen);
END.


PROGRAM EJER7B;
        USES CRT;
        VAR radio:REAL;
        VAR volumen:REAL;
BEGIN
        {Este programa calcula el volumen de una esfera}

        ClrScr;
```

```
        WRITELN ('PARA CALCULAR EL VOLUMEN DE LA ESFERA ESCRIBA EL RADIO: ');
        READLN (radio);

        volumen:=(4/3)*3.1416*(radio*radio*radio);

        WRITE ('VOLUMEN DE LA ESFERA: ');         WRITE(volumen:5:2);
END.


PROGRAM EJER07;
        USES CRT;
        VAR radio:REAL;
        VAR volumen:REAL;
BEGIN
     radio:=3;

     volumen:=(4/3)*3.1416*(radio*radio*radio);

     ClrScr;

     WRITE('VOLUMEN DE LA ESFERA: ');   WRITE(volumen);
END.
```

Escribir un programa en Pascal que evalúe la siguiente expresión:

(a+7*c)/(b+2-a)+2*b a = 3,b = 6,c = 4

```
PROGRAM EJER08;
        USES CRT;
        VAR a,b,c:REAL;
        VAR resultado:REAL;
BEGIN
        a:=3;
        b:=6;
        c:=4;

        resultado:=(a+7*c)/(b+2-a)+2*b;

        ClrScr;

        WRITE ('RESULTADO:');   WRITE (resultado:5);
END.

PROGRAM EJER8B;
        USES CRT;
        VAR a,b,c:REAL;
        VAR resultado:REAL;

BEGIN
        {Este programa calcula una expresion algebraica}

        ClrScr;

        WRITELN ('Este programa sirve para calcular la siguiente expresion:');
        WRITELN ('(a+7*c)/(b+2-a)+2*b');
        WRITELN (' ');
        WRITE ('Introduzca a: ');         READLN (a);
        WRITE ('Introduzca b: ');         READLN (b);
        WRITE ('Introduzca c: ');         READLN (c);

        resultado:=(a+7*c)/(b+2-a)+2*b;
        WRITELN (' ');
        WRITE ('RESULTADO: ');   WRITE (resultado:5:2);

END.


PROGRAM EJER08;
        USES CRT;
        VAR a,b,c:REAL;
        VAR resultado:REAL;
BEGIN
     a:=3;
```

```
    b:=6;
    c:=4;

    resultado:=(a+7*c)/(b+2-a)+2*b;

    ClrScr;

    WRITE('RESULTADO: ');      WRITE(resultado:5);
END.
```

Escribir un programa en Pascal que evalúe la siguiente expresión:

(a+5) * 3 / 2 * b - b a = 3,b = 6

```
PROGRAM EJER09;
        USES CRT;
        VAR a,b:REAL;
        VAR resultado:REAL;

BEGIN
        a:=3;
        b:=6;

        resultado:=((a+5)*3) / (2*b-b);

        ClrScr;

        WRITE ('RESULTADO: ');   WRITE(resultado:5:2);
END.


PROGRAM EJER9B;
        USES CRT;
        VAR a,b:REAL;
        VAR resultado:REAL;

BEGIN
        {Este programa calcula el resultado de una expresion algebraica}

        ClrScr;

        WRITE ('PARA CALCULAR LA SIGUIENTE EXPRESION: ');
        WRITELN ('((a+5)*3) / (2*b-b)');
        WRITELN (' ');
        WRITE ('Escriba a: ');  READ (a);
        WRITE ('Escriba b: ');  READ (b);
        WRITELN (' ');

        resultado:=((a+5)*3) / (2*b-b);

        WRITE ('RESULTADO: ');   WRITE(resultado:5:2);
END.
```

Escribir un programa en Pascal que evalúe la siguiente expresión:

(-b + √(b²-4*a*c)/(2*a)
(es la solución positiva de una ecuación de 2º grado)

```
PROGRAM EJER10;
        USES CRT;
        VAR a,b,c:REAL;
        VAR resultado:REAL;

BEGIN
        a:=6;
        b:=6;
        c:=1;

        resultado:=(-b+sqrt(sqr (b) - 4*a*c))/(2*a);
```

```
        ClrScr;

        WRITE ('RESULTADO:');   WRITE(resultado:5:2);
END.



PROGRAM EJER10B;
        USES CRT;
        VAR a,b,c:REAL;
        VAR resultado:REAL;

BEGIN
        {Calcula la incognita positiva de una ecuacion de 2º grado}

        ClrScr;

        WRITE ('Para calcular la incognita positiva de una ecuacion');
        WRITE (' de segundo grado escriba todas las variables:');
        WRITELN (' ');
        WRITELN (' ');

        WRITE ('Escriba a: ');      READLN (a);
        WRITE ('Escriba b; ');      READLN (b);
        WRITE ('Escriba c; ');      READLN (c);
        WRITELN (' ');

        resultado:=(-b +sqrt(sqr (b) - 4*a*c))/(2*a);

        WRITE ('RESULTADO: ');   WRITE(resultado:5:2);

END.
```

Escribir un programa en Pascal que calcule el área y el volumen de un cilindro:

$$A = (2 * (PI * r^2)) + ((2 * PI * r) * h)$$
$$V = (PI * r^2) * h$$

```
PROGRAM EJER11;
        USES CRT;
        VAR radio,altura:REAL;
        VAR area,volumen:REAL;

BEGIN
        radio:=3;
        altura:=6;

        area:= (2 * (3.1416 * radio * radio)) + ((2 * 3.1416 * radio) * altura);
        volumen:= (3.1416 * radio * radio) * altura;

{podriamos cambiar "radio*radio" por "sqr(radio)" para hacer el cuadrado del radio}

        ClrScr;

        WRITE ('AREA DEL CILINDRO:');          WRITE (area);      WRITELN (' m2');
        WRITE ('VOLUMEN DEL CILINDRO:');       WRITE (volumen);   WRITE (' m3');
END.


PROGRAM EJER11B;
        USES CRT;
        VAR radio,altura:REAL;
        VAR area,volumen:REAL;

BEGIN
        {Calcula el area y el volumen de un cilindro}

        ClrScr;

        WRITELN ('CALCULA EL AREA Y VOLUMEN DE UN CILINDRO');
        WRITELN (' ');

        WRITE ('Escriba el radio: ');         READLN (radio);
        WRITE ('Escriba la altura: ');        READLN (altura);
```

```
        WRITELN (' ');

        area:= (2 * (3.1416 * radio * radio)) + ((2 * 3.1416 * radio) * altura);
        volumen:= (3.1416 * radio * radio) * altura;

        WRITE ('AREA DEL CILINDRO: ');   WRITE (area:5:2); WRITELN (' m2');
        WRITE ('VOLUMEN DEL CILINDRO: '); WRITE (volumen:5:2); WRITE (' m3');
END.


PROGRAM EJER11;
        USES CRT;
        VAR r,h:REAL;
        VAR a,v:REAL;
BEGIN

     {AREA Y VOLUMEN DE UN CILINDRO}

     ClrScr;

     WRITE('RADIO DEL CILINDRO: ');     READLN(r);
     WRITE('ALTURA DEL CILINDRO: ');    READLN(h);

     a:=(2*(3.1416*sqr(r))) + ((2*3.1416*r)*h);
     v:=(3.1416*sqr(2))*h;

     ClrScr;

     WRITE('AREA DEL CILINDRO: ');          WRITELN(a:5:2);
     WRITE('VOLUMEN DEL CILINDRO: ');       WRITELN(v:5:2);
END.
```

Escribir un programa en Pascal que calcule el área y el volumen de un hexaedro

$A = (l*l) * 6$
$V = l^3$

```
PROGRAM EJER12;
        USES CRT;
        VAR lado:REAL;
        VAR area,volumen:REAL;

BEGIN
        lado:=4;

        area:= (lado * lado) * 6;
        volumen:= sqr(lado) * lado;

        ClrScr;

        WRITE ('AREA DEL HEXAEDRO:');     WRITE (area);    WRITELN (' m2');
        WRITE ('VOLUMEN DEL HEXAEDRO:');  WRITE (volumen); WRITE (' m3');

END.


PROGRAM EJER12B;
        USES CRT;
        VAR lado:REAL;
        VAR area,volumen:REAL;

BEGIN
        ClrScr;

        WRITE ('INTRODUCE EL LADO DEL HEXAEDRO: ');

        READLN (lado);
        WRITELN (' ');

        area:= (lado * lado) * 6;
        volumen:= sqr(lado) * lado;
```

```
        WRITE ('AREA DEL HEXAEDRO: ');      WRITE (area:5:2);     WRITELN (' m2');
        WRITE ('VOLUMEN DEL HEXAEDRO: ');   WRITE (volumen:5:2);  WRITE (' m3');

END.
```

Escribir un programa en Pascal que calcule el área y el volumen de un prisma

A = (2 *(l1 * l2)) + (2 * (l1 * l3)) + (2*(l2 * l3))
V = l1 * l2 * l3

```
PROGRAM EJER13;
        USES CRT;
        VAR l1,l2,l3:REAL;
        VAR area,volumen:REAL;

BEGIN
        l1:=3;
        l2:=6;
        l3:=4;

        area:=2 * (l1 * l2)+(2 * (l1 * l3)) + (2* (l2 * l3));
        volumen:= l1 * l2 * l3;

        ClrScr;

        WRITE ('AREA DEL PRISMA:');          WRITELN(area);
        WRITE ('VOLUMEN DEL PRISMA:');      WRITE (volumen);
END.


PROGRAM EJER13B;
        USES CRT;
        VAR l1,l2,l3:REAL;
        VAR area,volumen:REAL;

BEGIN
        {Calcula el area y volumen de un prisma}

        ClrScr;

        WRITELN ('PARA CALCULAR EL AREA Y EL VOLUMEN DEL PRISMA, ESCRIBA: ');
        WRITELN (' ');
        WRITE ('Lado1: ');  READLN (l1);
        WRITE ('Lado2: ');  READLN (l2);
        WRITE ('Lado3: ');  READLN (l3);
        WRITELN (' ');

        area:=2 * (l1 * l2)+(2 * (l1 * l3)) + (2* (l2 * l3));
        volumen:= l1 * l2 * l3;

        WRITE ('AREA DEL PRISMA: ');         WRITELN (area:5:2);
        WRITE ('VOLUMEN DEL PRISMA: ');     WRITE (volumen:5:2);

END.
```

Escribir un programa en Pascal que calcule el área y el volumen de un tetraedro

A = a^{2} * raízcuadrada(3)
V = (a^{3}/12) * raízcuadrada (2)

```
PROGRAM EJER14;
        USES CRT;
        VAR arista:REAL;
        VAR area, volumen:REAL;

BEGIN

        arista:=5;
```

```
        area:= sqr(arista) * sqrt(3);
        volumen:= ((sqr(arista) * arista) / 12) * sqrt(2);

        WRITE ('AREA DEL TETRAEDRO: ');             WRITELN (area);
        WRITE ('VOLUMEN DEL TETRAEDRO: ');          WRITE (volumen);
END.


PROGRAM EJER14B;
        USES CRT;
        VAR arista:REAL;
        VAR area, volumen:REAL;

BEGIN
        {Calcula el area y el volumen de un octaedro}

        WRITELN ('SI DESEA CALCULAR EL AREA Y EL VOLUMEN DE UN TETRAEDRO: ');
        WRITELN (' ');
        WRITE ('INTRODUZCA EL VALOR DE SU ARISTA: ');   READLN (arista);
        WRITELN (' ');


        area:= sqr(arista) * sqrt(3);
        volumen:= ((sqr(arista) * arista) / 12) * sqrt(2);

        WRITE ('AREA DEL TETRAEDRO: ');           WRITELN (area:5:2);
        WRITE ('VOLUMEN DEL TETRAEDRO: ');        WRITE (volumen:5:2);

END.
```

Escribir un programa en Pascal que calcule el área y el volumen de un octaedro

$A = 2 * a^2 *$ raízcuadrada(3)
$V=(a^3/3) *$ raízcuadrada(2)

```
PROGRAM EJER15;
        USES CRT;
        VAR arista:REAL;
        VAR area, volumen:REAL;

BEGIN

        arista:=4;

        area:= 2 * sqr(arista) * sqrt(3);
        volumen:= ((sqr(arista) * arista) / 3) * sqrt(2);

        WRITE ('AREA DEL OCTAEDRO: ');              WRITELN(area);
        WRITE ('VOLUMEN DEL OCTAEDRO: ');           WRITE(volumen);

END.



PROGRAM EJER15B;
        USES CRT;
        VAR arista:REAL;
        VAR area, volumen:REAL;

BEGIN
        {Sirve para calcular el area y el volumen de un tetraedro}

        WRITELN ('PARA CALCULAR EL AREA Y VOLUMEN DE UN TETRAEDRO: ');
        WRITE ('ESCRIBA EL VALOR DE LA ARISTA: ');   READLN (arista);
        WRITELN (' ');

        area:= 2 * sqr(arista) * sqrt(3);
        volumen:= ((sqr(arista) * arista) / 3) * sqrt(2);

        WRITE ('AREA DEL OCTAEDRO: ');           WRITELN (area:5:2);
        WRITE ('VOLUMEN DEL OCTAEDRO: ');        WRITE (volumen:5:2);

END.
```

Escribir un programa en Pascal que calcule el área y el volumen de un cono

$A = (PI * r * l)+ (PI * r^2)$
$V = (PI * r^2 * h)/3$

```
PROGRAM EJER16;
        USES CRT;
        VAR radio,lado,altura:REAL;
        VAR area,volumen:REAL;

BEGIN

        radio:=6;
        lado:=3;
        altura:=8;

        area:= (3.1416 * radio * lado) + (3.1416 * sqr(radio));
        volumen:= (3.1416 * sqr(radio) * altura) / 3;

        WRITE ('AREA DEL CONO: ');          WRITELN (area);
        WRITE ('VOLUMEN DEL CONO: ');       WRITE (volumen);
END.


PROGRAM EJER16B;
        USES CRT;
        VAR radio,lado,altura:REAL;
        VAR area,volumen:REAL;

BEGIN
        {Se utiliza para calcular el area y volumen de un cono}

        WRITELN ('Para calcular el area y el volumen de un cono: ');
        WRITELN (' ');
        WRITE ('Escriba el valor del radio: ');         READLN (radio);
        WRITE ('Escriba el valor del lado: ');          READLN (lado);
        WRITE ('Escriba el valor de la altura: ');      READLN (altura);
        WRITELN (' ');

        area:= (3.1416 * radio * lado) + (3.1416 * sqr(radio));
        volumen:= (3.1416 * sqr(radio) * altura) / 3;

        WRITE ('AREA DEL CONO: ');          WRITELN (area:5:2);
        WRITE ('VOLUMEN DEL CONO: ');       WRITE (volumen:5:2);

END.
```

Escribir un programa en Pascal que calcule el volumen de un elipsoide

$V = (4/3) * PI * a * b * c$

```
PROGRAM EJER17;
        USES CRT;
        VAR a,b,c:REAL;
        VAR volumen:REAL;

BEGIN

        a:=3;
        b:=5;
        c:=4;

        volumen:= (4/3) * 3.1416 * a * b * c;

        WRITE ('VOLUMEN DEL ELIPSOIDE:');       WRITE (volumen);
END.

PROGRAM EJER17B;
        USES CRT;
```

```
        VAR a,b,c:REAL;
        VAR volumen:REAL;

BEGIN
        {Calcula el volumen de un elipsoide}

        ClrScr;

        WRITELN ('PARA CALCULAR EL VOLUMEN DE UN ELIPSOIDE ESCRIBA: ');
        WRITELN (' ');

        WRITE ('A: ');  READLN (a);
        WRITE ('B: ');  READLN (b);
        WRITE ('C: ');  READLN (c);

        volumen:= (4/3) * 3.1416 * a * b * c;

        WRITE ('VOLUMEN DEL ELIPSOIDE: ');      WRITE (volumen:5:2);
END.
```

Escribir un programa en Pascal que calcule las raíces de una ecuación de 2º grado

```
PROGRAM EJER18;
        USES CRT;
        VAR a,b,c:REAL;
        VAR x1,x2:REAL;

BEGIN

        a:=6;
        b:=6;
        c:=1;

        x1:= (-b + sqrt(sqr(b) - (4 * a * c))) / 2 * a;
        x2:= (-b - sqrt(sqr(b) - (4 * a * c))) / 2 * a;

        WRITE ('SOLUCION 1:');  WRITELN (x1);
        WRITE ('SOLUCION 2:');  WRITE (x2);
END.



PROGRAM EJER18B;
        USES CRT;
        VAR a,b,c:REAL;
        VAR resultado1,resultado2:REAL;

BEGIN

        {Calcula ecuaciones de segundo grado}

        ClrScr;

        WRITE ('ESTE PROGRAMA SIRVE PARA CALCULAR ECUACIONES ');
        WRITELN ('DE SEGUNDO GRADO');
        WRITELN (' ');
        WRITELN ('Introduzca: a, b y c: ');
        WRITELN (' ');
        READLN (a);
        READLN (b);
        READLN (c);

        resultado1:=(-b + sqrt(sqr(b) - 4*a*c)) / (2*a);
        resultado2:=(-b - sqrt(sqr(b) - 4*a*c)) / (2*a);
        WRITELN ('RESULTADO DE LA EXPRESION: ');
        WRITE ('VALOR 1: ');   WRITELN (resultado1:5:2);
        WRITE ('VALOR 2; ');   WRITE (resultado2:5:2);
END.
```

Escribir un programa en Pascal que calcule el área y el volumen de un cilindro:

radio = 3

altura = 4

```
PROGRAM EJER19;
        USES CRT;
        VAR radio, altura:REAL;
        VAR area, volumen:REAL;
BEGIN
        radio:=3;
        altura:=4;

        area:= 2 * (3.1416 * sqr(radio)) + ((2 * 3.1416 * radio) * altura);
        volumen:= (3.1416 * sqr(radio)) * altura;

        ClrScr;

        WRITE ('EL AREA DEL CILINDRO ES: ');        WRITELN (area:6:2);
        WRITE ('EL VOLUMEN ES: ');                  WRITE (volumen:6:2);
END.


PROGRAM EJER19B;
        USES CRT;

        VAR radio, altura:REAL;
        VAR area, volumen:REAL;
BEGIN
        {Con este programa podremos calcular el area y el volumen
        de un cilindro}

        ClrScr;

        WRITELN ('PARA CALCULAR EL AREA Y VOLUMEN DE UN CILINDRO: ');
        WRITELN (' ');
        WRITE ('ESCRIBA EL RADIO- ');     READLN (radio);
        WRITE ('ESCRIBA LA ALTURA- ');    READLN (altura);
        WRITELN (' ');

        area:= 2 * (3.1416 * sqr(radio)) + ((2 * 3.1416 * radio) * altura);
        volumen:= (3.1416 * sqr(radio)) * altura;

        WRITE ('EL AREA DEL CILINDRO ES: ');        WRITELN (area:6:2);
        WRITE ('EL VOLUMEN ES: ');                  WRITE (volumen:6:2);
END.
```

Escribir un programa en Pascal que calcule la hipotenusa de un triángulo rectángulo

cateto 1 = 5
cateto 2 = 5

```
PROGRAM EJER20;
        USES CRT;

        VAR cateto1,cateto2:REAL;
        VAR hipotenusa:REAL;

BEGIN

        cateto1:=5;
        cateto2:=5;

        hipotenusa:= sqrt(sqr(cateto1) + sqr(cateto2));

        ClrScr;

        WRITE ('HIPOTENUSA DEL TRIANGULO: ');
        WRITE (hipotenusa:5:2); WRITE (' cm');

END.


PROGRAM EJER20B;
```

```
        USES CRT;

        VAR cateto1,cateto2:REAL;
        VAR hipotenusa:REAL;

BEGIN
        {Con este programa podremos calcular la hipotenusa de un triangulo}

        ClrScr;

        WRITE ('PARA CALCULAR LA HIPOTENUSA DEL TRIANGULO ');
        WRITELN ('ESCRIBA LOS CATETOS: ');
        WRITELN (' ');
        WRITE ('Cateto1: ');    READLN (cateto1);
        WRITE ('Cateto2: ');    READLN (cateto2);
        WRITELN (' ');

        hipotenusa:= sqrt(sqr(cateto1) + sqr(cateto2));

        WRITE ('HIPOTENUSA DEL TRIANGULO: ');
        WRITE (hipotenusa:5:2);

END.


PROGRAM EJER20;
        USES CRT;
        VAR c1,c2,h:REAL;

BEGIN
     ClrScr;

     WRITE('Introduzca cateto_1: ');    READLN (c1);
     WRITE('Introduzca cateto_2: ');    READLN (c2);

     h:=sqrt(sqr(c1)+sqr(c2));

     WRITE('Cateto_1 ------> ');        WRITELN (c1:5:2);
     WRITE('Cateto_2 ------> ');        WRITELN (c2:5:2);
     WRITE('Hipotenusa ----> ');        WRITELN (h:5:2);
END.
```

Escribir un programa en Pascal que calcula el equivalente en grados Fahrenheit o Celsius de las siguientes temperaturas.

Temperatura 1 = 32º Fahrenheit
Temperatura 2 = 10 º Celsius

{Regla de 3: Celsius / 5 = (Fahrenheit – 32) 9}

```
PROGRAM EJER21;
        USES CRT;

        VAR T1,T2:REAL;
        VAR T1C,T2F:REAL;
BEGIN
        T1:=32;
        T2:=10;

        T1C:=T1 - 32;
        T2F:=T2 + 32;

        ClrScr;

        WRITE ('TEMPERATURA EQUIVALENTE: ');
        WRITE (T1:3:0);  WRITE ('º Fahrenheit - ');
        WRITE (T1C:3:0); WRITELN ('º Celsius');

        WRITE ('TEMPERATURA EQUIVALENTE: ');
        WRITE (T2:3:0);  WRITE ('  Celsius - ');
        WRITE (T2F:3:0); WRITE ('º Fahrenheit');
```

```
END.


PROGRAM EJER21B;
        USES CRT;

        VAR Fahrenheit, Celsius:REAL;
        VAR T1C,T2F:REAL;
        VAR respuesta:CHAR;
BEGIN
        ClrScr;

        REPEAT
         BEGIN
           WRITE (' DESEA PASARLO A FAHRENHEIT O CELSIUS? F/C: ');
           READLN (respuesta);    WRITELN ('');
         END;
        UNTIL (respuesta='C') OR (respuesta='F') OR
              (respuesta='c') OR (respuesta='f');

        IF UPCASE(respuesta)='F' THEN
        BEGIN
          WRITELN ('Introduzca los grados para pasar a Fahrenheit: ');
          WRITE ('Celsius: ');     READLN (Celsius);

          Fahrenheit:= ((9 * Celsius) / 5) + 32;

          WRITE (Fahrenheit:5:2,' grados Fahrenheit.');
          WRITELN (' ');
        END

        ELSE IF UPCASE (respuesta)='C' THEN
        BEGIN
          WRITELN ('Introduzca los grados para pasar a Celsius: ');
          WRITE ('Fahrenheit: ');  READLN (Fahrenheit);

          Celsius:= ((Fahrenheit - 32) / 9) * 5;

          WRITE (Celsius:5:2,' grados Celsius.');
        END;
END.


PROGRAM EJER21;
        USES CRT;
        VAR t_C,t_F:REAL;
BEGIN

     ClrScr;

     WRITE('Introduzca temperatura: (ºCelsius): ');
     READLN(t_C);

     t_F:=((t_C*9)/5)+32;

     ClrScr;

     WRITE(t_C:5:2);    WRITE(' ºCelsius equivalen a ');
     WRITE(t_F:5:2);    WRITE(' ºFahrenheit');
END.
```

Escribir un programa que lea dos números enteros A y B, y obtenga los valores A div B, A mod B.

```
PROGRAM EJERDIV;
        Uses Crt;
             Var A,B: Integer;
             Var soluc: Integer;
Begin
     ClrScr;
     WRITELN('Introduzca dos numeros:');
     WRITELN;
     WRITE('A: '); READLN(A);
     WRITE('B: '); READLN(B);
     WRITELN;
```

```
     WRITE('A div B = ');
              soluc := A div B; {div hace la division de 2 numeros enteros}
     WRITELN(soluc);
     WRITELN;

     WRITE('A mod B = ');
              soluc := A mod B; {mod muestra el resto de una division de
                                 2 numeros enteros}
     WRITELN(soluc);
End.

PROGRAM divi(Input, Output);
Uses Crt;
  var A, B, aDb, aMb: integer;
Begin
 ClrScr;

 write('Dime un número entero: ');
 readln(A);
 write('Dime otro número entero: ');
 readln(B);
 aDb := A div B;
 aMb := A mod B;
 writeln('A div B = ',aDb);
 writeln('A mod B = ',aMb);
End.
```

Escribir un programa en Pascal que calcule el número de horas, minutos y segundos que hay en 3700 segundos.

```
PROGRAM EJER22;
        USES CRT;

        VAR horas, minutos, segundos:INTEGER;

BEGIN
        horas:= 3700 div 3600;
        minutos:= (3700 mod 3600) div 60;
        segundos:= (3700 mod 3600) - (minutos * 60);


        ClrScr;

        WRITELN ('EN 3700 SEGUNDOS HAY: ');
        WRITE (horas,' hora',' y ',minutos,' minutos ', segundos,' segundos');
END.


PROGRAM EJER22B;
        USES CRT;

        VAR horas, minutos, segundos:INTEGER;
        VAR cantidad:INTEGER;

BEGIN
        ClrScr;

        WRITE ('Escriba los segundos para transformarlo a horas,');
        WRITELN (' minutos y segundos');
        READLN (cantidad); {Es el numero de segundos que se introducen}
        WRITELN ('');

        horas:= cantidad div 3600;
        minutos:= (cantidad mod 3600) div 60;
        segundos:= (cantidad mod 3600) - (minutos * 60);
        {Los segundos son: las horas - los minutos pasados a segundos}

        WRITELN ('EN ' ,cantidad, ' SEGUNDOS HAY: ');
        WRITE (horas,' horas ',minutos,' minutos ',segundos,' segundos');
END.


PROGRAM EJER22;
```

```
        USES CRT;
        VAR h,m,s1,s2:INTEGER;
BEGIN

     ClrScr;

     WRITE('Introduzca segundos: ');    READLN(s1);

     h:=s1 div 3600;
     s2:=s1 mod 3600;

     m:=s2 div 60;
     s2:=s2 mod 60;

     ClrScr;

     WRITE(s1); WRITE(' segundos son -----> ');
     WRITE(h);  WRITE(' horas ');
     WRITE(m);  WRITE(' minutos ');
     WRITE(s2); WRITE( 'segundos ');
END.
```

Escribir un programa en Pascal que calcule el capital producido por un capital de 1.000.000 de pesetas, al cabo de un año depositado a un interés del 2%.

```
PROGRAM EJER23;
        USES CRT;7

        VAR capital,tiempo,interes:REAL;
        VAR capitalproducido:REAL;

BEGIN
        capital:=1000000;
        tiempo:=1;
        interes:=2;

        capitalproducido:= capital * 0.02;

        ClrScr;

        WRITE ('En un año se producira un capital de ');
        WRITE (capitalproducido:5:2);     WRITE (' pesetas');
END.


PROGRAM EJER23B;
        USES CRT;

        VAR capital,tiempo,interes:REAL;
        VAR capitalproducido:REAL;


BEGIN
        ClrScr;

        WRITELN ('PARA CALCULAR EL CAPITAL PRODUCIDO INTRODUZCA ');
        WRITELN ('');
        WRITE ('Capital: ');    READLN (capital);
        WRITE ('Tiempo: ');     READLN (tiempo);
        WRITE ('Interes:');     READLN (interes);

        WRITELN ('');

        capitalproducido:= (capital * (interes/100) * tiempo);

        WRITE ('En estos años se producira un capital de ');
        WRITE (capitalproducido:5:2);     WRITE (' pesetas.');
END.


PROGRAM EJER23;
       USES CRT;
       VAR capital,interes,intereses:REAL;
```

```
BEGIN

     ClrScr;

     WRITE('Capital: ');        READLN(capital);
     WRITE('Intefes: ');        READLN(interes);

     intereses:=capital*(interes/100);

     ClrScr;

     WRITE('Capital: ');        WRITELN(capital:5:2);
     WRITE('Interes: ');        WRITELN(interes:5:2);
     WRITE('Intereses: ');     WRITELN(intereses:5:2);
END.
```

Escribir un programa en Pascal que calcula la siguiente expresión trigonométrica para un valor angular de 90º

(sen x * cos x)/(tan x)

```
PROGRAM EJER24B;
        USES CRT;

        VAR resultado, resultado2, x:REAL;

BEGIN

        WRITE ('PARA CALCULAR LA EXPRESION: (sin(x) * cos(x)) / tan(x)');
        WRITELN (' INTRODUZCA EL VALOR DE X EN RADIANES: ');
        READLN (x);

        WRITELN ('');


        resultado:=(sin(x) * cos(x)) / (sin(x) / cos(x));


        WRITE ('El resultado de la expresion (sinx * cosx /tgx) es igual a: ');
        WRITE (resultado:5:2);
END.
```

Escribir un programa en Pascal que calcule el equivalente en pies de una longitud de 10 metros.

1 metro ----------▯ 39.27 pulgadas
12 pulgadas --------▯ 1 pie

```
PROGRAM EJER25;
        USES CRT;

        VAR metros,pulgadas,pies:REAL;

BEGIN

        metros:=10;
        pulgadas:=metros * 39.27;
        pies:=((1 * metros) * pulgadas) / (12 * metros);

        ClrScr;

        WRITE ('El equivalente en pies a una distancia de 10m es de: ');
        WRITE (pies:3:2); WRITE (' pies');
END.


PROGRAM EJER25B;
        USES CRT;
```

```
        VAR metros,pies:REAL;

BEGIN
        {Para calcular la equivalencia entre pies y metros}

        ClrScr;

        WRITELN ('INTRODUZCA LOS METROS PARA PASARLOS A PIES: ');
        WRITELN (' ');

        WRITE ('Metros: ');     READLN (metros);


        pies:= metros / (12/39.27);
        { 1 pie = 0.3048 metros}
        { 1 pulgada = 25.4 mm}

        WRITE ('El equivalente en pies es de: ');
        WRITE (pies:3:2); WRITE (' pies');

END.


PROGRAM EJER25;
        USES CRT;
        VAR longitud:REAL;
BEGIN
     ClrScr;

     WRITE('Longitud (metros): ');      READLN(longitud);

     WRITE((longitud*39.27)/12:5:2);    WRITE( 'pies');
END.
```

Escribir un programa en Pascal que calcule el área de un rectángulo a partir de sus coordenadas:

x1 = 10 x2 = 20
y1 = 10 y2 = 20

```
PROGRAM EJER26;
        USES CRT;

        VAR lado1,lado2:REAL;
        VAR area:REAL;

BEGIN

        lado1:=10;
        lado2:=10;

        area:= lado1 * lado2;

        ClrScr;

        WRITE ('El area del rectangulo es de: '); WRITE (area:5:2);

END.


PROGRAM EJER26B;
        USES CRT;

        VAR x1,x2,y1,y2:REAL;
        VAR area:REAL;

BEGIN
        {Sirve para calcular el area de un rectangulo a partir de
         coordenadas}

        WRITELN ('Para calcular el area del rectangulo ');
        WRITELN ('introduzca el valor de las coordenadas');
```

```
        WRITELN (' ');
        WRITE ('x1: ');       READLN (x1);
        WRITE ('y1: ');       READLN (y1);
        WRITE ('x2: ');       READLN (x2);
        WRITE ('y2: ');       READLN (y2);
        WRITELN (' ');

        area:= (x2 - x1) * (y2 - y1);
        {Se restan las coordenadas de X e Y para sacar los lados y
         luego se multiplican}

        WRITE ('El area del rectangulo es de: '); WRITE (area:5:2);
END.


PROGRAM EJER26;
        USES CRT;
        VAR x1,y1,x2,y2:REAL;
BEGIN
     ClrScr;

     WRITE('Introduaca coordenada x1: ');       READLN(x1);
     WRITE('Introduzca coordenada y1: ');       READLN(y1);
     WRITE('Introduzca coordenada x2: ');       READLN(x2);
     WRITE('Introduzca coordenada y2: ');       READLN(y2);

     WRITE('Area del rectangulo: '); WRITE((x2-x1)*(y2-y1):5:2);
END.
```

Un coche se mueve, partiendo del reposo, con una aceleración constante de 8 $m/s^{2.}$ Escribir un programa en Pascal que calcule:

a) La velocidad instantánea al cabo de5 segundos.
b) La velocidad media durante los primeros 5 segundos del recorrido.

velocidad instantánea = velocidad inicial + aceleración * tiempo
velocidad media = (velocidad inicial + velocidad final)/2

```
PROGRAM EJER27;
        USES CRT;

        VAR velocidad0,aceleracion,tiempo:REAL;
        VAR velocidad5,velocmedia5:REAL;
BEGIN
        velocidad0:=0;
        aceleracion:=8;
        tiempo:=5;

        velocidad5:=velocidad0 + (aceleracion * tiempo);
        velocmedia5:= (velocidad0 + velocidad5) / 2;

        ClrScr;

        WRITE ('LA VELOCIDAD AL CABO DE 5 s ES DE: '); WRITE (velocidad5:2:0);
        WRITELN (' m/s');
END.


PROGRAM EJER27B;
        USES CRT;

        VAR velocidad0,aceleracion,tiempo,velocidadfinal:REAL;
        VAR vinstantanea,vmedia:REAL;
BEGIN
        ClrScr;

        WRITE ('ESCRIBA EL VALOR DE LA VELOCIDAD INICIAL, LA ACELERACION');
        WRITE (' Y EL TIEMPO, EN UNIDADES DEL SISTEMA INTERNACIONAL,');
        WRITE (' PARA CALCULAR LA VELOCIDAD INSTANTANEA');
        WRITELN (' '); WRITELN (' ');
```

```
        WRITE ('Velocidad inicial: ');       READLN (velocidad0);
        WRITE ('Aceleracion: ');            READLN (aceleracion);
        WRITE ('Tiempo: ');                  READLN (tiempo);
        WRITELN ('');

        vinstantanea:=velocidad0 + (aceleracion * tiempo);

        IF vinstantanea > 0 THEN
           WRITE ('LA VELOCIDAD INSTANTANEA ES DE: ',vinstantanea:5:2,' m/s')
        ELSE
           WRITE ('EL COCHE ESTA PARADO.');

        WRITELN ('');

        IF vinstantanea < 0 THEN
        WRITE ('NO SE PUEDE HALLAR AL ESTAR PARADO');
        IF vinstantanea > 0 THEN
WRITE ('Si desea saber la velocidad media introduzca la velocidad final: ');
        READLN (velocidadfinal);
        WRITE ('');

        WRITELN ('');

        vmedia:= (velocidad0 + velocidadfinal) / 2;
        WRITE ('LA VELOCIDAD MEDIA ES DE: ',vmedia:5:2);
        WRITELN (' m/s');
END.


PROGRAM EJE27;
        USES CRT;
        VAR v,a,t:REAL;
BEGIN
     ClrScr;

     WRITE('Velocidad inicial (m/s) -> ');          READLN(v);
     WRITE('Aceleracion (m/s2) ------> ');          READLN(a);
     WRITE('Tiempo (s) --------------> ');          READLN(t);

     WRITE('Velocidad instantanea: ');              WRITELN(v+a*t:5:2);
     WRITE('Velocidad media: ');                    WRITELN((v+(v+a*t))/2:5:2);
END.
```

Un cohete se lanza verticalmente con una velocidad de 500 m7s calcular la velocidad al cabo de 40 segundos mediante un programa en Pascal

velocidad instantánea = (velocidad inicial) – (aceleración de la gravedad * tiempo)

```
PROGRAM EJER28;
        USES CRT;

        CONST gravedad = 9.81;
        VAR velocidad0, tiempo, velocidadfinal:REAL;
BEGIN
        velocidad0:=500;
        tiempo:=40;

        velocidadfinal:=velocidad0 - (gravedad * 40);

        ClrScr;


        WRITE ('La velocidad a los 40 s es de: ');
        WRITE (velocidadfinal:4:2,' m/s');
END.


PROGRAM EJER28B;
        USES CRT;

        CONST gravedad = 9.81;
        VAR velocidad0, tiempo, velocidadfinal:REAL;
```

```
BEGIN
        {Este programa sirve para calcular la velocidad instantanea
        de un cohete}

        ClrScr;

        WRITE ('PARA CALCULAR LA VELOCIDAD DE UN COHETE EN UN INSTANTE, ');
        WRITELN ('INTRODUZCA LOS DATOS:');
        WRITELN (' ');

        WRITE ('INTRODUZCA LA VELOCIDAD INICIAL: ');       READLN (velocidad0);
        WRITE ('INTRODUZCA EL TIEMPO:');                   READLN (tiempo);
        WRITELN (' ');

        velocidadfinal:=velocidad0 - (gravedad * tiempo);

        IF velocidadfinal <= 0 THEN
           WRITE ('El cohete ya se ha parado.');

        IF velocidadfinal > 0 THEN
            WRITE ('La velocidad final es de: ');

        IF velocidadfinal > 0 THEN
            WRITE (velocidadfinal:4:2);

        IF velocidadfinal > 0 THEN
            WRITE (' m/s.');
END.


PROGRAM EJER28;
        USES CRT;
        VAR v,g,t:REAL;
BEGIN
     ClrScr;

     g:=9.8;

     WRITE('Velocidad inicial (m/s) -> ');          READLN(v);
     WRITE('Tiempo (s) --------------> ');          READLN(t);

     WRITE('Velocidad instantanea: ');              WRITELN(v-(g*t):5:2);
END.
```

Escribir un programa en Pascal que detecte si un número introducido desde le teclado es positivo o negativo.

```
PROGRAM EJER29;
        USES CRT;
        VAR num:INTEGER;


BEGIN

        ClrScr;

        WRITE ('Introduzca un numero entero: ');        READLN (num);

        IF num > 0 THEN
           WRITE ('El numero es positivo')
        ELSE IF num < 0 THEN
           WRITE ('El numero es negativo')
        ELSE
           WRITE ('El numero no es positivo ni negativo, es 0');
END.
```

Escribir un programa en Pascal que detecte si se han introducido en orden creciente tres números introducidos por el usuario.

```
PROGRAM EJER30;
```

```
        USES CRT;
        VAR num1,num2,num3:INTEGER;
BEGIN
        ClrScr;

        WRITE ('Introduzca un numero (1) : ');  READLN (num1);
        WRITE ('Introduzca un numero (2) : ');  READLN (num2);
        WRITE ('Introduzca un numero (3) : ');  READLN (num3);

        IF ((num1 < num2) AND (num2 < num3)) THEN
           WRITE ('Los numeros se han introducido en orden creciente')
        ELSE
           WRITE ('Los numeros no se han introducido en orden creciente');
END.
```

Escribir un programa en Pascal que detecte el carácter introducido por el usuario.

```
PROGRAM EJER31;
        USES CRT;
        VAR caracter:CHAR;

BEGIN

     ClrScr;

     WRITE ('Introduzca un caracter alfanumerico: ');   READLN (caracter);
     WRITE ('El caracter introducido es -----> ' + caracter)
END.


PROGRAM EJER31;
        USES CRT;
        VAR pato_donald:CHAR;
BEGIN
     ClrScr;

     WRITE('Introduzca un caracter alfanumerico: ');    READLN(pato_donald);
     WRITE('El caracter introducido es ----> ' + pato_donald)
END.
```

Escribir un programa en Pascal que muestre un mensaje afirmativo si el numero introducido es múltiplo de 5.

```
PROGRAM EJER32;
       USES CRT;

       var num:Integer;
Begin
       ClrScr;

       WRITE('Introduzca un numero : ');    READLN(num);
       IF num mod 5 = 0 THEN
              WRITE('El numero introducido es múltiplo de 5')
       ELSE
              WRITE('El numero introducido no es múltiplo de 5');
End.
```

Escribir un programa en Pascal que lea un numero y lo devuelva multiplicado por 5 y dividido por 7.

```
PROGRAM EJER34;
       USES CRT;

       var num, soluc:Real;
Begin
       ClrScr;
        WRITE('Introduzca un numero: ');
       READLN(num);
```

```
        WRITELN;

        {multiplicamos y dividimos el numero obtenido}
        soluc := (num * 5) / 7;

        WRITE('(',num:5:2,' * 5) / 7) = ',soluc:5:2);
        {Poniendo ":5:2" le decimos el formato de salida del numero,
        5 posiciones y de ellas 2 decimales - Prueba a cambiarlo como mas te guste}
End.
```

Escribir un programa en Pascal que determine si un número leído desde el teclado es par o impar.

```
PROGRAM EJER34;
        USES CRT;

        VAR num:INTEGER;

BEGIN
        ClrScr;

        WRITE (Introduzca un numero entero: ');       READLN (num);

        IF num = 0 THEN
           WRITE ('El numero introducido no es par ni impar, es 0')
        ELSE IF ((num mod 2 = 0)) THEN
           WRITE ('El numero introducido es par')
        ELSE
            WRITE ('El numero introducido es impar')
END.
```

Escribir un programa en Pascal que detecte si un número leído desde el teclado es mayor o menor que 100.

```
PROGRAM EJER35;
        USES CRT;

        VAR num:INTEGER;

BEGIN
        ClrScr;

        WRITE ('Escriba un numero entero:');    READLN (num);
        WRITELN ('');


        IF num < 100 THEN
        WRITE ('El numero que ha escrito es menor de 100')
        ELSE IF num > 100 THEN
        WRITE ('El numero que ha escrito es mayor de 100')
        ELSE
        WRITE ('El numero es 100')
END.


PROGRAM EJER35;
        USES CRT;
        VAR num:REAL;

BEGIN
     ClrScr;

     WRITE('Introduzca un numero : ');  READLN(num);

     IF (num <= 100) THEN
         WRITE('NUMERO MENOR O IGUAL A 100 ')
     ELSE
         WRITE('NUMERO MAYOR DE 100')
END.
```

Escribir un programa en Pascal que dado un número del 1 a 7 escriba el correspondiente nombre del día de la semana.

```
PROGRAM EJER36;
        USES CRT;

        VAR num:INTEGER;

BEGIN

     ClrScr;

     WRITE ('Escriba un numero para ver con que dia corresponde: ');
     READLN (num);

     IF num=1 THEN
       WRITE ('Lunes');
     IF num=2 THEN
       WRITE ('Martes');
     IF num=3 THEN
       WRITE ('Miercoles');
     IF num=4 THEN
       WRITE ('Jueves');
     IF num=5 THEN
       WRITE ('Viernes');
     IF num=6 THEN
       WRITE ('Sabado');
     IF num=7 THEN
       WRITE ('Domingo');
END.
```

```
PROGRAM EJER36;
        USES CRT;
        VAR num_dia_sem:INTEGER;
BEGIN
     ClrScr;

     WRITE('Dia de la semana (numero) -> ');    READLN(num_dia_sem);

     CASE num_dia_sem OF
          1: WRITELN('Lunes');
          2: WRITELN('Martes');
          3: WRITELN('Miercoles');
          4: WRITELN('Jueves');
          5: WRITELN('Viernes');
          6: WRITELN('Sabado');
          7: WRITELN('Domingo');
     ELSE
         WRITELN('No es un dia de la semana');
     END;
END.
```

Escribir un programa en Pascal que lea dos números desde el teclado y si el primero es mayor que el segundo intercambie sus valores.

```
PROGRAM EJER37;
        USES CRT;

        VAR num1,num2:INTEGER;
BEGIN
        ClrScr;

        WRITELN ('Escriba dos numeros: ');
        READLN (num1);  WRITE ('');    READLN (num2);
        WRITELN ('');

        IF num1 > num2 THEN
        BEGIN
             WRITE(num2,' ',num1,'. El primer numero introducido es mayor.');
             WRITE(' Se cambia el orden.');
```

```
        END

        ELSE
        BEGIN
             WRITE(num1,' ',num2,'. El segundo numero introducido es mayor.');
             WRITE(' No se cambia el orden.');
        END;
END.


PROGRAM EJER37;
        USES CRT;
        VAR num1,num2,temp:INTEGER;
BEGIN
     ClrScr;

     WRITE('Numero 1: ');       READLN(num1);
     WRITE('Numero 2: ');       READLN(num2);

     IF (num1 > num2) THEN
        BEGIN
             temp:=num1;
             num1:=num2;
             num2:=temp;
             WRITELN('Numero intercambiados');
             WRITE('Numero 1: '); WRITELN(num1);
             WRITE('Numero 2: '); WRITELN(num2);
        END
     ELSE
        BEGIN
             WRITELN('Numeros sin intercambiar');
             WRITE('Numero 1: '); WRITELN(num1);
             WRITE('Numero 2: '); WRITELN(num2);
        END;
END.
```

Escribir un programa en Pascal que dada una calificación en valor alfabético (A,B,C,D ó E) indique su equivalente en valor numérico (4,5,6,7 u 8).

```
PROGRAM EJER38;
        USES CRT;

        VAR valor:CHAR;
BEGIN
        ClrScr;

        WRITE ('Escriba una calificacion entre a y e: ');
        READLN (valor);
        WRITELN ('');

        CASE UPCASE(valor) OF
                'A': WRITE ('El valor correspondiente es: 4');
                 'B': WRITE ('El valor correspondiente es: 5');
                'C': WRITE ('El valor correspondiente es: 6');
                 'D': WRITE ('El valor correspondiente es: 7');
                'E': WRITE ('El valor correspondiente es: 8')
             ELSE
                WRITE ('La calificacion no existe');
        END;
END.


PROGRAM EJER38;
        USES CRT;
        VAR cal:CHAR;
BEGIN
     ClrScr;

     WRITE('Introduzca una calificacion (A-E):');
     READLN(cal);

     CASE cal OF
          'A': WriteLn('Calificacion numerica --> 4');
```

```
          'B': WriteLn('Calificacion numerica --> 5');
          'C': WriteLn('Calificacion numerica --> 6');
          'D': WriteLn('Calificacion numerica --> 7');
          'E': WriteLn('Calificacion numerica --> 8');
     ELSE
         WriteLn('Calificacion incorrecta');
     END;
END.
```

Escribir un programa en Pascal que lea desde teclado el importe bruto de una factura y determine el importe neto según los siguientes criterios.

- Importe bruto menor de 20.000 -> sin descuento
- Importe bruto mayor de 20.000 -> 15% de descuento

```
PROGRAM EJER39;
        USES CRT;

        VAR importe_bruto:REAL;
        VAR descuento, total:REAL;

BEGIN
        ClrScr;

        WRITE ('Indique el importe de su factura para ver ');
        WRITELN ('si le "descontamos" algo');
        WRITELN ('');
        READLN (importe_bruto);
        WRITELN ('');

        {calcula el importe bruto con descuento del 15%}
        descuento:=importe_bruto * 0.15;

        IF importe_bruto > 20000 THEN

        BEGIN
             WRITELN ('SE MERECE UN DESCUENTO DE: ',descuento:5:2, ' PTS');
             total:=importe_bruto - descuento;
             WRITELN ('El total es de la factura es de: ',total:5:2,' pts')
        END

        ELSE
             WRITE ('CON ESE DINERO NO SE MERECE UN DESCUENTO')

END.


PROGRAM EJER39;
        USES CRT;
        VAR imp_bru,imp_net:REAL;
BEGIN
     ClrScr;

     WRITE('Importe Bruto -> ');        READLN(imp_bru);

     IF imp_bru <= 20000 THEN
        imp_net:=imp_bru
     ELSE
        imp_net:=imp_bru-(0.15*imp_bru);

     WRITE('Importe a pagar: ');        WRITE(imp_net:5:2)
END.
```

Escribir un programa en Pascal que una vez leída una hora en formato (horas, minutos, segundos) indique cual será el tiempo dentro de un segundo.

```
PROGRAM EJER40;
        USES CRT;
        {Las variables son: horas, minutos y segundos}
```

```
        {Son las horas, minutos y segundos introducidos por el usuario}
        VAR h, m, s:INTEGER;
        VAR h2,m2,s2:INTEGER;
        {Son las horas, minutos y seguntos a los que se les sumara}
BEGIN
     ClrScr;

     WRITE ('Escriba en formato horas, minutos y segundos');
     WRITELN ('');
     WRITE ('Horas ');          READLN (h);
     WRITE ('Minutos ');       READLN (m);
     WRITE ('Segundos ');    READLN (s);
     WRITELN ('');
     WRITELN ('Se le sumara un segundo a la hora actual.');
     WRITELN ('');

     s:= s + 1;

     IF s = 60 THEN
        s2 := 0
     ELSE
        s2 := s;

     m:= ((m * 60) + s) div 60;

     IF m = 60 THEN
        m2 := 0
     ELSE
        m2 := m;

     h2:=((h * 60) + m) div 60;

     IF h2 = 24 THEN
        h2 := 0;

     WRITELN (h2,':',m2,':',s2);
END.


PROGRAM EJER40;
        USES CRT;
        VAR h1,m1,s1:INTEGER;
        VAR h2,m2,s2:INTEGER;
BEGIN
     Clrscr;

     WRITE('Horas ------> ');   READLN(h1);
     WRITE('Minutos ----> ');   READLN(m1);
     WRITE('Segundos ---> ');   READLN(s1);

     s2:=s1+1;

     IF s2=60 THEN
     BEGIN
          s2:=0;
          m2:=m1+1;
     END;

     IF m2=60 THEN
     BEGIN
          m2:=0;
          h2:=h1+1;
     END;

     IF h2=24 THEN
     BEGIN
          s2:=0;
          m2:=0;
          h2:=0;
     END;

     WRITE(h1); WRITE(' hh ');
     WRITE(m1); WRITE(' mm ');
     WRITE(s1); WRITE(' ss ');

     WRITE(' + 1 segundo son: ');
```

```
     WRITE(h2); WRITE(' hh ');
     WRITE(m2); WRITE(' mm ');
     WRITE(s2); WRITE(' ss ');
END.
```

Escribir un programa en Pascal que calcule el salario semanal de un trabajador en base a las horas trabajadas y el pago por hora trabajada.

- Horas ordinarias (40 primeras horas de trabajo) – 2.000 Pts/hora
- 1.5 veces precio hora ordinaria

```
PROGRAM EJER41;
        USES CRT;

        VAR htrab, ptsh:REAL; {Horas trabajadas y pts hora}
        VAR nhextra, hextra:REAL; {Numero de horas extra y horas extra}
        VAR salario_semanal:REAL;
BEGIN
     ClrScr;

     WRITE ('Introduzca las horas trabajadas y las pts/hora que se cobran ');
     WRITELN ('para calcular el salario semanal.');
     WRITELN ('');

     WRITE ('Horas trabajadas: ');   READLN (htrab);
     WRITE ('Pts/hora: ');                  READLN (ptsh);
     WRITE ('Horas extra: ');               READLN (nhextra);
     WRITELN ('');

     hextra:=nhextra * (ptsh * 1.5);
     Salario_semanal:= (htrab) * (ptsh) + hextra;

     WRITE ('El salario semanal son ',salario_semanal:5:0,' pts.');
END.
```

```
PROGRAM EJER41;
        USES CRT;
        VAR pre_hor,hor_tra,hor_ext,sal_sem:REAL;
BEGIN
     ClrScr;

     pre_hor:=2000;

     WRITE('Horas trabajadas '); READLN(hor_tra);

     IF hor_tra<=40 THEN
        sal_sem:=hor_tra*pre_hor
     ELSE
     BEGIN
          hor_ext:=hor_tra-40;
          sal_sem:=(40*pre_hor)+(hor_ext*(pre_hor*1.5));
     END;

     WRITE('Salario semanal: '); WRITELN(sal_sem:5:2);
END.
```

Escribir un programa en Pascal que realice un bucle con While y muestre en pantalla del 1 al 10.

```
PROGRAM EJER42;
        USES CRT;
        VAR x:INTEGER;
BEGIN
     x:=0;

     ClrScr;
```

```
     WHILE X <= 10 DO
     BEGIN
          WRITELN (x);
          x:=x+1;

     END;
END.
```

Escribir un programa en Pascal que realice un bucle con Repeat y muestre en pantalla del 1 al 10.

```
PROGRAM EJER43;
        USES CRT;
        VAR x:INTEGER;
BEGIN
     x:=0;

     ClrScr;

     REPEAT
           WRITELN (x);
           x:=x+1;
     UNTIL x=10;
END.
```

Escribir un programa en Pascal que realice un bucle con For y muestre en pantalla del 1 al 10.

```
PROGRAM EJER44;
        USES CRT;
        VAR x:INTEGER;
BEGIN
     ClrScr;

     FOR x:=0 TO 10 DO
         WRITELN(x);
END.
```

Escribir un programa en Pascal que visualice en pantalla los números pares entre 1 y 25.

```
PROGRAM EJER45;
        USES CRT;

        VAR num:INTEGER;


BEGIN

     num:=2;

     ClrScr;

     REPEAT

           WRITELN (num);
           num:= num + 2;
     UNTIL num= 26;

END.


PROGRAM EJER45;
        USES CRT;
        VAR sem:INTEGER;
BEGIN
     ClrScr;
```

```
     sem:=1;

     WHILE sem <= 25 DO
     BEGIN
          IF sem mod 2=0 THEN
             WRITELN(sem);
             sem:=sem + 1;
          END;
END.
```

Escribir un programa en Pascal que visualice en pantalla los números múltiplos de 5 comprendidos entre 1 y 100.

```
PROGRAM EJER46;
        USES CRT;

        VAR num:INTEGER;
BEGIN
     num:= 5;

     ClrScr;

     WHILE num <= 100 DO
     BEGIN
          WRITELN (num);
          num:= num + 5;
     END;
END.
```

```
PROGRAM EJER46;
        USES CRT;
        VAR sem:INTEGER;
BEGIN
     ClrScr;

     FOR sem:= 1 TO 100 DO
     BEGIN
          IF sem mod 5=0 THEN
             WRITELN(sem);
     END;
END.
```

Escribir un programa en Pascal que sume los números comprendidos entre 1 y 10.

```
PROGRAM EJER47;
        USES CRT;

        VAR num, x:INTEGER;
BEGIN
     ClrScr;

     num:=1;
     x:=1;

     WHILE num <= 10 DO
     BEGIN
          WRITELN (x);
          num:= num + 1;
          x:= x + num;
     END;
END.
```

Escribir un programa en Pascal que genere la tabla de multiplicar de un número introducido por el teclado.

```
PROGRAM EJER48;
        USES CRT;

        VAR tabla, x, num:INTEGER;
BEGIN
     ClrScr;

     WRITE ('Introduzca un numero para hacer su tabla de multiplicar: ');

     READLN (num);      WRITELN ('');

     REPEAT
           WRITELN (tabla);
           x:= x + 1;
           tabla:= num * x;
     UNTIL x=11;
END.
```

```
PROGRAM EJER48;
        USES CRT;
        VAR num,sem:INTEGER;
BEGIN
     ClrScr;

     WRITE('Introduzca un numero entero: ');    READLN(num);

     FOR sem:=1 TO 10 DO
     BEGIN
          WRITE(num); WRITE(' * '); WRITE(sem); WRITE(' = ');
          WRITELN(num*sem);
     END;
END.
```

Escribir un programa en Pascal que realice la pregunta ¿Desea continuar S/N? y que no deje de hacerla hasta que el usuario teclee N.

```
PROGRAM EJER49;
        USES CRT;

        VAR respuesta:CHAR;

BEGIN

     ClrScr;

     REPEAT
         WRITELN ('DESEA CONTINUAR: S/N ');  READLN (respuesta);
     UNTIL respuesta='N';
     {Pascal distingue entre ‘N’ y ‘n’}

END.
```

```
PROGRAM EJER49B;
        USES CRT;

        VAR respuesta:STRING;

BEGIN

     ClrScr;

     REPEAT
         WRITELN ('DESEA CONTINUAR: SI/NO '); READLN (respuesta);
     UNTIL respuesta='NO';

END.
```

```
PROGRAM EJER49;
        USES CRT;
```

```
        VAR resp:CHAR;

BEGIN
     ClrScr;

     resp:='S';

     WHILE UPCASE(resp)='S' DO
     BEGIN
          WRITE('Desea continuar? '); READLN(resp);
     END;
END.
```

Escribir un programa en Pascal que calcule cuantos años tarda en duplicarse un capital depositado al 5% de interés anual

```
PROGRAM EJER50;
        USES CRT;

        VAR tiempo:REAL;
        VAR cap_ini,cap_fin:REAL;
        CONST interes = 0.05;
BEGIN
     ClrScr;

     WRITE ('Intruduzca el capital para calcular cuanto tardara ');
     WRITE ('en duplicarse, con un interes del 5%: ');

     READLN (cap_ini);
     WRITELN ('');

     IF cap_ini < 0 THEN
     BEGIN
        WRITE ('No se puede incluir un capital negativo');
        EXIT;
     END;

     tiempo:=0;
     cap_fin:= cap_ini;

     REPEAT
            cap_fin:=cap_fin + (cap_fin * interes);
            tiempo:= tiempo + 1;
     UNTIL cap_fin > (cap_ini * 2);

     WRITELN ('Tardara',tiempo:3:0,' años en duplicarse');
     WRITELN ('Capital final: ',cap_fin:5:2,' pts');
END.
```

```
PROGRAM EJER50;
        USES CRT;
        VAR cap_ini,cap_fin:REAL;
        VAR num_year:INTEGER;
        const INTERES=0.05;
BEGIN
     ClrScr;
     num_year:=0;

     WRITE('Capital inicial -----: ');  READLN(cap_ini);

     cap_fin:=cap_ini;

     WHILE cap_fin < (cap_ini*2) DO
     BEGIN
          cap_fin:=cap_fin+(cap_fin*interes);
          num_year:=num_year + 1;
     END;

     WRITE('Capital inicial -----: ');  WRITELN(cap_ini:5:2);
     WRITE('Capital final -------: ');  WRITELN(cap_fin:5:2);
     WRITE('Capital duplicado en ');     WRITE(num_year);  WRITE(' años');
END.
```

Escribir un programa que calcule la suma de los números hasta un número dado (introducido por el usuario).

```
PROGRAM EJER51;
        USES CRT;

        VAR x, y, num:INTEGER;
BEGIN
     ClrScr;

     WRITE ('Este programa calcula la suma de los numeros hasta uno ');
     WRITE ('introducido por el usuario: ');

     READLN (num);       WRITELN ('');
     x:=0;

     WHILE num >= 0 DO
     BEGIN
          WRITELN (x);
          x:= x + num;
          num:=num - 1;
     END;
END.
```

```
PROGRAM EJER51;
        USES CRT;
        VAR i,num:INTEGER;
        VAR suma:LONGINT;
BEGIN
     ClrScr;

     WRITE('Introduzca un numero -> '); READLN(num);

     FOR i:=0 TO num DO
         suma:=suma+ i;

     WRITE('Suma '); WRITE('0-'); WRITE(num); WRITE('---->'); WRITE(suma);
END.
```

Escribir un programa que pida un número y si el que se introduce por el teclado es menor de 100 que vuelva a solicitarlo.

```
PROGRAM EJER52;
        USES CRT;

        VAR num:INTEGER;
BEGIN
{Este programa no finaliza hasta que se escribe un numero mayor a 100}

     ClrScr;

     REPEAT
           WRITELN ('Introduzca un numero: ');
           READLN (num);
     UNTIL num > 100;
END.
```

```
PROGRAM EJER52;
        USES CRT;
        VAR num:INTEGER;
BEGIN
     ClrScr;

     WRITE('Introduzca un numero -> '); READLN(num);

     WHILE num<=100 DO
     BEGIN
          WRITE('Introduzca un numero -> '); READLN(num);
     END;
```

```
END.
```

Escribir un programa en Pascal que calcule el factorial de un número.

```
PROGRAM EJER53;
        USES CRT;

        VAR factorial, x, num, y:REAL;
BEGIN
     {Este programa hace el factorial de un numero}

     ClrScr;

     WRITE ('Introduzca un numero para hacer su factorial: ');

     READLN (num);      WRITELN ('');

     x:=1;

     WHILE num > 1 DO
     BEGIN
           x:=x * num;
           num:=num - 1;
           WRITELN (x);
     END;
END.


PROGRAM EJER53;
        USES CRT;
        VAR temp,num,fac:LONGINT;
BEGIN
     ClrScr;

     fac:=1;
     temp:=num;

     WRITE('Introduzca un numero -> '); READLN(num);

     temp:=num;

     WHILE num>=1 DO
     BEGIN
          fac:=fac*num;
          num:=num-1;
     END;

     WRITE('El factorial de '); WRITE(temp); WRITE(' es '); WRITE(fac);
END.
```

Escribir un programa en Pascal que calcule la media de 5 números introducidos por el teclado.

```
PROGRAM EJER54;
        USES CRT;
        VAR n1, n2, n3, n4, n5:REAL;
        VAR resultado:REAL;
BEGIN
     ClrScr;

     WRITELN ('Introduzca 5 numeros para hacer su media');
     WRITELN ('');

     WRITE ('Nº 1: ');  READLN (n1);
     WRITE ('Nº 2: ');  READLN (n2);
     WRITE ('Nº 3: ');  READLN (n3);
     WRITE ('Nº 4: ');  READLN (n4);
     WRITE ('Nº 5: ');  READLN (n5);
     WRITELN ('');

     resultado:= (n1 + n2 + n3 + n4 + n5) / 2;
```

```
     WRITE (resultado:5:2);
END.

PROGRAM EJER54;
        USES CRT;
        VAR i:INTEGER;
        VAR num,suma:REAL;
BEGIN
     ClrScr;

     i:=0;

     REPEAT
           WRITE('Introduzca un numero: ');     READLN(num);
           suma:=suma+num;
           i:=i+1;
     UNTIL i=5;

     WRITE('La media es: ');    WRITELN(suma/i:5:2);
END.
```

Escribir un programa en Pascal que calcule el salario neto semanal de un trabajador en función del número de horas trabajadas y la tasa de impuestos de acuerdo a las siguientes hipótesis.

- Las primeras 35 horas se pagan a tarifa normal
- Las horas que pasen de 35 se pagan 1.5 veces la tarifa normal
- Las tasas de impuestos son:
  - a: Los primeros 50 dólares son libres de impuestos
  - b: Los siguientes 40 dólares tienen un 25% de impuestos
  - c: Los restantes de 45% de impuestos

```
PROGRAM EJER55;
        USES CRT;

        VAR sns:REAL; {salario neto semanal}
        VAR h_trabajadas, h_extra:REAL;
        VAR precio_h, precio_h_extra:REAL;
        VAR total:REAL;
        VAR impuestos0,impuestos25,impuestos45:REAL;
        {Impuestos con cada % correspondiente}
BEGIN
     ClrScr;

     WRITELN ('INTRODUZCA LOS DATOS PARA CALCULAR EL SALARIO NETO SEMANAL');
     WRITELN ('');
     WRITE ('Las horas trabajadas semanales son 35, las demas debe ');
     WRITELN ('considerarlas como horas extra. Escriba en dolares.');
     WRITELN ('');
     WRITE ('Horas trabajadas: ');          READLN (h_trabajadas);
     WRITE ('Horas extra: ');                       READLN (h_extra);
     WRITE ('Precio por cada hora: ');      READLN (precio_h);
     WRITELN ('');

     precio_h_extra:=precio_h * 1.5;
     sns:=(h_trabajadas * precio_h) + (h_extra * precio_h_extra);

     impuestos0:=0;
     impuestos25:=sns - ((sns - 50) * 0.25);
     impuestos45:=sns - ((sns - 90) * 0.45);

     IF sns <= 50 THEN
     WRITELN ('El salario neto semanal es: ',sns:5:2)
     ELSE IF sns < 90 THEN
     WRITELN ('El salario neto semanal es: ',impuestos25:5:2)
     ELSE IF sns > 90 THEN
     WRITELN ('El salario neto semanal es: ',impuestos45:5:2);
END.
```

```
PROGRAM EJER55;
        USES CRT;
        VAR hor_tra,sal_bru,tas_imp,sal_net:real;
        CONST tar_hor=2;
        CONST tasa_imp1=0.25;
        CONST tasa_imp2=0.45;
BEGIN
     ClrScr;

     WRITE('Numero de horas trabajadas: ');     READLN(hor_tra);

     {Calculo del salario bruto}

     IF hor_tra <= 35 THEN
        sal_bru:=hor_tra*tar_hor
     ELSE
        sal_bru:=(35*tar_hor)+((hor_tra-35)*(1.5*tar_hor));

     {Calculo de impuestos}

     IF sal_bru <= 50 THEN
        tas_imp:=0
     ELSE IF sal_bru <= 90 THEN
        tas_imp:=(sal_bru-50)*tasa_imp1
     ELSE
        tas_imp:=(40*tasa_imp1)+((sal_bru-90)*tasa_imp2);

     {Calculo salario neto}

     sal_net:=sal_bru-tas_imp;

     WRITE('Horas trabajadas -------> ');       WRITELN(hor_tra:5:2);
     WRITE('Salario bruto-----------> ');       WRITELN(sal_bru:5:2);
     WRITE('Impuestos---------------> ');       WRITELN(tas_imp:5:2);
     WRITE('Salario neto------------> ');       WRITELN(sal_net:5:2);
END.
```

Escribir un programa en Pascal que detecte si un número es primo o no. Un número es primo si sólo es divisible por sí mismo y por la unidad.

Ejemplo: 2,3,4,7,11,17,19 son números primos
9 no es número primo, es divisible por 1, 9, 3

El algoritmo para resolver este problema pasa por dividir sucesivamente el número estudiado por 2,3,4, etc., hasta el propio número.

```
Program primo (INPUT,OUTPUT);
     Uses Crt;

     Var
        num: Word; {Los numeros primos son enteros mayores que 1 sin
                    divisores enteros positivos, exceptuando el 1 y ellos
                    mismos. Todos los primos son impares, excepto el 2.
                    Solo es necesario comprobar la divisibilidad por
                    numeros superiores a la raiz cuadrada del numero.}
        raiznum: Integer;   {Guardamos el valor de la raiz del numero}
        noesprimo: Boolean; {Variable para decir que un numero no es primo}
        par: Boolean;       {Nos sirve para marcar los numeros que son pares}
        i: Byte;            {Variable que usamos dentro del bucle}

Begin {p.p}
     ClrScr;

     Repeat
        Write('Introduzca un numero entero para ver si es primo: ');
        Read(num);
        Writeln;
     Until num > 1; {Pedimos un numero y no lo aceptamos hasta que sea > 1}

     par := num mod 2 = 0; {par va a ser True cuando el numero sea par}
```

```
     If num = 2 Then
        Write('El 2 es primo, es el unico numero par que lo es.')

     Else if par Then
        Write('Todo numero par no es primo, excepto el 2.')

     Else
        Begin
           raiznum := round(sqrt(num));
           {Almacenamos la raiz del numero, redondeada}

           For i := raiznum To (num - 1) Do
             Begin
                If (num mod i) = 0 Then
                   noesprimo := true
             End;
             {Comprobamos la divisibilidad de los numeros superiores
              a la raiz cuadrada del numero introducido. Si algun numero
              divide al numero, noesprimo toma el valor true.}

           If noesprimo Then
               Write(num,' no es un numero primo')
           Else
               Write(num,' es un numero primo');
           {Mostramos por pantalla si el numero es primo o no}
        End;

End. {p.p}



PROGRAM EJER56;
        USES CRT;
        VAR i,num,flag:INTEGER;

BEGIN
     ClrScr;
     flag:=0;

     WRITE('Introduzca un numero -> '); READLN(num);

     FOR i:=2 TO (num-1) DO
     BEGIN
          IF (num mod i)=0 THEN
          flag:=1;
     END;

     IF flag=1 THEN
     BEGIN
          WRITE(num);   WRITE(' no es un numero primo');
     END
     ELSE
     BEGIN
          WRITE(num);   WRITE(' es un numero primo');
     END;
END.
```

Escribir un programa en Pascal que calcule la depreciación de un objeto según el método de la línea recta. Calcular el número de años que tarda en hacerse 0. En este método el valor original del objeto se divide por su vida (número de años). El cociente resultante será la cantidad en la que el objeto se deprecia anualmente. Por ejemplo, si un objeto se deprecia 8000 dólares en diez años, entonces la depreciación anual será 8000/10=800 dólares. Por tanto, el valor del objeto habrá disminuido en 800 dólares cada año. Nótese que la depreciación anual es la misma cada año cuando se utiliza este método.

```
PROGRAM EJER57;
        USES CRT;

        VAR annos, valor:REAL;
```

```
        VAR depreciacionanno:REAL;
        VAR x:INTEGER;
BEGIN
     ClrScr;

     WRITELN ('Escriba los datos para conocer la depreciacion anual');
     WRITELN ('');

     WRITE ('Valor original: ');             READLN (valor);
     WRITE ('Numero de años: ');    READLN (annos);
     WRITELN ('');

     depreciacionanno:= valor / annos;
     x:=0;

     WHILE valor > 0.1 DO
     BEGIN
            valor:=valor - depreciacionanno;
            x:= x + 1;
            WRITELN ('AÑO ',x:2,': ',valor:8:0,' pts.');
     END;
END.



PROGRAM EJER57;
        USES CRT;

        VAR val_ini,val_dep,tas_dep:REAL;
        VAR i,anios:INTEGER;
BEGIN
     ClrScr;

     WRITE('Valor inicial -----> ');    READLN(val_ini);
     WRITE('Numero de a os ----> ');    READLN(anios);

     tas_dep:=val_ini/anios;;
     val_dep:=val_ini-tas_dep;

     FOR I:=1 TO anios DO
     BEGIN
          WRITE('A o ');
          WRITE(i:3);
          WRITE(' ------ ');
          WRITELN(val_dep:10:2);
          val_dep:=val_dep-tas_dep;
     END;
END.
```

Escribir un programa en Pascal que calcule la depreciación de un objeto según el método del balance doblemente declinante. En este método, el valor del objeto disminuye cada año en un porcentaje constante. Por tanto, la verdadera cantidad depreciada, en dólares, variara de un año al siguiente. Para obtener el factor de depreciación, dividimos por dos la vida del objeto. Este factor se multiplica por el valor del objeto al comienzo de cada año (y no el valor del original del objeto) para obtener la depreciación anual. Supongamos, por ejemplo que deseamos depreciar un objeto de 8000 dólares por diez años; el factor de depreciación será 2/10=0.2. Por tanto, la depreciación el primer año será 0,2 X 8000 = 1600 dólares, la depreciación del segundo año será 0,2 X 6400=1280 dólares; la depreciación del tercer año será 0,2 X 5120 = 1024 dólares, y así sucesivamente.

```
PROGRAM EJER58;
        USES CRT;
        VAR fdd:REAL; {Factor de depreciacion}
        VAR vidaobj:REAL; {Vida del objeto = años}
        VAR depreciacionanno:REAL;
        VAR valorobj:REAL; {Valor del objeto}
        VAR x:INTEGER;
BEGIN
```

```
     ClrScr;
     WRITE ('PARA CALCULAR LA DEPRECIACION POR EL METODO DEL BALANCE ');
     WRITELN ('DOBLEMENTE DECLINANTE ESCRIBA LOS DATOS:');
     WRITELN ('');

     WRITE ('VALOR DEL OBJETO: ');          READLN (valorobj);
     WRITE ('AÑOS: ');                      READLN (vidaobj);
     WRITELN ('');

     fdd:= 2 / vidaobj;
     x:=1;

     REPEAT
          depreciacionanno:= fdd * valorobj;
          valorobj:=valorobj - depreciacionanno;
          WRITELN ('AÑO ',x:2,': ',valorobj:8:2,' pts.');
          x:= x + 1;
     UNTIL x > vidaobj;

     WRITELN ('');
     x:= x - 1;
     WRITELN ('EL VALOR A LOS ',x,' A OS SERA DE: ',valorobj:6:2,' pts.');
END.

PROGRAM EJER58;
        USES CRT;

        VAR val_ini,val_dep,coe_dep,tas_dep:REAL;
        VAR i,anios:INTEGER;
BEGIN
     ClrScr;

     WRITE('Valor inicial ----->');     READLN(val_ini);
     WRITE('Numero de a os ----> ');    READLN(anios);

     coe_dep:=2/anios;
     tas_dep:=val_ini*coe_dep;
     val_dep:=val_ini-tas_dep;

     FOR i:=1 TO anios DO
     BEGIN
          WRITE('A o ');
          WRITE(i:3);
          WRITE(' ------ ');
          WRITELN(val_dep:10:2);
          val_dep:=val_dep-(val_dep*coe_dep);
     END;
END.
```

Escribir un programa que calcule las raíces de la ecuación ($ax^2 + bx + c = 0$) teniendo en cuenta los siguientes casos:

a. Si a es igual a 0 y b es igual a 0, imprimiremos un mensaje diciendo que la ecuación es degenerada.
b. Si a es igual a 0 y b no es igual a 0, existe una raíz única con valor –c/b.
c. En los demás casos, utilizaremos la fórmula siguiente:
   xi = (-b ± √(b2 – 4ac))/(2a)
   La expresión $d = b^2 – 4ac$ se denomina discriminante.
   · Si d es mayor o igual que 0 entonces hay dos raíces reales
   · Si d es menor que 0 entonces hay dos raíces complejas de la forma: x+yi,x-yi.
   Siendo x el valor –b/2a e y el valor absoluto de $\sqrt{(b^2-4ac)}/(2a)$

```
PROGRAM EJER59;
        USES CRT;
        VAR a,b,c,d,r1,r2,x,y:REAL;
BEGIN
     ClrScr;
```

```
     WRITE('Coeficiente a -> ');        READLN(a);
     WRITE('Coeficiente b -> ');        READLN(b);
     WRITE('Coeficiente c -> ');        READLN(c);

     IF (a=0) AND (b=0) THEN
     BEGIN
          WRITE('La ecuacion es degenerada');
          EXIT;
     END
     ELSE IF (a=0) AND (b<>0) THEN
     BEGIN
          r1:=c/b;
          WRITE('r1 = '); WRITELN(r1:5:2);
          EXIT;
     END
     ELSE
     BEGIN
          d:=sqr(b)-4*a*c;
          IF (d>=0) THEN
          BEGIN
               r1:=(-b+sqrt(sqr(b)-4*a*c))/(2*a);
               r2:=(-b-sqrt(sqr(b)-4*a*c))/(2*a);
               WRITE('r1 = ');      WRITELN(r1:5:2);
               WRITE('r2 = ');      WRITELN(r2:5:2);
          END
          ELSE
          BEGIN
               x:=-b/(2*a);
               y:=-b-sqrt(abs(sqr(b)-4*a*c));
               WRITE('r1 = '); WRITE(x:5:2); WRITE(' + '); WRITE(y:5:2);
               WRITELN('i');
               WRITE('r2 = '); WRITE(x:5:2); WRITE(' - '); WRITE(y:5:2);
               WRITELN('i');
          END;
     END;
END.
```

Escribir un programe en Pascal que resuelva una matriz 3 x 3. (Veremos formas mejores de realizar este ejercicio)

```
PROGRAM DETER3;
        USES CRT;

        VAR a11,a12,a13,a21,a22,a23,a31,a32,a33:REAL;
        VAR M:REAL;

BEGIN

     ClrScr;

     WRITELN ('Introduzca los valores de la matriz 3x3');
     WRITELN (' ');

     WRITE ('a11: ');   READLN (a11);
     WRITE ('a12: ');   READLN (a12);
     WRITE ('a13: ');   READLN (a13);
     WRITE ('a21: ');   READLN (a21);
     WRITE ('a22: ');   READLN (a22);
     WRITE ('a23: ');   READLN (a23);
     WRITE ('a31: ');   READLN (a31);
     WRITE ('a32: ');   READLN (a32);
     WRITE ('a33: ');   READLN (a33);

     WRITELN (' ');

     M:=(a11*a22*a33)+(a21*a32*a13)+(a31*a12*a23)
       -(a13*a22*a31)-(a11*a23*a32)-(a12*a21*a33);

     WRITE ('El resultado es: ');        WRITE (M:5:2);

END.
```

Escribir un programa en Pascal que sume dos cadenas. (los datos vienen en el cuerpo del ejercicio).

```
PROGRAM EJER60;
        USES CRT;

        VAR tit_pel1:STRING;
        VAR tit_pel2:STRING;
        VAR tit_pel3:STRING;
BEGIN
     tit_pel1:='Alien';
     tit_pel2:='Blade ';
     tit_pel3:='Runner';

     ClrScr;

     WRITELN ('TITULOS DE PELICULAS');

     WRITELN (tit_pel1);
     WRITELN (tit_pel2 + tit_pel3);

END.
```

A partir de las cadenas de caracteres 70809207 y Q, construir y visualizar en la pantalla la cadena 70809207-Q.

```
PROGRAM EJER61;
        USES CRT;

        VAR numero:LONGINT;
        VAR letra:STRING;
BEGIN
     ClrScr;

     numero:=56789312;
     letra:='F';

     WRITE ('EL NUMERO DEL DNI ES: ');
     WRITE (numero,'-',+ letra);
END.


PROGRAM EJER61;
        USES CRT;
        VAR cad1,cad2:STRING;
BEGIN
     cad1:='56789312';
     cad2:='X';

     ClrScr;

     WRITE('NIF:' + cad1 + '-' + cad2);
END.
```

Transformar la cadena *Esto es una prueba*, en la cadena *Esto es prueba*. Utilizar la función DELETE(S,P,N) que elimina N caracteres de la cadena S, a partir de la posición P.

```
PROGRAM EJER62;
        USES CRT;

        VAR frase:STRING;
BEGIN
     ClrScr;

     {Transforma la cadena "Esto es una prueba" en "Esto es prueba"}

     WRITELN ('Este programa suprime un numero determinado de letras');
     WRITELN ('');
```

```
     frase:='Esto es una prueba';
     WRITELN (frase);

     DELETE (frase,9,4);
     WRITELN (frase);
END.
```

Transformar la cadena *Curso de Pascal* en la cadena *Curso de Programación en Pascal.* Utilizar la función INSERT(SO,SD,P) que inserta la cadena SO en la cadena SD a partir de la posición P.

```
PROGRAM EJER63;
        USES CRT;

        VAR frase1:STRING;
        VAR frase2:STRING;
BEGIN
     ClrScr;

     {Este programa inserta la frase2 en la frase1}

     WRITELN ('Se introducira la frase2 en la frase1');
     WRITELN ('');

     frase1:='Curso de Pascal';
     frase2:='Programacion en ';

     WRITE ('Frase1: ');      WRITELN (frase1);
     WRITE ('Frase2: ');      WRITELN (frase2);
     WRITELN ('');

     insert (frase2,frase1,10);

     WRITELN (frase1);
END.
```

Transformar los valores 91 y 8631217 almacenados en dos variables de tipo REAL en la cadena 91-8631217. Previamente transformar los dos valores numéricos a variables tipo cadena con la función STR(X,S) que almacena en S, como una cadena alfanumérica, el valor X.

```
PROGRAM EJER64;
        USES CRT;

        VAR valor1, valor2:REAL;
        VAR valor_1, valor_2:STRING;
BEGIN
     ClrScr;

     valor1:=91;
     valor2:=5550908;

     STR(valor1:5:2, valor_1);
     STR(valor2:10:2, valor_2);

     WRITELN (valor_1,' - ', + valor_2);
END.
```

```
PROGRAM EJER64;
        USES CRT;
        VAR val1,val2:REAL;
        VAR cad1,cad2:STRING;
BEGIN
     val1:=91;
     val2:=5550908;
```

```
     STR(val1,cad1);
     STR(val2,cad2);

     ClrScr;

     WRITELN(cad1 + '-' + cad2);
END.
```

Concatenar las cadenas *El*, *hombre*, *invisible* en una sola *el hombre invisible*. Utilizar la función CONCAT (S1,S2,S3...) que retorna la concatenación de todas las cadenas pasadas como parámetros.

```
PROGRAM EJER65;
        USES CRT;

        VAR S1, S2, S3:STRING;
        VAR cadena_final:STRING;
BEGIN
     ClrScr;

     S1:='El';
     S2:=' hombre';
     S3:=' invisible';

     cadena_final:=CONCAT(S1,S2,S3);

     WRITE (cadena_final);
END.
```

```
PROGRAM EJER65;
        USES CRT;
        VAR cad1,cad2,cad3,cad4,esp:STRING;
BEGIN
     cad1:='el';
     cad2:='hombre';
     cad3:='invisible';
     esp:=' ';

     cad4:=CONCAT(cad1,esp,cad2,esp,cad3);

     ClrScr;

     WRITELN(cad1);
     WRITELN(cad2);
     WRITELN(cad3);
     WRITELN(cad4);
END.
```

Extraer la cadena SOFIA de la cadena FILOSOFIA. Utilizar la función COPY(S,P,N), que devuelve una subcadena de S, de N caracteres, el primero de los cuales ocupa la posición P de S.

```
PROGRAM EJER66;
        USES CRT;

        VAR S1,S2:STRING;
BEGIN
     ClrScr;

     S1:='FILOSOFIA';

     S1:=COPY (S1,5,5); {'SOFIA'}

     WRITE (S1);
END.
```

```
PROGRAM EJER66;
```

```
        USES CRT;
        VAR cad1,cad2:STRING;
BEGIN
     cad1:='FILOSOFIA';
     cad2:=COPY(cad1,5,5);

     ClrScr;

     WRITELN(cad1);
     WRITELN(cad2);
END.
```

Obtener la longitud de la cadena *esternocleidomastoideo*, utilizando la función LENGTH(S), que devuelve la longitud de la cadena S.

```
PROGRAM EJER67;
        USES CRT;

        VAR cadena:STRING;
BEGIN
     ClrScr;

     WRITELN ('Introduzca una cadena para saber su longitud:');
     WRITELN ('');

     READLN (cadena);

     WRITELN ('');
     WRITE ('La longitud de la cadena es de: ');

     WRITE (LENGTH(cadena),' caracteres');
END.
```

Obtener la primera posición en la que comienza la subcadena *fragi* en la cadena *supercalifragilisticoexpialidoso*. Utilizar la función POS(SB,S) que devuelve la posición de la primera aparición de la subcadena SB en la cadena S.

```
PROGRAM EJER68;
        USES CRT;

        VAR cadena:STRING;
        VAR subcadena:STRING;
        VAR posi:INTEGER;
BEGIN
     ClrScr;

     cadena:='supercalifragilisticoexpialidoso';
     subcadena:='fragi';

     posi:=POS(subcadena,cadena);

     WRITE (posi);
END.
```

Transformar la cadena Lenguaje Pascal en la cadena LENGUAJE PASCAL. Utilizar la función UPCASE(C) que transforma un carácter C a mayúscula.

```
PROGRAM EJER69;
        USES CRT;

        VAR cadena:STRING;
        VAR i:INTEGER;
BEGIN
     ClrScr;

     cadena:='Lenguaje Pascal';
```

```
     i:=0;

     FOR i:=1 TO LENGTH(cadena) DO
            cadena[i]:=UPCASE(cadena[i]);

     WRITE (cadena);
     i:= i + 1;
END.



PROGRAM EJER69;
        USES CRT;
        VAR i:INTEGER;
        VAR cad1:STRING;
BEGIN
     i:=0;
     cad1:='Lenguaje Pascal';

     ClrScr;

     WRITELN(cad1);

     WHILE i <= LENGTH(cad1) DO
     BEGIN
          cad1[i]:=UPCASE(cad1[i]);
          i:=i+1;
     END;

     WRITELN(cad1);
END.
```

Transformar la cadena VACA en la cadena vaca.

```
PROGRAM EJER70;
        USES CRT;

        VAR cadena:STRING;
        VAR i, v_ascii:INTEGER;
BEGIN
     ClrScr;

     cadena:='VACA';

     FOR i:=1 TO LENGTH (cadena) DO
     BEGIN
          v_ascii:=ORD (cadena[i]);
          cadena[i]:=(CHR(v_ascii+32));
     END;

     WRITE (cadena);
END.


PROGRAM EJER70;
        USES CRT;
        VAR i,val_asc:INTEGER;
        VAR cad1:STRING;
BEGIN
     i:=0;
     cad1:='VACA';

     ClrScr;

     WRITELN(cad1);

     WHILE i <= LENGTH(cad1) DO
     BEGIN
          val_asc:=ORD(cad1[i]);
          cad1[i]:=(CHR(val_asc+32));
          i:=i+1;
     END;
```

```
     WRITELN(cad1);
END.
```

Escribir un programa en Pascal que lea 4 datos, calcule y visualice en pantalla su producto, suma y media aritmética.

```
PROGRAM EJER71;
        USES CRT;

        VAR a,b,c,d:REAL;
        VAR producto,suma,media:REAL;
BEGIN
     ClrScr;

     WRITELN('Introduzca 4 datos para calcular las soluciones:');
     WRITELN ('');
     WRITE ('a: ');     READLN (a);
     WRITE ('b: ');     READLN (b);
     WRITE ('c: ');     READLN (c);
     WRITE ('d: ');     READLN (d);
     WRITELN ('');

     producto:= a * b * c * d;
     suma:= a + b + c + d;
     media:= suma / 2;

     WRITELN ('El valor del producto de los numeros es: ',producto:10:2);
     WRITELN ('El valor de la suma de los numeros es:   ',suma:10:2);
     WRITELN ('El valor de la media de los numero ses:  ',media:10:2);
END.
```

```
PROGRAM EJER71;
        USES CRT;
        VAR i:INTEGER;
        VAR dato,producto,suma,media:REAL;
BEGIN
     ClrScr;

     producto:=1;
     suma:=0;

     FOR i:=1 TO 4 DO
     BEGIN
          WRITE('Dato ');  WRITE(i);  WRITE('-> ');
          READLN(dato);
          producto:=producto*dato;
          suma:=suma+dato;
     END;

     media:=suma/i;

     WRITE('Producto:   '); WRITELN(producto:5:2);
     WRITE('Suma:       '); WRITELN(suma:5:2);
     WRITE('Media:      '); WRITELN(media:5:2);
END.
```

Escribir un programa en Pascal que lea un peso en libras y a continuación visualice su equivalente en kilos y en gramos. 1 libra = 0.45 Kg

```
PROGRAM EJER72;
        USES CRT;
        VAR libras,kilos,gramos:REAL;
BEGIN
     ClrScr;

     WRITE ('Introduzca el numero de libras para pasarlo a kilos y gramos: ');
     READLN (libras);
     WRITELN ('');

     kilos:= libras * 0.45;
```

```
     gramos:= kilos * 1000;

     WRITELN ('Son ',kilos:5:2,' kilos y ',gramos:5:2,' gramos.');
END.

PROGRAM EJER72;
        USES CRT;
        VAR libras:REAL;
BEGIN
     ClrScr;

     WRITE('Libras     ->   '); READLN(libras);
     WRITE('Kilogramos ->   '); WRITELN(libras*0.45:7:2);
     WRITE('Gramos     ->   '); WRITELN((libras*0.45)/1000:7:4);
END.
```

Escribir un programa en Pascal que calcule y escriba el cuadrado de 821.

```
PROGRAM EJER73;
        USES CRT;

        VAR numero, cuadrado:REAL;
BEGIN
     ClrScr;

     WRITE ('Escriba un numero para hacer su cuadrado: ');
     READLN (numero);
     WRITELN ('');

     cuadrado:= sqr(numero);

     WRITELN ('El cuadrado es: ',cuadrado:5:2)
END.


PROGRAM EJER73;
        USES CRT;
        VAR numero:LONGINT;
BEGIN
     ClrScr;

     numero:=821;

     WRITE('Cuadrado de 821: ');        WRITE(sqr(numero));
END.
```

Escribir un programa en Pascal que escriba los números comprendidos entre 1 y 100. El programa escribirá en la pantalla los números en grupos de 20, solicitando al usuario si quiere o no continuar visualizando el siguiente grupo de números.

```
PROGRAM EJER74;
        USES CRT;

        VAR num, flag, x:INTEGER;
        VAR resp:CHAR;
BEGIN
     ClrScr;
     num:=0;
     x:=1;

     FOR num:=1 TO 100 DO
     BEGIN
          IF (num mod 20)= 0 THEN
          flag := x;
          WRITELN (num);
            IF flag = x THEN
            BEGIN
                 WRITE('DESEA CONTINUAR: S/N --> '); READLN(resp);
                 IF UPCASE (resp)<>'S' THEN
                 BEGIN
```

```
                WRITE ('Este programa ha finalizado'); EXIT
                END;
           END;
         x:= x + 20;
     END;
END.

PROGRAM EJER74;
        USES CRT;
        VAR fila,i:INTEGER;
        VAR resp:CHAR;
BEGIN
     fila:=1;
     ClrScr;

     FOR i:=1 TO 100 DO
     BEGIN
          WRITELN(i);
          fila:=fila+1;

          IF fila = 21 THEN
          BEGIN
               WRITE('Desea continuar (S/N)?');
               READLN(resp);
               IF UPCASE(resp)='S' THEN
               BEGIN
                    ClrScr;
                    fila:=0;
                    CONTINUE;
               END
               ELSE
                   EXIT;
          END;
     END;
END.
```

Escribir un programa en Pascal que calcule, independientemente, la suma y la media de los números pares e impares comprendidos entre 1 y 200.

```
PROGRAM EJER75;
        USES CRT;

        VAR media_p, media_i:REAL;
        VAR suma_p, suma_i:LONGINT;
        VAR i, total_p, total_i:LONGINT;
BEGIN
     ClrScr;

     FOR i:=1 TO 200 DO
     BEGIN
          IF (i mod 2)=0 THEN
                BEGIN
                suma_p:=suma_p + i;
                total_p:=total_p + 1;
                END
          ELSE
                BEGIN
                suma_i:=suma_i + i;
                total_i:=total_i + 1;
                END;
     END;
         media_i:= suma_i / total_i;
         media_p:= suma_p / total_p;
              WRITELN ('La suma de los impares es:  ',suma_i);
              WRITELN ('La suma de los pares es  :  ',suma_p);
              WRITELN ('La media de los impares es: ',media_i:5:2);
              WRITELN ('La media de los pares es:   ',media_p:5:2);
END.

PROGRAM EJER75;
        USES CRT;
        VAR i,conpar,conimp,sumapar,sumaimp:INTEGER;
BEGIN
     ClrScr;
```

```
     sumapar:=0;        sumaimp:=0;     conpar:=0;      conimp:=0;

     FOR i:=1 TO 200 DO
     BEGIN
          IF (i mod 2) = 0 THEN
          BEGIN
               sumapar:=sumapar+i;
               conpar:=conpar+1;
          END
          ELSE
          BEGIN
               sumaimp:=sumaimp+i;
               conimp:=conimp+1;
          END;
     END;

     WRITE('Suma pares:     ');         WRITELN(sumapar:7);
     WRITE('Media pares:    ');         WRITELN(sumapar div conpar:7);

     WRITE('Suma impares:   ');         WRITELN(sumaimp:7);
     WRITE('Media impares:  ');         WRITELN(sumaimp div conimp:7);
END.
```

Escribir un programa en Pascal que calcule el importe de una factura sabiendo que el IVA a aplicar es del 12% y que si el importe bruto de la factura es superior a 50.000 pts se debe realizar un descuento del 5%.

```
PROGRAM EJER76;
        USES CRT;

        VAR i_bruto, i_neto, importe:REAL;
        CONST IVA=0.12;
BEGIN
     ClrScr;

     WRITE ('Escriba el importe bruto: ');   READLN (i_bruto);
     WRITELN ('');

     i_bruto:= i_bruto + (IVA * i_bruto);

     IF i_bruto > 50000 THEN

     BEGIN
     i_neto:= i_bruto - (i_bruto * 0.05);

     WRITE ('El importe neto con descuento del 5% es de: ',i_neto:5:2,' pts')
     END

     ELSE

     BEGIN
     i_neto:= i_bruto;
     WRITE ('El importe bruto sin descuento es de: ',i_neto:5:2,' pts.');
     END;
END.


PROGRAM EJER76;
        USES CRT;
        VAR imp_bruto:REAL;
BEGIN
     ClrScr;

     WRITE('Importe bruto ->  '); READLN(imp_bruto);

     IF imp_bruto <= 50000 THEN
     BEGIN
          WRITE('Importe neto:     ');
          WRITELN(imp_bruto+(imp_bruto*0.12):9:2);
     END
     ELSE
     BEGIN
          WRITE('Importe neto ->  ');
          imp_bruto:=imp_bruto-(imp_bruto*0.05);
```

```
          WRITELN(imp_bruto+(imp_bruto*0.12):9:2);
     END;
END.
```

Escribir un programa en Pascal que calcule la suma de los cuadrados de los 100 primeros números enteros.

```
PROGRAM EJER77;
        USES CRT;

        VAR x:INTEGER;
        VAR suma:REAL;
BEGIN
     ClrScr;

     FOR x:=1 TO 100 DO

     BEGIN
     suma:= suma + (sqr(x));
     END;

     WRITE (suma);
END.
```

```
PROGRAM EJER77;
        USES CRT;
        VAR i:INTEGER;
        VAR sumacuad:LONGINT;
BEGIN
     ClrScr;

     FOR i:=1 TO 100 DO
         sumacuad:=sumacuad+sqr(i);

     WRITE('Suma de Cuadrados (1-100) ---> ');  WRITE(sumacuad);
END.
```

Escribir un programa en Pascal que visualice una tabla de conversión de kilómetros a millas marinas y millas terrestres:

1 milla marina = 1852 metros
1 milla terrestre = 1609 metros

```
PROGRAM EJER78;
        USES CRT;

        VAR m, m_mar, m_terr:REAL; {metros, millas marinas y terrestres}
BEGIN
     ClrScr;

     WRITE ('Introduzca el numero de metros: ');        READLN (m);
     WRITELN ('');

     m_mar:=m/1852;
     m_terr:=m/1609;

     WRITE (m:5:2,' metros son: ',m_mar:5:2,' millas marinas y ');
     WRITE (m_terr:5:2,' millas terrestres');
END.
```

```
PROGRAM EJER78;
        USES CRT;
        VAR milla_marina:REAL;
        VAR milla_terrestre:REAL;
        VAR kilometros:INTEGER;
BEGIN
```

```
     ClrScr;

     WRITELN('Kilometros' + ' ---- ' + 'Millas M.' + ' ---- ' + 'Millas T.');

     FOR kilometros:=1 TO 10 DO
     BEGIN
          WRITE(kilometros:6);
          WRITE(kilometros/1.852:15:2);
          WRITELN(kilometros/1.609:15:2);
     END;
END.
```

Escribir un programa en Pascal que lea 10 datos desde el teclado y sume sólo aquellos que sean negativos.

```
PROGRAM EJER79;
        USES CRT;

        VAR y, suma:REAL;
        VAR x:INTEGER;
BEGIN
     ClrScr;

     WRITELN ('Debe introducir 10 datos, se sumaran solo los negativos:');
     WRITELN ('');

     FOR x:=1 TO 10 DO

     BEGIN
          WRITE('Introduzca el dato ',x:2,': ');  READLN (y);
          IF y < 0 THEN suma := suma + y;
     END;

     WRITELN ('');
     WRITE ('El resultado de la suma de los numeros negativos es: ');
     WRITELN (suma:5:2);
END.
```

```
PROGRAM EJER79;
        USES CRT;
        VAR i:INTEGER;
        VAR num,sumaneg:REAL;
BEGIN
     ClrScr;

     sumaneg:=0;

     FOR i:=1 TO 10 DO
     BEGIN
          WRITE('Numero '); WRITE(i); WRITE(' -> '); READLN(num);
          IF num < 0 THEN
             sumaneg:=sumaneg+num;
     END;

     WRITE('Suma de negativos:       ');        WRITE(sumaneg);
END.
```

Escribir un programa en Pascal que calcule el sueldo semanal de un trabajador a partir del número de horas trabajadas por día y las siguientes tarifas:

600 pts/hora Turno de mañana
800 pts/hora Turno de noche
1000 pts/hora Turno de días festivos

```
PROGRAM EJER80;
        USES CRT;
```

```
        VAR t_mannana, t_noche, t_festivos:INTEGER;
        VAR total:LONGINT;
BEGIN
     ClrScr;

     WRITE ('Introduzca las horas trabajadas por dia, en cada turno, ');
     WRITELN (' para calcular el sueldo semanal.');   WRITELN ('');

     WRITE ('Horas del turno de mañana:        ');    READLN (t_mannana);
     WRITE ('Horas del turno de noche:         ');    READLN (t_noche);
     WRITE ('Horas del turno de dias festivos: ');    READLN (t_festivos);
     WRITELN ('');

     total:=(600 * t_mannana) + (800 * t_noche) + (1000 * t_festivos);

     WRITE ('El sueldo semanal es de: ',total);
END.
```

```
PROGRAM EJER80;
        USES CRT;
        VAR turno,resp:CHAR;
        VAR numhoras:REAL;
BEGIN
     ClrScr;
     resp:='S';

     WRITE('Turno ? (M/N/F)       ---> ');     READLN(turno);
     WRITE('Numero de horas (dia) ---> ');     READLN(numhoras);

     WHILE UPCASE(resp) = 'S' DO
     BEGIN
          IF UPCASE(turno) = 'M' THEN
          BEGIN
               WRITE('Salario neto: '); WRITELN(numhoras*5*600:7:2);
          END
          ELSE IF UPCASE(turno) = 'N' THEN
          BEGIN
               WRITE('Salario neto: '); WRITELN(numhoras*5*800:7:2);
          END
          ELSE IF UPCASE (turno) = 'F' THEN
          BEGIN
               WRITE('Salario neto: '); WRITELN(numhoras*5*1000:7:2);
          END
          ELSE
              WRITELN('Turno incorrecto');
          WRITE('¿Desea continuar (S/N)? ---> ');
          READLN(resp);

          IF UPCASE(resp) = 'S' THEN
          BEGIN
               ClrScr;
               WRITE('Turno ? (M/N/F)       --->');     READLN(turno);
               WRITE('Numero de horas (dia) --->');     READLN(numhoras);
          END
          ELSE
              EXIT;
     END;
END.
```

Escribir un programa en Pascal que rellene un array con los números enteros comprendidos entre 4 y 14.

```
PROGRAM EJER81;
        USES CRT;
        VAR arr_num:ARRAY [4..14] of INTEGER;
        VAR i:INTEGER;
BEGIN
     ClrScr;
```

```
     FOR i:=4 TO 14 DO
         arr_num[i]:=i;

     FOR i:=4 TO 14 DO
     BEGIN
          WRITELN('Numero: ',arr_num[i]);
     END;
END.


PROGRAM EJER81;
        USES CRT;
        VAR arr_num:ARRAY[1..11] of INTEGER;
        VAR i,num:INTEGER;
BEGIN
     ClrScr;

     num:=4;

     FOR i:=1 TO 11 DO
     BEGIN
          arr_num[i]:=num;
          num:=num+1;
     END;

     FOR i:=1 TO 11 DO
     BEGIN
          WRITE('Posición ',i:2, ': '); WRITELN(arr_num[i]:5);
     END;
END.
```

Escribir un programa en Pascal que rellene un array con los números pares comprendidos entre 1 y 10.

```
PROGRAM EJER82;
        USES CRT;
        VAR arr_num:ARRAY [1..10] of INTEGER;
        VAR i, b:INTEGER;
BEGIN
     ClrScr;
     i:=1;

     WHILE i <= 10 DO
     BEGIN
          arr_num[i]:=i;

          IF (i mod 2)=0 THEN
          BEGIN
             WRITELN(arr_num[i]);
          END;

          i:= i + 1;
     END;
END.


PROGRAM EJER82;
        USES CRT;
        VAR arr_num:ARRAY[1..10] of INTEGER;
        VAR i,num:INTEGER;
BEGIN
     ClrScr;
     i:=1;
     num:=1;

     WHILE num<=10 DO
     BEGIN
          IF num mod 2 = 0 THEN
          BEGIN
               arr_num[i]:=num;
               i:=i+1;
          END;
```

```
          num:=num+1;
     END;

     arr_num[i]:=3;
     i:=1;

     WHILE arr_num[i] <> 3 DO
     BEGIN
          WRITE('Posicion ',i:2,' : '); WRITELN(arr_num[i]:5);
          i:=i+1;
     END;
END.
```

Escribir un programa en Pascal que rellene un array con los números comprendidos entre 25 y 35 divididos por 3.

```
PROGRAM EJER83;
        USES CRT;
        VAR arr_num:ARRAY [1..11] of REAL;
        VAR i:INTEGER;
BEGIN
     ClrScr;

     FOR i:=25 TO 35 DO
         arr_num[i]:=i;

     FOR i:=25 TO 35 DO
         WRITELN(arr_num[i] / 3:5:2);
END.


PROGRAM EJER83;
        USES CRT;
        VAR arr_num:ARRAY[1..11] of REAL;
        VAR i,num:INTEGER;
BEGIN
     ClrScr;

     i:=1;
     num:=025;

     FOR i:=1 TO 10 DO
     BEGIN
          arr_num[i]:=num/3;
          num:=num+1;
     END;

     i:=1;

     WHILE i <= 10 DO
     BEGIN
          WRITE('Posicion ',i:2, ': '); WRITELN(arr_num[i]:5);
          i:=i+1;
     END;
END.
```

Escribir un programa en Pascal que rellene un array con cinco números enteros consecutivos y haga una copia de ese array en otro.

```
PROGRAM EJER84;
        USES CRT;
        VAR arr_num1,arr_num2:ARRAY [5..10] of INTEGER;
        VAR i:INTEGER;
BEGIN
     ClrScr;

     FOR i:=5 TO 10 DO
     BEGIN
         arr_num1[i]:=i;
         arr_num2[i]:=arr_num1[i];
     END;
```

```
     FOR i:=5 TO 10 DO
         WRITELN (arr_num2[i]);

END.
```

```
PROGRAM EJER84;
        USES CRT;
        VAR arr_num1,arr_num2:ARRAY[1..5] of INTEGER;
        VAR i,num:INTEGER;
BEGIN
     ClrScr;

     i:=1;
     num:=100;

     FOR i:=1 TO 5 DO
     BEGIN
          arr_num1[i]:=num;
          num:=num+1;
     END;

     FOR i:=1 TO 5 DO
          arr_num2[i]:=arr_num1[i];

     i:=1;

     WRITELN('ARRAY 1   ARRAY 2':30);

     WHILE i <= 5 DO
     BEGIN
          WRITE('Posicion ',i:2, ': ');
          WRITE(arr_num1[i]:5);
          WRITELN(arr_num2[i]:10);
          i:=i+1;
     END;
END.
```

Escribir un programa en Pascal que rellene un array de 10 elementos con los números comprendidos entre 23 y 32 y copie en otro array esos números multiplicados por 0.35.

```
PROGRAM EJER85;
        USES CRT;
        VAR arr_num1, arr_num2:ARRAY [23..32] of REAL;
        VAR i:INTEGER;
BEGIN
     ClrScr;

     FOR i:=23 TO 32 DO
     BEGIN
          arr_num1[i]:=i;
          arr_num2[i]:=(arr_num1[i] * 0.35);
     END;

     FOR i:=23 TO 32 DO
         WRITELN(arr_num2[i]:5:2);
END.
```

```
PROGRAM EJER85;
        USES CRT;
        VAR arr_num1,arr_num2:ARRAY[1..10] of REAL;
        VAR i,num:INTEGER;
BEGIN
     ClrScr;

     i:=1;
     num:=23;

     FOR i:=1 TO 10 DO
     BEGIN
          arr_num1[i]:=num;
          num:=num+1;
```

```
     END;

     FOR i:=1 TO 10 DO
         arr_num2[i]:=arr_num1[i]*0.35;

     i:=1;
     WRITELN('ARRAY 1   ARRAY 2':30);

     WHILE i <= 10 DO
     BEGIN
          WRITE('Posicion ',i:2, ': ');
          WRITE(arr_num1[i]:5:2);
          WRITELN(arr_num2[i]:10:2);
          i:=i+1;
     END;
END.
```

Escribir un programa en Pascal que rellene un array con los veinte primeros números pares y calcule su suma.

```
PROGRAM EJER86;
        USES CRT;
        VAR arr_pares:ARRAY [1..40] of INTEGER;
        VAR i, suma:INTEGER;
BEGIN
     ClrScr;

     i:=1;

     FOR i:= 1 TO 40 DO
     BEGIN
          IF (i mod 2) = 0 THEN
          BEGIN
               arr_pares[i]:=i;
               suma:= suma + i;
          END;
     END;

     WRITELN('La suma de los 20 primeros numeros pares es: ',suma);
END.


PROGRAM EJER86;
        USES CRT;
        VAR arr_num:ARRAY[1..25] of INTEGER;
        VAR i,num,suma_par:INTEGER;
BEGIN
     ClrScr;

     i:=1;
     num:=1;
     suma_par:=0;

     WHILE i<=20 DO
     BEGIN
          IF num mod 2 = 0 THEN
          BEGIN
               arr_num[i]:=num;
               i:=i+1;
               suma_par:=suma_par+num;
          END;
          num:=num+1;
     END;

     i:=1;

     WHILE i <= 20 DO
     BEGIN
          WRITE('Posición ',i:2, ': ');
          WRITELN(arr_num[i]:5);
          i:=i+1;
     END;
     WRITE('SUMA: ', suma_par:12);
END.
```

Escribir un programa en Pascal que solicite cinco números, los almacene en un array y luego calcule la media aritmética de esos números.

```
PROGRAM EJER87;
        USES CRT;
        VAR arr_num:ARRAY [1..5] of REAL;
        VAR i, num:INTEGER;
        VAR media:REAL;
BEGIN
     ClrScr;

     WRITELN ('Escriba 5 numeros para hacer su media aritmetica: ');

     FOR i := 1 TO 5 DO
     BEGIN
         READLN(num);
         arr_num[i]:=num;
     END;

     FOR i:=1 TO 5 DO
     media:= media + arr_num[i];

     media:= media / i;

     WRITELN ('La media aritmetica es: ',media:5:2);
END.
```

```
PROGRAM EJER87;
        USES CRT;
        VAR arr_num:ARRAY[1..10] of REAL;
        VAR num,suma,media:REAL;
        VAR i:INTEGER;
BEGIN
     ClrScr;

     i:=0;
     suma:=0;
     media:=0;

     WHILE i<5 DO
     BEGIN
          WRITE('Numero ',i+1,'--->');  READLN(num);
          arr_num[i]:=num;
          suma:=suma+num;
          i:=i+1;
     END;

     media:=(suma/i);

     WRITE('Media: ', media:5:2);
END.
```

Escribir un programa en Pascal que tras asignar los números, 23, 45, 68, 99, 10, 15 y 4 a un array, determine la posición del array en la que se encuentra el máximo valor.

```
PROGRAM EJER88;
        USES CRT;
        CONST arr_num:ARRAY [1..7] of INTEGER=(23,45,68,99,10,15,4);
        VAR i:INTEGER;
BEGIN
     ClrScr;

     FOR i:=1 TO 7 DO
     BEGIN

     IF arr_num[i]=99 THEN
```

```
        WRITE ('La posicion del mayor numero (',arr_num[i],') es: ',i);
     END;
END.



PROGRAM EJER88;
        USES CRT;
        CONST arr_num:ARRAY[1..7] of INTEGER=(23,45,68,99,10,15,4);
        VAR i,posi_max,val_max:INTEGER;
BEGIN
     ClrScr;

     FOR i:=1 TO 7 DO
     BEGIN
          IF arr_num[i] > val_max THEN
          BEGIN
               val_max:=arr_num[i];
               posi_max:=i;
          END;
     END;

     WRITE('VALOR MAXIMO: ', val_max, ' POSICION: ', posi_max);
END.
```

Escribir un programa en Pascal que tras asignar los números, -2, 5, 8, -9, 10, 15 y –4 a un array calcule, independientemente, la suma de los elementos positivos y negativos.

```
PROGRAM EJER89;
        USES CRT;
        CONST arr_num:ARRAY [1..7] of INTEGER=(-2,5,8,-9,10,15,-4);
        VAR i:INTEGER;
        VAR suma_p, suma_n:INTEGER;
BEGIN
     ClrScr;

     FOR i:=1 TO 7 DO
       BEGIN
          IF arr_num[i] >= 0 THEN
             suma_p:= suma_p + arr_num[i]
          ELSE IF arr_num[i] < 0 THEN
             suma_n:= suma_n + arr_num[i];
       END;

     WRITELN ('La suma de los numeros positivos es:    ',suma_p);
     WRITELN ('La suma de los numeros negativos es:    ',suma_n);
END.

PROGRAM EJER89;
        USES CRT;
        CONST arr_num:ARRAY[1..7] of INTEGER=(-2,5,8,-9,10,15,-4);
        VAR i,suma_pos,suma_neg:INTEGER;
BEGIN
     ClrScr;

     FOR i:=1 TO 7 DO
       BEGIN
          IF arr_num[i] > 0 THEN
              suma_pos:=suma_pos+arr_num[i]
          ELSE
              suma_neg:=suma_neg+arr_num[i];
       END;

     WRITELN('SUMA POSITIVOS: ', suma_pos);
     WRITELN('SUMA NEGATIVOS: ', suma_neg);
END.
```

Escribir un programa en Pascal que tras asignar los números, 23, 45, 68, 99, 10, 15 y 4 a un array, determine las posiciones del array en las que se encuentran el máximo y el mínimo valor.

```
PROGRAM EJER90;
        USES CRT;
        CONST arr: ARRAY [1..7] of INTEGER=(23,45,68,99,10,15,4);
        VAR i, mayor, menor,posi_mayor,posi_menor:INTEGER;
BEGIN
     ClrScr;

     mayor:= arr[1];
     menor:= arr[1];

     FOR i:=2 TO 7 DO
     BEGIN
          IF arr[i] >= mayor THEN
          BEGIN
              mayor:= arr[i];
              posi_mayor:=i;
          END
          ELSE
              CONTINUE;
     END;

     WRITELN ('El numero mayor es: ',mayor:3);
     WRITELN ('Su posicion es:     ',posi_mayor:3);
     WRITELN ('');

     FOR i:=2 TO 7 DO
     BEGIN
          IF arr[i] <= menor THEN
          BEGIN
              menor:= arr[i];
              posi_menor:=i;
          END
          ELSE
              CONTINUE;
     END;

     WRITELN ('El numero menor es: ',menor:3);
     WRITELN ('Su posicion es:     ',posi_menor:3);
END.


PROGRAM EJER90;
        USES CRT;
        CONST arr_num:ARRAY[1..7] of INTEGER=(23,45,68,99,10,15,4);
        VAR i,val_max,val_min,pos_max,pos_min:INTEGER;
BEGIN
     ClrScr;

     val_min:=arr_num[1];
     val_max:=arr_num[1];

     FOR i:=1 TO 7 DO
     BEGIN
          IF arr_num[i] > val_max THEN
          BEGIN
               val_max:=arr_num[i];
               pos_max:=i;
          END;

          IF arr_num[i] < val_min THEN
          BEGIN
               val_min:=arr_num[i];
               pos_min:=i;
          END;
     END;

     WRITELN('VALOR MAXIMO: ', val_max:3, ' POSICIàN: ', pos_max:3);
     WRITELN('VALOR MINIMO: ', val_min:3, ' POSICIàN: ', pos_min:3);
END.
```

Escribir un programa en Pascal que determine la posición de la siguiente matriz en la que se encuentra el valor máximo.

23 45 68
34 99 12
25 78 89

```
PROGRAM EJER91B;
        USES CRT;
        CONST arr_num:ARRAY[1..3,1..3] of INTEGER=( (23,45,68),
                                                    (34,99,12),
                                                    (25,78,89) );
        VAR i,j,val_max,pos_max_i,pos_max_j:INTEGER;
BEGIN
     ClrScr;

     val_max:=arr_num[1,1];

     FOR i:=1 TO 3 DO
     BEGIN
          FOR j:=1 TO 3 DO
          BEGIN
               IF arr_num[i,j] > val_max THEN
               BEGIN
                    val_max:=arr_num[i,j];
                    pos_max_i:=i;
                    pos_max_j:=j;
               END;
          END;
     END;

     WRITELN(    'VALOR MAXIMO: ', val_max:3,
                 ' POSICION: ',    pos_max_i:3,pos_max_i:3);
END.
```

Escribir un programa en Pascal que sume, independientemente, los elementos positivos y negativos de la siguiente matriz:

-12 23 32
45 -56 -10
25 78 89

```
PROGRAM EJER92;
        USES CRT;
        CONST arr_num:ARRAY[1..3,1..3] of INTEGER=( (-12,23,-32),
                                                    (45,-56,-10),
                                                    (25,78,89)   );
        VAR i,j,suma_pos,suma_neg:INTEGER;
BEGIN
     suma_pos:=0;
     suma_neg:=0;
     ClrScr;

     FOR i:=1 TO 3 DO
     BEGIN
          FOR j:=1 TO 3 DO
          BEGIN
               IF arr_num[i,j] < 0 THEN
                   suma_neg:=suma_neg+arr_num[i,j]
               ELSE
                   suma_pos:=suma_pos+arr_num[i,j]
          END;
     END;

     WRITELN('SUMA POSITIVOS: ', suma_pos:5);
     WRITELN('SUMA NEGATIVOS: ', suma_neg:5);
END.
```

Escribir un programa en Pascal que multiplique por dos los elementos de la siguiente matriz:

4	7	8
6	9	1
5	0	3

```
PROGRAM EJER93;
        USES CRT;
        CONST arr_num:ARRAY[1..3,1..3] of INTEGER=( (4,7,8),
                                                    (6,9,1),
                                                    (5,0,3) );
        VAR i,j:INTEGER;
BEGIN
     ClrScr;

     FOR i:=1 TO 3 DO
         FOR j:=1 TO 3 DO
             arr_num[i,j]:=arr_num[i,j]*2;

     FOR i:=1 TO 3 DO
     BEGIN
          FOR j:=1 TO 3 DO
              WRITE(arr_num[i,j]:3);
          WRITELN (' ');
     END;
END.
```

Escribir un programa en Pascal que almacene en la segunda fila de la siguiente matriz los cuadrados de los datos de la primera fila:

3	6	7	8	9
0	0	0	0	0

```
PROGRAM EJER94;
        USES CRT;
        CONST arr_num:ARRAY [1..2,1..5] of INTEGER=( (3,6,7,8,9),
                                                     (0,0,0,0,0) );
        VAR i,j,cuad:INTEGER;
BEGIN
     ClrScr;

     i:=1;

     FOR j:=1 TO 5 DO
     BEGIN
          FOR i:=1 TO 1 DO
          BEGIN
              cuad:=sqr(arr_num[i,j]);
              arr_num[2,j]:= cuad;
              WRITELN (arr_num[2,j]);
          END;
     END;
END.
```

```
PROGRAM EJER94;
        USES CRT;
        CONST arr_num:ARRAY[1..2,1..5] of INTEGER=( (3,6,7,8,9),
                                                    (0,0,0,0,0) );
        VAR i,j:INTEGER;
BEGIN
     ClrScr;

     FOR i:=1 TO 1 DO
         FOR j:=1 TO 5 DO
             arr_num[i+1,j]:=sqr(arr_num[i,j]);
```

```
     FOR i:=1 TO 2 DO
     BEGIN
          FOR j:=1 TO 5 DO
              WRITE(arr_num[i,j]:3);
          WRITELN(' ');
     END;
END.
```

Escribir un programa en Pascal que sume los datos de cada una de las filas de la siguiente matriz; el resultado se almacenará en la última posición de cada fila:

3	6	7	8	9	0
1	4	3	2	7	0

```
PROGRAM EJER95;
        USES CRT;
        CONST arr_num:ARRAY [1..2,1..6] of INTEGER=( (3,6,9,7,8,0),
                                                     (1,4,3,2,7,0) );
        VAR i,j,suma1, suma2:INTEGER;
BEGIN
     ClrScr;

     FOR i:=1 TO 2 DO
     BEGIN
          FOR j:=1 TO 6 DO
          BEGIN
               IF i=1 THEN suma1:= suma1 + arr_num[1,j];
               IF i=2 THEN suma2:= suma2 + arr_num[2,j];
          END;
     END;

          WRITELN ('La suma de la fila 1 es: ',suma1);
          WRITELN ('La suma de la fila 2 es: ',suma2);
END.


PROGRAM EJER95;
        USES CRT;
        CONST arr_num:ARRAY[1..2,1..6] of INTEGER=( (3,6,9,7,8,0),
                                                    (1,4,3,2,7,0) );
        VAR suma_fila,i,j:INTEGER;
BEGIN
     ClrScr;

     FOR i:=1 TO 2 DO
     BEGIN
          suma_fila:=0;
          FOR j:=1 TO 6 DO
              suma_fila:=suma_fila+arr_num[i,j];
          arr_num[i,j]:=suma_fila;
     END;

     FOR i:=1 TO 2 DO
     BEGIN
          FOR j:=1 TO 6 DO
              WRITE(arr_num[i,j]:3);
          WRITELN(' ');
     END;
END.
```

Escribir un programa en Pascal que sume los datos de cada una de las columnas de la siguiente matriz; el resultado se almacenará en la última posición de cada columna:

3	2
4	6
8	9
0	0

```
PROGRAM EJER96;
        USES CRT;
        CONST arr_num:ARRAY [1..4,1..2] of INTEGER=( (3,2),(4,6),
                                                     (8,9),(0,0) );
        VAR i,j,suma1,suma2:INTEGER;
BEGIN
     ClrScr;

     FOR j:=1 TO 2 DO
     BEGIN
          FOR i:=1 TO 4 DO
          BEGIN
               IF j=1 THEN suma1:= suma1 + arr_num[i,j];
               IF j=2 THEN suma2:= suma2 + arr_num[i,j];
          END;
     END;

     WRITELN ('El resultado de la suma 1 es: ',suma1);
     WRITELN ('El resultado de la suma 2 es: ',suma2);
END.
```

Escribir un programa en Pascal que sume los elementos de cada una de las filas y de las columnas de la siguiente matriz; el resultado de cada suma se almacenará en la última posición de la fila o columna correspondiente. Además la suma total de todos los elementos de la matriz se almacenará en el elemento de la esquina inferior derecha de la matriz:

1	7	0
5	6	0
6	4	0
7	3	0
0	0	0

```
PROGRAM EJER97;
        USES CRT;
        CONST arr_num:ARRAY [1..5,1..3] of INTEGER=( (1,7,0),(5,6,0),
                                                     (6,4,0),(7,3,0),
                                                     (0,0,0) );
        VAR i,j,total:INTEGER;
        VAR suma_h,suma_v:INTEGER; {Es la suma horizontal y vertical}

BEGIN
     ClrScr;

     total := 0;

     FOR i:=1 TO 5 DO
     BEGIN
          suma_h:=0;
          FOR j:=1 TO 3 DO
            suma_h:= suma_h + arr_num[i,j];


          WRITELN ('La suma de la fila ',i,' es: ',suma_h:3);
          total:=total + suma_h;
     END; WRITELN ('');

     FOR j:=1 TO 2 DO
     BEGIN
          suma_v:=0;
          FOR i:=1 TO 5 DO
            suma_v:= suma_v + arr_num[i,j];

          WRITELN ('La suma de la columna ',j,' es: ',suma_v:3);
          total:=total + suma_v;
```

```
     END; WRITELN ('');

     WRITELN ('La suma total es: ',total);
END.


PROGRAM EJER97;
        USES CRT;
        CONST arr_num:ARRAY[1..5,1..3] of INTEGER=( (1,7,0),
                                                    (5,6,0),
                                                    (6,4,0),
                                                    (7,3,0),
                                                    (0,0,0) );
        VAR suma_fila,suma_colu,suma_tota,i,j:INTEGER;
BEGIN

     ClrScr;

     FOR i:=1 TO 4 DO
     BEGIN
          suma_fila:=0;
          FOR j:=1 TO 2 DO
          BEGIN
               suma_fila:=suma_fila+arr_num[i,j];
               suma_tota:=suma_tota+arr_num[i,j];
          END;
          arr_num[i,j+1]:=suma_fila;
     END;
          arr_num[i+1,j+1]:=suma_tota;

     FOR j:=1 TO 2 DO
     BEGIN
          suma_colu:=0;
          FOR i:=1 TO 4 DO
          BEGIN
               suma_colu:=suma_colu+arr_num[i,j];
          END;
          arr_num[i+1,j]:=suma_colu;
     END;

     FOR i:=1 TO 5 DO
     BEGIN
          FOR j:=1 TO 3 DO
              WRITE(arr_num[i,j]:3);
          WRITELN(' ');
     END;
END.
```

Escribir un programa en Pascal que divida todos los elementos de una matriz M (3,4) por el elemento situado en la posición 2,2.

```
PROGRAM EJER98;
        USES CRT;
        CONST arr_num:ARRAY[1..3,1..4] of INTEGER=( (23,45,-68,99),
                                                    (45,65,-76,34),
                                                    (56,-75,34,98) );
        VAR i,j:INTEGER;
        VAR divi:REAL;
BEGIN
     ClrScr;

     FOR i:=1 TO 3 DO
     BEGIN
          FOR j:=1 TO 4 DO
          BEGIN
               divi:= arr_num[i,j] / arr_num[2,2];
               WRITE   ('Dividido ',arr_num[i,j]:3,' por el numero ');
               WRITELN (arr_num[2,2]:3,': ',divi:5:2);
          END;
     END;
END.
```

```
PROGRAM EJER98;
        USES CRT;
        CONST matriz_m:ARRAY[1..3,1..4] OF REAL = ((2,3,4,12),
                                                   (7,9,8,11),
                                                   (5,6,1,19));
        VAR i,j:INTEGER;
        VAR ele_22:REAL;
BEGIN
     ele_22:=matriz_m[2,2];

     ClrScr;

     FOR i:=1 TO 3 DO
     BEGIN
          FOR j:=1 TO 4 DO
              WRITE(matriz_m[i,j]:5:2,' ');
          WRITELN(' ');
     END;

     FOR i:=1 TO 3 DO
         FOR j:=1 TO 4 DO
             matriz_m[i,j]:=matriz_m[i,j]/ele_22;

     WRITELN(' ');

     FOR i:=1 TO 3 DO
     BEGIN
          FOR j:=1 TO 4 DO
              WRITE(matriz_m[i,j]:5:2,' ');
          WRITELN(' ');
     END;
END.
```

Escribir un programa en Pascal que almacene en un array los números primos comprendidos entre 1 y 100.

```
PROGRAM EJER99;
        USES CRT;
        VAR arr_num:ARRAY [1..100] of INTEGER;
        VAR flag:INTEGER;
        VAR i,num:INTEGER;
BEGIN
     ClrScr;

     num:=1;

     WHILE num < 100 DO
     BEGIN
          FOR i:=2 TO (num-1) DO
          BEGIN
               IF (num mod i)=0 THEN
                  flag:=1;
          END;

          IF flag<>1 THEN
             WRITELN (num:3,' es un n mero primo.');

          num:= num + 1;
          flag:=0;
     END;
END.
```

```
PROGRAM EJER99;
        USES CRT;
        VAR arra_prim:ARRAY[1..100] OF INTEGER;
        VAR i,divisor:INTEGER;
        VAR flag,num:INTEGER;
BEGIN
     i:=1;

     FOR num:=2 TO 100 DO
```

```
     BEGIN
          flag:=1;

          FOR divisor:=2 TO num-1 DO
          BEGIN
               IF num MOD divisor = 0 THEN
                  flag:=0;
          END;

          IF flag=1 THEN
          BEGIN
               arra_prim[i]:=num;
               i:=i+1;
          END;
     END;
     arra_prim[i]:=0;

     i:=1;
     WHILE(arra_prim[i]<>0) DO
     BEGIN
          WRITE(arra_prim[i],' ');
          i:=i+1;
     END;
END.
```

Escribir un programa en Pascal que genera la matriz transpuesta de una matriz de 3 filas y 4 columnas. La matriz transpuesta de una matriz M(m,n) se obtiene intercambiando filas por columnas y viceversa; el resultado se tiene que almacenar en una nueva matriz M_TRANS(n,m).

```
PROGRAM EJERC100;
        USES CRT;
        CONST matriz:ARRAY [1..3,1..4] of INTEGER=( (12,67,-23,-45),
                                                    (45,-34,23,-12),
                                                    (-34,22,88,-10) );
        VAR m_tra:ARRAY [1..4,1..3] of INTEGER;
        VAR f,c:INTEGER;
BEGIN
     ClrScr;

     FOR c:=1 TO 3 DO
     BEGIN
          FOR f:=1 TO 4 DO
          BEGIN
               m_tra[f,c]:=matriz[c,f];
               WRITE ('(',f,',',c,') ');
               WRITELN (m_tra[f,c]:3);
          END;
     END;
END.
```

```
PROGRAM EJERC100;
        USES CRT;
        CONST m_orig:ARRAY[1..3,1..4] OF REAL = ((2,3,4,12),
                                                 (7,9,8,11),
                                                 (5,6,1,19));
        VAR m_tran:ARRAY[1..4,1..3] OF REAL;
        VAR i,j:INTEGER;
BEGIN
     ClrScr;

     FOR i:=1 TO 3 DO
         FOR j:=1 TO 4 DO
             m_tran[j,i]:=m_orig[i,j];

     FOR i:=1 TO 3 DO
     BEGIN
          FOR j:=1 TO 4 DO
              WRITE(m_orig[i,j]:5:2,' ');
          WRITELN(' ');
     END;
```

```
     WRITELN(' ');

     FOR i:=1 TO 4 DO
     BEGIN
          FOR j:=1 TO 3 DO
              WRITE(m_tran[i,j]:5:2,' ');
          WRITELN(' ');
     END;
END.
```

Escribir un programa en Pascal que genera la inversa de una cadena de caracteres. La cadena original y la invertida deben almacenarse en arrays independientes.

```
PROGRAM EJERC101;
        USES CRT;
        VAR original: ARRAY [1..4] of STRING;
        VAR invertida:ARRAY [1..4] of STRING;
        VAR cadena:STRING;
        VAR i:INTEGER;
BEGIN
     ClrScr;
     cadena:='hola';

     FOR i:=1 TO LENGTH(cadena) DO
     BEGIN
           original[i]:= cadena[i];
           WRITE (original[i]);
     END;

     WRITELN ('');

     FOR i:=LENGTH(cadena) DOWNTO 1 DO
     BEGIN
          invertida[i]:=cadena[i];
          WRITE (invertida[i]);
     END;
END.
```

```
PROGRAM EJERC101;
        USES CRT;
        CONST cad_orig:ARRAY[1..9] of CHAR='GUAYABITA';
        VAR cad_copi:STRING;
        VAR i,j:INTEGER;
BEGIN
     ClrScr;
     WRITELN(cad_orig);
     j:=9;

     FOR i:=1 TO 9 DO
     BEGIN
          cad_copi[j]:=cad_orig[i];
          j:=j-1;
     END;

     FOR i:=1 TO 9 DO
     BEGIN
          WRITE(cad_copi[i]);
     END;
END.
```

Escribir un programa en Pascal que sume dos matrices bidimensionales. Las matrices para que puedan sumarse deben tener las mismas dimensiones.

```
PROGRAM EJERC102;
        USES CRT;
        CONST m1:ARRAY [1..2,1..2] of INTEGER=( (3,1),(4,5) );
        CONST m2:ARRAY [1..2,1..2] of INTEGER=( (1,3),(4,2) );
        VAR m3:  ARRAY [1..2,1..2] of INTEGER;
        VAR f, c:INTEGER;
```

```
BEGIN
     ClrScr;
     FOR f:=1 TO 2 DO
          FOR c:=1 TO 2 DO
          BEGIN
              m3[f,c]:=(m1[f,c] + m2[f,c]);
              WRITE ('(',f,',',c,') '); {Muestra la posicion}
              WRITELN (m3[f,c]);
          END;
END.
```

```
PROGRAM EJERC102;
        USES CRT;
        CONST m_1:ARRAY[1..3,1..4] OF REAL= (   (12,13,14,10),
                                                (15,16,17,10),
                                                (18,19,20,10) );
        CONST m_2:ARRAY[1..3,1..4] OF REAL= (   (1,1,1,1),
                                                (1,1,1,1),
                                                (1,1,1,1) );
        VAR m_suma:ARRAY[1..3,1..4] OF REAL;
        VAR i,j:INTEGER;
BEGIN
     ClrScr;

     FOR i:=1 TO 3 DO
         FOR j:=1 TO 4 DO
             m_suma[i,j]:=m_1[i,j]+m_2[i,j];

     FOR i:=1 TO 3 DO
     BEGIN
          FOR j:=1 TO 4 DO
              WRITE(m_1[i,j]:5:2,' ');
          WRITELN(' ');
     END;

     WRITELN(' ');

     FOR i:=1 TO 3 DO
     BEGIN
          FOR j:=1 TO 4 DO
              WRITE(m_2[i,j]:5:2,' ');
          WRITELN(' ');
     END;

     WRITELN(' ');

     FOR i:=1 TO 3 DO
     BEGIN
          FOR j:=1 TO 4 DO
              WRITE(m_suma[i,j]:5:2,' ');
          WRITELN(' ');
     END;
END.
```

Escribir un programa en Pascal que elimine los blancos de una cadena de caracteres. La cadena original y la transformada deben almacenarse en arrays independientes.

```
PROGRAM EJERC103;
        USES CRT;
        VAR cad_tra:ARRAY [1..20] of STRING;
        CONST cad_ori:STRING='la casa es azul';
        VAR i:INTEGER;

BEGIN
     ClrScr;

     FOR i:=1 TO LENGTH(cad_ori) DO
     BEGIN
          IF cad_ori[i]<>' ' THEN
              BEGIN
                   cad_tra[i]:=cad_ori[i];
                   WRITE (cad_tra[i]);
```

```
              END;
     END;
END.


PROGRAM EJERC103;
        USES crt;
        CONST cad_orig:STRING='Archipielago de Cabo Verde';
        VAR cad_tran:STRING;
        VAR i,j,nc:INTEGER;
BEGIN
     ClrScr;

     i:=1;
     nc:=LENGTH(cad_orig);

     j:=1;
     FOR i:=1 TO nc DO
     BEGIN
          IF cad_orig[i] <> ' ' THEN
          BEGIN
               cad_tran[j]:=cad_orig[i];
               j:=j+1;
          END
     END;

     WRITELN(cad_orig);

     FOR i:=1 TO j-1 DO
     BEGIN
          WRITE(cad_tran[i]);
     END;
END.
```

Escribir un programa en Pascal que cuente las mayúsculas de una cadena de caracteres.

```
PROGRAM EJERC104;
        USES CRT;
        CONST cadena:STRING=('EstO es PROGraMAcion');
        VAR i, mayus:INTEGER;

BEGIN
     ClrScr;

     FOR i:=1 TO LENGTH(cadena) DO
     BEGIN
          IF cadena[i] = UPCASE(cadena[i]) THEN
             mayus:=mayus + 1;
          IF cadena[i]=' ' THEN
             mayus:=mayus - 1;

     END;

     WRITELN ('El numero de mayusculas es: ', mayus);
END.


PROGRAM EJERC104;
        USES CRT;
        CONST cad_orig:STRING='Archipielago de Cabo Verde';
        VAR i,nc,n_may:INTEGER;
BEGIN

     ClrScr;

     nc:=LENGTH(cad_orig);
     n_may:=0;

     FOR i:=1 TO nc DO
     BEGIN
          IF (ORD(cad_orig[i]) >= 65) AND (ORD(cad_orig[i]) <= 90) THEN
             n_may:=n_may+1;
     END;
```

```
     WRITELN(cad_orig);
     WRITELN('MAYUSCULAS: ',n_may);
END.
```

Escribir un programa en Pascal que cambie las mayúsculas de una cadena de caracteres a minúsculas y viceversa.

```
PROGRAM EJERC105;
        USES CRT;
        VAR cadena:STRING;
        VAR v_ascii,i:INTEGER;
BEGIN
     ClrScr;

     {Este programa cambia las mayusculas a minusculas y viceversa}

     cadena:='ViCtOr';

     FOR i:=1 TO LENGTH(cadena) DO
     BEGIN
          IF cadena[i] = UPCASE (cadena[i]) THEN
          BEGIN
               v_ascii:=ORD(cadena[i]);
               cadena[i]:=(CHR(v_ascii+32));
          END
          ELSE
          BEGIN
               cadena[i]:=UPCASE (cadena[i]);
          END;

              WRITE (cadena[i]);
     END;
END.
```

```
PROGRAM EJERC105;
        USES CRT;
        CONST cad_orig:STRING='Archipielago de Cabo Verde';
        VAR i,nc:INTEGER;
BEGIN
     ClrScr;

     WRITELN(cad_orig);

     nc:=LENGTH(cad_orig);

     FOR i:=1 TO nc DO
     BEGIN
          IF (ORD(cad_orig[i]) >= 65) AND (ORD(cad_orig[i]) <= 90) THEN
             cad_orig[i]:=CHR(ORD(cad_orig[i]) + 32)
          ELSE IF (ORD(cad_orig[i]) >= 97) AND (ORD(cad_orig[i]) <= 122) THEN
             cad_orig[i]:=CHR(ORD(cad_orig[i])-32);
     END;

     WRITELN(cad_orig);
END.
```

Escribir un programa en Pascal que encripte una cadena de caracteres sumando 2 al código ASCII de cada uno de sus caracteres.

```
PROGRAM EJERC106;
        USES CRT;
        VAR cadena:STRING;
        VAR encrip:INTEGER;
        VAR i:INTEGER;
BEGIN
     ClrScr;

     WRITE ('Introduzca una cadena para encriptarla: ');
     READLN (cadena);
```

```
     WRITELN('');

     FOR i:=1 TO LENGTH(cadena) DO
     BEGIN
          encrip:=ORD(cadena[i]);
          cadena[i]:=(CHR(encrip + 2));
          WRITE(cadena[i]);
     END;
END.


PROGRAM EJERC106;
        USES CRT;
        CONST cad_orig:STRING='Archipielago de Cabo Verde';
        VAR i,nc:INTEGER;
BEGIN
     ClrScr;

     WRITELN(cad_orig);

     nc:=LENGTH(cad_orig);

     FOR i:=1 TO nc DO
         cad_orig[i]:=CHR(ORD(cad_orig[i])+2);

     WRITELN(cad_orig);
END.
```

Escribir un programa en Pascal que encripte los caracteres de una cadena sumando 2 a los que situados en posiciones pares y 3 a los situados en posiciones impares.

```
PROGRAM EJERC107;
        USES CRT;
        VAR cadena:STRING;
        VAR encrip, i:INTEGER;
BEGIN
     ClrScr;

     WRITE ('Introduzca una cadena para encriptarla: ');
     READLN (cadena);
     WRITELN ('');

     FOR i:=1 TO LENGTH (cadena) DO
     BEGIN
          IF (i mod 2)=0 THEN
          BEGIN
          encrip:=ORD(cadena[i]);
          cadena[i]:= (CHR(encrip + 2));
          END
          ELSE
          BEGIN
          encrip:=ORD(cadena[i]);
          cadena[i]:= (CHR(encrip + 3));
          END;

          WRITE(cadena[i]);
     END;
END.
```

Escribir un programa que lea tres números enteros e indique si están o no, en orden numérico ascendente o descendente.

```
PROGRAM EJER001;
Uses Crt;
         var num1,num2,num3:Integer;

Begin
     ClrScr;

     WRITELN('Introduzca tres numeros:');
```

```
     WRITE('Numero 1: '); READLN(num1);
     WRITE('Numero 2: '); READLN(num2);
     WRITE('Numero 3: '); READLN(num3);

     WRITELN;
     WRITELN;

     If (num1 > num2) and (num2 > num3) then
        WRITELN('Ha introducido los numeros en orden decreciente.')
     Else if (num1 < num2) and (num2 < num3) then
        WRITELN('Ha introducido los numeros en orden creciente.')
     Else
        WRITELN('No ha introducido los numeros en orden.');
End.

Program EJ001(Input, Output);
Uses Crt;
Var n1, n2, n3: integer;

     Function ordenados(i, j: Integer): Boolean;
     {Función booleana de dos parámetros enteros, que devuelve TRUE si el
     primer número es menor o igual que el segundo y FALSE si es mayor:}

     begin
     ordenados := (i <= j)
     end;

Begin
ClrScr;
writeln('*** Introduce tres números enteros ***');
write('Primero: ');
readln(n1);
write('Segundo: ');
readln(n2);
write('Tercero: ');
readln(n3);
writeln;
{Mediante la función, se comprueba si el primer número es menor que
el segundo, y si el segundo es además menor que el tercero.}

if ordenados(n1,n2) and ordenados(n2,n3)
     then writeln('Los tres números están en orden.')
     else writeln('Los tres números NO están en orden');
readkey
end.
```

Escribir un programa que lea 5 valores de temperatura, y escriba el número de veces que estuvo bajo 0º.

```
PROGRAM EJER002;
Uses Crt;
         var temp,i,cont:Integer;
Begin
     WRITELN ('Introduzca 5 valores de temperatura:');
     WRITELN;

     cont := 0;

     For i := 0 to 4 do
         Begin
              WRITE('Valor ',i + 1,' : '); Readln(temp);
              If temp < 0 Then
                 inc(cont);
         End;

     WRITELN;
     WRITE('La temperatura ha estado ',cont);

     If cont = 1 then
         WRITE (' vez bajo 0')
     Else
         WRITE (' veces bajo 0');
End.
```

```
program EJ002(Input, Output);
var cont, temp, index: integer;
begin
cont := 0;
{Contador, nº de veces que la temperatura es inferior a 0 grados.}

for index := 1 to 7 do
     begin
     write('Dime temperatura(',index,'): ');
     readln(temp);
     { Leer temperatura desde teclado}

     if temp < 0 then cont := cont + 1
     { Si es menor que cero, incrementar el contador}

     end;
write('La temperatura fue ',cont);
{ Escribir resultado por pantalla.}

if cont = 1 then write(' vez ') else write(' veces ');
writeln('inferior a cero.')

end.
```

Se realiza un examen, se piden las notas del número de alumnos introducidos por el usuario, las cuales pueden ser únicamente enteras entre 0 y 10. Debe decirse cuantos 0, 5 y 10 han aparecido. También deben decirse las notas introducidas que no sean 0, 5 o 10.

```
PROGRAM EJER003;
Uses Crt;
         var cero, cinco, diez: Integer;
         var nota, i, alumnos: Integer;
         var notas_no: Array [1..30] of Integer;

Begin
     ClrScr;

     WRITE('Introduzca el numero de alumnos: '); READLN(alumnos);
     WRITELN;

     WRITELN('Introduzca las calificaciones: (en valores enteros) ');
     WRITELN;
     For i := 1 to alumnos do
         Begin
              WRITE('Alumno: '); READLN(nota);
              If       nota = 0  then inc(cero)
              else if  nota = 5  then inc(cinco)
              else if  nota = 10 then inc(diez)
              else
                   notas_no[i] := nota;
         End;

     WRITELN;
     WRITE('Ha habido ',cero);
               If cero = 1 then  WRITELN(' cero.')  else WRITELN(' ceros.');
     WRITE('Ha habido ',cinco);
               If cinco = 1 then WRITELN(' cinco.') else WRITELN(' cincos.');
     WRITE('Ha habido ',diez);
               If diez = 1 then  WRITELN(' diez.')  else WRITELN(' dieces.');
     WRITELN;
     WRITELN('Las notas aparecidas que no son 0, 5 o 10 son: ');
                 For i := 1 to i DO
                     IF notas_no[i] <> 0 then WRITE(notas_no[i],', ');
END.

program EJ003(Input, Output);
Uses Crt;
var Notas: Set of 0..10;
    index, alumno, veces0, veces5, veces10, N: integer;
```

```
begin
clrscr;
Notas := [];
{Se inicializa a conjunto vacío el conjunto donde se guardar n
las calificaciones obtenidas por los alumnos}

write('Cuántos alumnos hay en clase?: ');
readln(N);
veces0 := 0;
veces5 := 0;
veces10 := 0;
{Se inicializan a cero los contadores que guardarán las veces que
se obtuvieron puntuaciones de 0, 5, y 10 }

for index := 1 to N do
{N es el número de alumnos.}
{Repetir N veces el siguiente proceso:}

     begin
     write('Nota de alumno(',index,'): ');
     readln(alumno);
     {Leer la nota del alumno. Si la nota no esta en el conjunto,
     entonces se añade:}

     if not(alumno in notas) then notas := notas + [alumno];
     case alumno of
     {si la nota es cero, cinco o diez, se incrementa en uno el contador
     correspondiente: veces0, veces5 o veces10}

       0: veces0 := veces0 + 1;
       5: veces5 := veces5 + 1;
      10: veces10 := veces10 + 1
          end
     end;
writeln;
writeln('Número de alumnos con un cero: ',veces0); {Se muestran los }
writeln('Número de alumnos con un cinco:',veces5);      {resultados }
writeln('Número de alumnos con un diez: ',veces10);
writeln;
writeln('Ningún alumno ha obtenido ninguna de las siguientes puntuaciones:');
{Se muestran las notas que no est n en el conjunto, que no estar n, porque
ningún alumno habrá  obtenido esa calificación.}

for index := 0 to 10 do
     if not(index in notas) then write(index,'  ');
readkey
end.
```

Decir el numero de vocales que aparecen en una frase introducida por el usuario. Debe acabar con un punto.

```
PROGRAM EJER004;
Uses Crt;
Const vocales: Set of Char = (['A','E','I','O','U']);
{Poniendolas en mayusculas conseguimos unificar el valor de las vocales
 y evitamos decir al usuario que las introduzca en mayusculas o minusculas}
var cont, i:Integer;
var letra: Char;
var frase: Array[1..85] of Char;

Begin
     ClrScr;
     WRITELN('Escriba una frase, acabando con un punto'); WRITELN;
     i:=0;
     cont:=0;

     REPEAT

     letra := Readkey;{Readkey no mostrara los caracteres en pantalla}
     WRITE (letra); {Debemos incluirlo para que nos muestre los caracteres}
```

```
     if UPCASE(letra) in vocales then {preguntamos si la letra introducida es una vocal}
        inc(cont);

     frase[i]:=letra; {guardamos el valor de la letra en el array frase}

     UNTIL letra = '.';
     WRITELN;
     WRITELN;
     WRITE ('El numero de vocales introducidas es ',cont);
End.


Program EJ004(Input, Output);
Uses Crt;
Const vocales: Set of Char = (['A','E','I','O','U',' ',',','¡','ó','£',' ']);
      LetBuenas: Set of Char = (['a'..'z','A'..'Z','0'..'9',
          ' ','.',',',':',';','(',')','-','¿',' ',',','¡','ó','£',
          ' ','?','','!','"','%','/','<','>']);

Var  letra: Char;       {para guardar cada carácter introducido desde teclado.}
     Frase: array [1..80] of Char;
     index, numvocales: integer;

Begin
ClrScr;
Writeln('Escribe tu frase, y termina con un punto.');
index := 0;
numvocales := 0;
     repeat
     letra := readkey;
     if letra in letbuenas then
     {para no guardar caracteres especiales.}

          begin
          inc(index);
          {incrementar el ¡índice del array.}

          write(letra);
          {Readkey no muestra por pantalla el carácter leído.}

          frase[index] := letra;
          {se asigna el carácter al array.}

          if upcase(letra) in vocales then inc(numvocales)
          {Si el carácter está en el conjunto "vocales" se incrementa el
          contador de vocales "numvocales":}

          end
     until (index >= 80) or (letra = '.'); {La frase termina con un punto.}
writeln;
writeln;
writeln('La frase tiene ',numvocales,' vocales.');
readkey
end.
```

Pedir que se introduzca una frase, acabada con un punto, en la que se lean las letras del abecedario introducidas y se muestren posteriormente.

```
PROGRAM EJER005;
Uses Crt;
         var caracter : Char;
         var abc : String;
         var i, longitud: Integer;

Begin
     ClrScr;

     WRITELN('Escriba una frase, terminando con un punto');
     WRITELN;

     i := 0;
     longitud := 0;
```

```
     REPEAT
           caracter := Upcase(Readkey);
           WRITE(caracter);
           If caracter in ['A'..'Z'] then
              Begin
                   abc[i] := caracter;
                   {el array abc quedara con espacios vacios,los que no sean
                   letras del abedecario, si no se pone el incremento de i dentro+
                   de este begin-end. Prueba a quitar el begin-end si tienes
                   curiosidad en verlo}
                   inc(i);
              End;
     UNTIL caracter = '.';
     longitud := i;

     WRITELN;
     WRITELN;
     WRITELN('Las letras del abecedario introducidas son: ');
     WRITELN;

     For i := 0 To longitud Do
         WRITE(abc[i],' ');
End.

program EJ005(Input, Output);
Uses Crt;
var Estan: Set of Char;
    Entrada: Char;
begin
clrscr;
Writeln('Introduce una frase, y termina con un * (asterisco):');
writeln;
Estan := [];
{Se inicializa el conjunto "Estan" como vacío}

     repeat
     entrada := upcase(readkey);
     {Se lee de teclado un carácter, y se pasa a mayúsculas con
     la función upcase().}

     if entrada in ['A'..'Z',' ',',','.',';'] then write(entrada);
     {Si el carácter introducido es una letra, un signo de puntuación,
      o un espacio en blanco, entonces se muestra por pantalla. }

     if not(entrada in Estan) then Estan := Estan + [entrada]
     {Si el carácter no esta en el conjunto "Estan", entonces se añade}

     until entrada = '*';
{Se repite el proceso hasta que se introduzca un *}

writeln;
writeln;
writeln('Las siguientes letras han aparecido:');
{Se procede a mostrar los caracteres que pertenecen al conjunto}

for entrada := 'A' to 'Z' do
{Se recorren los valores desde la A hasta la Z}

     if entrada in Estan then write(entrada,' ');
     {para cada letra, si est  en el conjunto "Estan", significa
     que ha sido introducido por teclado, y entonces se muestra
     por pantalla}

writeln;
writeln;
writeln('Las siguientes letras NO han aparecido:');
{Ahora se procede a mostrar los que no pertenecen al conjunto. Para ello
se sigue el mismo proceso que antes, pero mostrando la letra sólo si NO
pertenece al conjunto. }

for entrada := 'A' to 'Z' do
     if not(entrada in Estan) then write(entrada,' ');

readkey
end.
```

Escribir un programa en Pascal que lea una frase introducida desde el teclado y la escriba al revés.

```
PROGRAM EJER006;
Uses Crt;
         var frase: String; {Se puede hacer con arrays}
         var f_inv: String; {Cada uno debe elegir la manera que mejor entida,
                             y controle, eso si, siempre hay que dominar las
                             dos formas.}
         var i: Integer;
Begin
     ClrScr;

     i := 0;

     WRITELN('Escriba una frase:');
     READLN(frase);
     WRITELN;
     WRITELN;

     For i := 0 to length(frase) do {desde la primera posicion de la frase hasta
                                     la ultima almacenamos la frase en una variable}
         f_inv[i] := frase[i];

     FOR i := length(frase) downto 0 do
           WRITE(f_inv[i]);
End.

PROGRAM EJER006_2;
Uses Crt;
         var frase: String;
         var i: Integer;
Begin
     ClrScr;

     i := 0;

     WRITELN('Escriba una frase:');
     WRITELN;
     READLN(frase);
     WRITELN;

     FOR i := length(frase) downto 1 do {Si ponemos "downto 0" nos pasariamos en un
caracter}
           WRITE(frase[i]);

     {Si lo unico que quieres es mostrar la cadena de caracteres a la inversa,
      en esta version lo hemos hecho unicamente con una variable}
End.

Program EJ006(Input, Output);
Uses Crt;
Const long_frase = 80;
Type frase = array [1..long_frase] of Char;
Var  Frase1: frase;
     iguales: boolean;
     index, max: integer;

     Procedure leerfrase(var arraychar: frase; var index: integer);
     {Lee una frase desde teclado, y la almacena en un array.
     Además, devuelve en una variable entera la longitud de la frase.}

     var letra: Char;

     begin
     index := 0;
     repeat
          letra := readkey;
          inc(index);
          write(letra);
          arraychar[index] := letra;
     until (index >= long_frase) or (letra = '.');
```

```
     writeln
     end;

Begin
ClrScr;
writeln('Visualizar una frase al revés.');
Writeln('Escribe la frase, y termina con un punto:');
leerfrase(frase1,max);
for index := max downto 1 do write(frase1[index]);
{Para visualizar la frase al revés, se recorre el array que la contiene
empezando desde el final, que se ha guardado en la variable "max".}

writeln;
readkey
end.
```

Escribir un programa que compare dos arrays de caracteres y nos diga si son idénticos o no.

```
PROGRAM EJER007;
        Uses Crt;
        var frase1, frase2: String;
        var index,long1,long2: Integer; {long1 y long2 son la longitud de las cadenas}
        var letra: Char;
        var iguales: Boolean;
Begin
     ClrScr;

     WRITELN ('Introduzca la primera cadena de caracteres, finalizando con un punto');
     WRITELN;

     index := 0;
     long1 := 0;
     long2 := 0;

     REPEAT
     Begin
          letra := Readkey;
          write(letra); {Readkey no muestra los caracteres en pantalla, write si}
          frase1[index] := letra; {vamos almacenando cada letra en la variable}
          index := index + 1;
     End;
     UNTIL letra='.';
     long1 := index; {medimos la longitud de la cadena}

     WRITELN;
     WRITELN;
     WRITE('Introduzca la segunda cadena de caracteres, finalizando con un punto');
     WRITELN;
     WRITELN;

     letra := ' ';
     index := 0;

     REPEAT
     Begin
          letra := Readkey;
          write(letra);
          frase2[index] := letra; {vamos almacenando cada letra en la variable}
          index := index + 1;
     End;
     UNTIL letra='.';
     long2 := index; {medidos la longitud de la cadena}

     WRITELN;
     WRITELN;

     If long1 <> long2 then {Si la longitud es distinta, SEGURO que no son iguales}
         WRITELN ('Las cadenas de caracteres son distintas')
         {*Pongo dos mensajes distintos para que veamos en que bucle ha entrado
           en los diferentes casos que se pueden dar}
     Else
         Begin
           For index := 0 to long1 do
           Begin
```

```
               if frase1[index] <> frase2[index] then
                  Begin
                   WRITE('Las cadenas de caracteres no son iguales');
                     {*Pongo dos mensajes distintos para que veamos en que bucle ha
                     entrado en los diferentes casos que se pueden dar}
                   exit;
                  End
               else
                   iguales := true; {almacenamos en una variable que las cadenas son
                                     iguales}
          End;
        End;

        If iguales = true then WRITELN ('Las cadenas de caracteres son iguales');
End.


Program EJ007(Input, Output);
Uses Crt;
Const long_frase = 80;    {Máxima longitud permitida para una frase.}

Type frase = array [1..long_frase] of Char;

Var  Frase1, frase2: frase;  {Arrays donde se almacenarán las frases.}
     iguales: boolean;
     index: integer;

     Procedure leerfrase(var arraychar: frase);
     {Lee desde teclado una frase, y la almacena en un array de caracteres.}

     var letra: Char;
         index: integer;
     begin
     index := 0;
     repeat
          letra := readkey;
          inc(index);
          write(letra);
          arraychar[index] := letra;
     until (index >= long_frase) or (letra = #13);
     {La frase termina con INTRO   }
     writeln
     end;
Begin
ClrScr;
Writeln('Escribe la primera frase, y termina con INTRO.');
leerfrase(frase1);
{Leer la primera frase}
Writeln('Escribe la segunda frase, y termina con INTRO.');
leerfrase(frase2);
{Leer la segunda frase}
index := 1;
iguales := (frase1[index] = frase2[index]);
{Se inicializa "iguales" a true o false según el primer carácter de
"frase1" sea igual al primer carácter de "frase2" }

while iguales and (index <= long_frase) and (frase1[index] <> #13) do
     {Mientras que iguales sea true y no se alcance el final de la frase,
     que puede ser porque se detecte un INTRO o porque se llegue a la
     longitud máxima de frase "long_frase".}
     begin
     inc(index);
     iguales := (frase1[index] = frase2[index])
     end;
if iguales then writeln('Las dos frases son idénticas.')
     else begin
          writeln('Las frases NO son idénticas.');
          writeln('Difieren a partir del carácter nº', index)
          end;
readkey
end.
```

Escribir un programa en Pascal que sume los valores que hay por encima de la diagonal principal. Los valores se pueden asignar como una constante.

```
PROGRAM EJER008;
Uses Crt;
        Const N = 4; {Numero de filas y columnas de la matriz}
        Const matriz: Array [1..N,1..N] of Integer = ((1, 2, 3, 4),
                                                      ( 5, 6, 7, 8),
                                                      ( 9,10,11,12),
                                                      (13,14,15,16));
        {Declaramos la matriz}
        var i, j, suma: Integer;
        {i se corresponde con las filas y j con las columnas}
Begin
     ClrScr; {limpiamos la pantalla}

     {los valores de la diagonal principal son (1,1) (2,2) (3,3) (4,4).
      Como este ejercicio consiste en sumar los valores que hay encima de
      la diagonal principal, sumares: 2 + 3 + 4 + 7 + 8 + 12}

     {Escribimos la matriz original y coloreamos los valores que necesitamos}

     For i := 1 to N Do
         Begin
              For j := 1 to N Do
                  Begin
                       If j > i then
                           Textcolor(9) {Cambiando el numero elegiremos el color}
                       Else
                           TextColor(7);
                       WRITE(matriz[i,j]:3);
                  End;
              WRITELN(' ');
         End;

     WRITELN;{Suma de los valores de la primera fila de la matriz}

     i := 1;

     FOR j := 2 to N do
         Begin
              suma := suma + matriz[i,j];
              WRITELN(suma); {Lo escribimos para chequear los valores, podriamos
                              escribir un write al final y seria suficiente}
         End;

     {Suma de los valores de la segunda fila}

     i := 2;

     FOR j := 3 to N do
         Begin
              suma := suma + matriz[i,j];
              WRITELN(suma);
         End;

     WRITELN;{Suma de los valores de la tercera fila}

     i := 3;

     FOR j := 4 to N do {N ya vale 4, por lo que solo se realiza 1 vez el bucle}
         Begin
              suma := suma + matriz[i,j];
              WRITELN('El resultado final es: ',suma);
         End;
End.

Program t7e20(Input, Output);
Uses Crt;
Const N = 4;
      mat1: array [1..N, 1..N] of integer
                  = ( ( 1, 2, 3, 4),
                      ( 5, 6, 7, 8),
                      ( 9,10,11,12),
                      (13,14,15,16) );

Type matriz = array [1..N, 1..N] of integer;
```

```
Var  col, mfila: integer;
     suma: integer; {Ir  acumulando la suma de los elementos deseados.}

Begin
ClrScr;
writeln;
writeln('   Se procede a sumar los elementos de la matriz');
writeln('   que se encuentren por encima de la diagonal: ');
for mfila := 1 to N do
     for col := 1 to N do
     {Todos los elementos por encima de la diagonal, cumplen la propiedad
     de ser su ¡índice de columna mayor que su ¡índice de fila}
          begin
          if col > mfila then textcolor(12)
               else textcolor(7);
          {se escribir n en rojo los elementos a sumar.}

          gotoxy(4*mcol+18,mfila+4);
          write(mat1[mfila,mcol]:2)
          end;
suma := 0;
for mfila := 1 to N do
     for col := mfila+1 to N do
          suma := suma + mat1[mfila,mcol];
writeln;
writeln;
writeln('   La suma de los elementos');
write('   por encima de la diagonal es: ');
textcolor(12);
writeln(suma);
textcolor(7);
readkey
end.
```

Escribir un programa en Pascal que almacene en un array de registros los nombres de los alumnos, sus notas parciales y finales. Hallar la nota media y mostrar un mensaje de APTO si el alumno supera o iguala la calificación de 5 o NO APTO si no lo alcanza. Hacerlo para un número de 5 alumnos.

```
PROGRAM EJER009;
Uses Crt;
         Const numalumnos = 5;

         Type tnotas = record
              nombre2 : String;
              n_parcial2, n_final2: Real; {Nota parcial y final}
              end;
         {Hacemos una fila dividida en 4 partes:tnotas, nombre2,
          n_parcial2, n_final2. Es como una tabla de Word, la cual
          iremos rellenando con los datos obtenidos}

         notas = Array[1..numalumnos] of tnotas;
         {Ahora la copiamos tantas veces como numero de alumnos hay,
          es decir, obtenemos 5 filas iguales}

         var clase : notas;
         var nombre: String;
         var n_parcial, n_final, n_media: Real;
         var i: Integer;
Begin
     ClrScr;

     For i := 1 to numalumnos Do
         Begin
               WRITE('Introduzca el nombre del alumno ',i,' : ');
               READLN(nombre);
               WRITE('Introduzca su nota parcial: ');
               READLN(n_parcial);
               WRITE('Introduzca su nota final: ');
               READLN(n_final);
               WRITELN;
               With clase[i] Do
```

```
                    Begin
                         n_parcial2 := n_parcial;
                         n_final2 := n_final;
                         nombre2 := nombre;
                    End;
          End;

          ClrScr;

          WRITELN('NOMBRE':25,'Parcial':8,'Final':8,'Media':8,'CALIFICACION':15);
          WRITELN;

          For i := 1 to numalumnos do
              With clase[i] do
                   Begin
                        n_media := (n_parcial2 + n_final2) / 2;
                        WRITE(nombre2:25,n_parcial2:8:2,n_final2:8:2);
                        textcolor(14); WRITE(n_media:8:2);
                        If n_media >= 5 then
                          Begin
                              textcolor(11);
                              WRITELN ('APTO  :-)':15);
                          End
                        Else
                          Begin
                              textcolor(1);
                              WRITELN ('NO APTO :-(':15);
                          End;
                        textcolor(7);
                   End;
End.

Program EJ009(Input, Output);
Uses Crt;
Const numalumnos = 5;
Type tiponotas = record
           nombre: String;
           parcial, final: real
           end;
     notasclase = array [1..Numalumnos] of tiponotas;
Var  I3: notasclase;
     nota1, nota2: real;
     alumno: String;
     index: integer;
Begin
ClrScr;
for index := 1 to numalumnos do
     begin
     write('Nombre de alumno(',index,'): ');
     readln(alumno);
     write('Nota del examen parcial: ');
     readln(nota1);
     write('Nota del examen final: ');
     readln(nota2);
     writeln;
     with i3[index] do
          begin
          nombre := alumno;
          parcial := nota1;
          final := nota2
          end
     end;
ClrScr;
writeln('NOMBRE ':30,'Parcial':10,'Final':10,'Media':10,'  CALIFICACION');
for index := 1 to 75 do write('-');
writeln;
for index := 1 to numalumnos do
     with i3[index] do
          begin
          {Escribir la lista con los resultados.}
          nota1 := (parcial+final)/2;
          {Se calcula la media.}
          write(nombre:30,parcial:10:2,final:10:2);
          write(nota1:10:2);
          {Si la nota media es superior a 5, el alumno est  aprobado:}
          if nota1 >= 5 then writeln('  *** APTO *** ')
```

```
           else writeln('    NO APTO')
         end;
readkey
end.
```

Escribir un programa en Pascal que almacene en un array de registros las características de cada persona: nombre, sexo, edad, peso, color de pelo, color de piel, color de ojos, nacionalidad y teléfono.

```
PROGRAM EJER010;
Uses Crt;
         Const numpersonas = 2; {Cambiando este valor lo podremos hacer para
                                 el numero de personas que deseemos}
         Type caracteristicas = record
              nombre2, nacionalidad2, sexo2: String;
              edad2: Integer;
              c_ojos2: Char;
              tf2: Real;
         {Creamos una fila con diferentes apartados}
              end;

              personas = Array[1..numpersonas] of caracteristicas;
         {La copiamos tantas veces como personas haya}

         var persons : personas;
             nombre, nacionalidad, sexo: String;
             edad, i: Integer;
             c_ojos: Char;
             tf: Real;
Begin
     ClrScr;

     For i := 1 to numpersonas do
     Begin
          WRITELN('Introduzca los datos de la persona numero ',i,' : ');
          WRITELN;
          WRITE('Nombre:       '); READLN(nombre);
          WRITE('Edad:         '); READLN(edad);
          WRITE('Nacionalidad: '); READLN(nacionalidad);
          Repeat
                WRITE('Sexo (H, M):  '); READLN(sexo);
          Until (sexo = 'H') or (sexo = 'M') or (sexo = 'h') or (sexo = 'm');
          WRITE('Telefono:     '); READLN(tf);
          Repeat
          WRITE('Color de ojos (A, V, M):  '); READLN(c_ojos);
                c_ojos := UPCASE(c_ojos);
          Until (c_ojos = 'A') or (c_ojos = 'V') or (c_ojos = 'M');
          WRITELN;
          With persons[i] do
            Begin
              nombre2 := nombre;
              edad2 := edad;
              nacionalidad2 := nacionalidad;
              If (sexo = 'H') or (sexo = 'h') then
                  sexo2 := 'Si gracias';

              tf2 := tf;
              c_ojos2 := c_ojos;
              {Almacenamos los datos dentro del array de registro}
            End;
     End;
     textcolor(11);
     WRITELN('Nombre':14,'Edad':6,'Nacionalidad':14,'Sexo':12,'Telefono':12,'Color
ojos':12);
     textcolor(7);
     For i := 1 to numpersonas do
         Begin
              with persons[i] do
                   Begin
                        WRITELN(nombre2:14,edad2:6,Nacionalidad2:14,sexo2:12,tf2:12:0,c_o
jos2:12);
                   End;
         End;
```

```
     {Las caracteristicas que se piden al usuario para obtener informacion de las
      diferentes personas son simples ejemplos. Si se quieren cambiar, y poner
      por ejemplo, si esta casado o soltero, sus estudios, etc., unicamente habra
      que crear nuevas variables o sustituir las existentes.}
End.

Program EJ010B(Input, Output);
Uses Crt;
Const totalmuestreo = 5;
Type datos = record
          nombre: String[25];
          nacion, region: String[11];
          edad: integer;
          altura: real;
          sexo, ColOjos, colPelo, colPiel: Char
          end;
     estadistica = array [1..totalmuestreo] of datos;

Var  Grupo1: Estadistica;
     nom: String[25];
     cont, index: integer;

Begin
ClrScr;
index := 1;
repeat
{Repetir hasta que se introduzca un nombre en blanco:}
     with grupo1[index] do
     begin
     write('Nombre (',index,'): ');
     readln(nom);
     {Leer el nombre.}
     if not(nom = '') then
     {Si no se introdujo un nombre vacío, entonces leer el resto de datos.}
          begin
          nombre := nom;
          write('País de origen: ');
          readln(nacion);
          write('región: ');
          readln(region);
          write('Sexo [V/M]: ');
               repeat
               sexo := upcase(readkey)
               until sexo in ['V','M'];
               {restringe la entrada a "V" o "M".}
          writeln(sexo);
          write('Altura [x.xx metros]: ');
          readln(altura);
          write('Color de ojos [V, A, M, N, G]: ');
               repeat
               colojos := upcase(readkey)
               until colojos in ['V','A','M','N','G'];
          writeln(colojos);
          write('Color de piel [N, B, A, R]: ');
               repeat
               colpiel := upcase(readkey)
               until colpiel in ['N','B','A','R'];
          writeln(colpiel);
          write('Color de pelo [N, B, C, R, P]: ');
               repeat
               colpelo := upcase(readkey)
               until colpelo in ['N','B','C','R','P'];
          writeln(colpelo);
          writeln;
          index := index + 1
          {Se incrementa el número de muestras.}
          end
     end
until (index >= totalmuestreo) or (nom = '');

ClrScr;
write('NOMBRE':25,'NACION':12,'REGION':12,' EDAD',' SEXO',' ALT.');
writeln(' OJOS',' PIEL',' PELO');
for cont := 1 to 79 do write('Í');
writeln;
for cont := 1 to index do
```

```
{Presentar los resultados por pantalla.}

     with grupo1[cont] do
     {Cada elemento del array es un registro.}
          begin
          write(nombre:25,nacion:12,region:12,edad:5,sexo:5,altura:5:2);
          writeln(colojos:4,colpiel:5,colpelo:5)
          end;
readkey
end.
```

Escribir un programa que lea dos números enteros A y B, y obtenga los valores A div B, A mod B.

```
PROGRAM EJERDIV;
        Uses Crt;
             Var A,B: Integer;
             Var soluc: Integer;
Begin
     ClrScr;
     WRITELN('Introduzca dos numeros:');
     WRITELN;
     WRITE('A: '); READLN(A);
     WRITE('B: '); READLN(B);
     WRITELN;

     WRITE('A div B = ');
              soluc := A div B; {div hace la division de 2 numeros enteros}
     WRITELN(soluc);
     WRITELN;

     WRITE('A mod B = ');
              soluc := A mod B; {mod muestra el resto de una division de
                                 2 numeros enteros}
     WRITELN(soluc);
End.


program ejerdivb(Input, Output);
Uses Crt;
var A, B, aDb, aMb: integer;
begin
ClrScr;
write('Dime un número entero:');
readln(A);
write('Dime otro número entero:');
readln(B);
aDb := A div B;
aMb := A mod B;
writeln('A div B = ',aDb);
writeln('A mod B = ',aMb);
readkey
end.
```

Escribir un programa que convierta un número de segundos en su equivalente en minutos y segundos.

```
PROGRAM EJERSEG;
        Uses Crt;
             var seg0,seg,min: Integer;
Begin
     ClrScr;
            WRITE('Introduzca los segundos: '); READLN(seg0);
            WRITELN;

            min := seg0 div 60;
            seg := seg0 mod 60;

            WRITE(seg0,' segundos son ',min,' minutos y ',seg,' segundos.');
End.
```

```
program ejersegb(Input, Output);
uses Crt;
var iniseg, segundos, minutos: integer;
begin
ClrScr;
write('Dime un número de segundos:');
readln(iniseg);
minutos := iniseg div 60;
{Cada 60 segundos, son 1 minuto}

segundos := iniseg mod 60;
{Son los segundos que sobran de hacer grupos de 60 segundos}

writeln(iniseg,' segundos son ',minutos,' minutos y ',segundos,' segundos.');
readkey
end.
```

Imprimir la media de los elementos que se encuentran en las posiciones pares y la media de los elementos que se encuentran en las posiciones impares de un vector numérica.

```
PROGRAM EJEMEDIA;
        Uses Crt;
             var sumapar,sumaimp,n_par,n_imp: Integer;
             var i: Integer;
             var media_p,media_i: Real;
             const num=10;
             var numeros: Array[1..num] of Real;
Begin
     ClrScr;
            WRITELN('Introduzca los ',num,' numeros');
            WRITELN;

            sumapar := 0;
            sumaimp := 0;
            n_par   := 0;
            n_imp   := 0; {Inicializamos las variables a 0 para evitar sorpresas}

            For i := 1 to 10 do
                Begin
                     WRITE('Introduzca el numero ',i,' : '); READLN(numeros[i]);
                     If num mod 2 = 0 then
                     {para que sea par, el resto de una division debe ser 0}
                        Begin
                           n_par := n_par + 1; {tambien se puede hacer con “inc(n_par)"}
                           sumapar := sumapar + num;
                        End
                     Else {si no es par, DEBE ser impar}
                        Begin
                             n_imp := n_imp + 1;
                             sumaimp := sumaimp + num;
                        End;
                End;

            {Vamos a hallar la media de los pares y los impares}

            WRITELN;
            media_p := sumapar / n_par;
            media_i := sumaimp / n_imp;
            WRITELN('La media de los numeros pares es: ',media_p:5:2);
            WRITELN;
            WRITELN('La media de los numeros impares es: ',media_i:5:2);
End.


Program ejemediab(Input, Output);
uses Crt;
Const maxnum = 5;
Type listadenumeros = array [1..maxnum] of real;
Var  lista: listadenumeros;
     pares, impares, index: integer;
     imedia, pmedia, isuma, psuma: real;
```

```
Begin
ClrScr;
writeln('Dada una lista de números, calcular la media de los que ocupan ');
writeln('posiciones pares, y la de los que ocupan posiciones impares.');
writeln;
writeln('Introduce los ',maxnum,' números de la lista:');
isuma := 0;   {acumulador de impares}
psuma := 0;   {acumulador de pares}
pares := 0;   {contador de pares}
impares := 0; {contador de impares}
for index := 1 to maxnum do
     begin
     write('Elemento ',index,': ');
     readln(lista[index]);       {leer elemento de la lista.}
     if odd(index) then
     {si ocupa posición impar:}

          begin
          inc(impares);
          {incrementar contador de números impares,}

          isuma := isuma + lista[index]
          {sumar al acumulador de impares}
          end
     else begin
     {si no ocupa posición impar:}

          inc(pares);
          {incrementar contador de números pares,}

          psuma := psuma + lista[index]
          {sumar al acumulador de pares}
          end
     end;
imedia := isuma / impares; {calcular la media de impares}
pmedia := psuma / pares;   {calcular la media de pares}
writeln;
writeln(impares,' elementos impares y ',pares,' elementos pares.');
writeln;
writeln('Media de los elementos impares: ',imedia:10:5);
writeln('Media de los elementos pares..: ',pmedia:10:5);
readkey
end.
```

Escribir un programa que muestre en pantalla VS2 realizado por nosotros (usar procedimiento).

```
PROGRAM VS2 (INPUT, OUTPUT);
        Uses Crt;

        PROCEDURE letra_V;
                BEGIN   WRITELN('V       V');
                        WRITELN(' V     V ');
                        WRITELN('  V   V  ');
                        WRITELN('   V V   ');
                        WRITELN('    V    ');
                END;
        PROCEDURE letra_S;
                BEGIN
                        WRITELN('SSSSSSS');
                        WRITELN('S      ');
                        WRITELN('SSSSSSS');
                        WRITELN('      S');
                        WRITELN('SSSSSSS');
                END;
        PROCEDURE num_2;
                BEGIN
                        WRITELN('2222222');
                        WRITELN('      2');
                        WRITELN('2222222');
                        WRITELN('2      ');
                        WRITELN('2222222');
                END;
BEGIN {empezamos el programa principal}
```

```
        ClrScr;
        letra_V; WRITELN;
        letra_S; WRITELN;
        num_2;
        REPEAT Until Keypressed; {mantenemos la pantalla viendo la solución hasta que se
pulse una tecla}
END.
```

Hacer un programa que incremente un número usando un procedimiento.

```
PROGRAM incrementar (INPUT, OUTPUT);
        Uses Crt;
             VAR num: INTEGER;
             PROCEDURE incremento;
                  BEGIN
                       num := num + 1;
                  END;
BEGIN
     ClrScr;
     WRITE('Introduzca un numero para incrementarle: '); READLN(num);
     WRITELN;
     incremento;
     WRITE('El numero, incrementado en una unidad, es: ',num);
     REPEAT Until Keypressed;
END.
```

Escribir un programa que, utilizando procedimientos con parámetros, lea desde el teclado las unidades y el precio que quiere comprar, y en función de las unidades introducidas le haga un descuento o no.

```
PROGRAM productos (INPUT, OUTPUT);
Uses Crt;
CONST
     Desc = 15; {le haremos un 15% de descuento}
VAR
   Unidades, precio: INTEGER;
   Total, cantDesc: REAL;
PROCEDURE descuento (VAR cantidad, descuento: REAL; porciento: INTEGER);
          BEGIN
               Descuento := cantidad * porciento/100; {el descuento es el 15% del total}
               Cantidad := cantidad - descuento;
                                   {la cantidad final es la cantidad - el descuento}
          END;
BEGIN
     ClrScr;
     WRITE('Introduzca el numero de unidades: ');
     READLN(unidades);
     WRITELN;
     WRITE('Introduzca el precio: ');
     READLN(precio);
     WRITELN;
     Total := precio * unidades; {Calculamos el total}
     IF (unidades > 5) THEN descuento (total, cantDesc, desc) {aplicamos el descuento}
        ELSE cantDesc := 0;
     WRITELN('Total: ',total:5:2,' Descuento: ',cantdesc:5:2);
                                     {escribimos en pantalla el total y el descuento}
     REPEAT Until Keypressed;
END.
```

Hacer un programa que calcule el area de un círculo (usar un procedimiento).

```
PROGRAM area (INPUT, OUTPUT);
        Uses Crt;

             VAR radiocirc, resultado: REAL;
        PROCEDURE areacirculo (radio: REAL; VAR area: REAL);
             CONST pi = 3.1415926535;
             BEGIN
                  area := pi * SQR(radio);
```

```
            END;
BEGIN
     ClrScr;
            WRITE('Introduzca el radio del circulo: '); READLN(radiocirc);
            WRITELN;
            IF (radiocirc > 0) THEN
               BEGIN
                    areacirculo(radiocirc, resultado);
                              {radiocirc se corresponde con radio y resultado con area}
                    GOTOXY(20,5);
                    WRITELN('El area del circulo es: ',resultado:8:2);
               END
            ELSE
                WRITE('No puede introducir un radio negativo.');

     REPEAT Until Keypressed;
END.
```

Escribir un programa, que con funciones, verifique si un caracter introducido es un número o no.

```
PROGRAM escaracter (INPUT, OUTPUT);
        Uses Crt;
             VAR carac: CHAR;
        FUNCTION verificar (caracter: CHAR) : BOOLEAN;
            BEGIN
                 verificar := (caracter >= '0') AND (caracter <= '9');
            END;

BEGIN
     ClrScr;
            WRITE('Introduce un caracter para ver si es numerico.'); READLN(carac);
            WRITELN;
            IF verificar (carac) THEN
                WRITELN('El caracter introducido es numerico.')
            ELSE
                WRITELN('El caracter introducido no es numerico.');
     REPEAT Until Keypressed;
END.
```

Escribir un programa en Pascal que reciba un numero del 1 al 12 desde el teclado y muestre el número de dias correspondiente al mes que corresponda con ese día (usar funciones).

```
PROGRAM diames (INPUT, OUTPUT);
        Uses Crt;
             VAR mes: INTEGER;
        FUNCTION dia_mes (i: INTEGER): INTEGER;
                  BEGIN
                       CASE i OF
                            1,3,5,7,8,10,12: dia_mes := 31;
                            4,6,9,11: dia_mes := 30;
                            2: dia_mes := 28; {emitiremos un mensaje diciendo que puede
ser bisiesto}
                       END;
                  END;
BEGIN
     ClrScr;
            WRITE('Introduzca un numero del 1 al 12: '); READLN(mes);
            WRITELN;
            IF (mes < 1) OR (mes > 12) THEN
               WRITE('El numero introducido no corresponde a ningun mes.')
            ELSE IF mes = 2 THEN
               WRITE('Febrero tiene 28 dias, si es bisiesto 29')
            ELSE
               WRITE('El mes tiene ',dia_mes(mes),' dias.');
     REPEAT Until Keypressed;
END.
```

Eliminar los espacios que existen delante del caracter salto de carro de un vector que

contiene un texto de tamaño N. El final del texto de marcará con *.

```
PROGRAM texto (INPUT, OUTPUT);
        Uses Crt;
             CONST longitud = 80;
             Type frase = array [1..longitud] of CHAR;
             VAR frase1: frase;
                 blancos, conta, cont2, long: INTEGER;
                 letra: CHAR;

             PROCEDURE leerfrase(var arraychar: frase; var pos_intro, index: integer);

                       VAR letra: CHAR;
                           pulsado: boolean;

                       BEGIN
                            WRITELN('La frase acaba cuando se pulse *');
                            pulsado := false;
                            i := 0;
                       REPEAT
                             letra := readkey;
                             inc(i);
                             IF letra <> #13 then
                                BEGIN
                                     WRITE(letra);
                                     arraychar[i] := letra;
                                END
                             ELSE
                                BEGIN
                                     IF NOT pulsado THEN
                                        BEGIN
                                             pulsado := true;
                                             write('<enter>');
                                             arraychar[i] := letra;
                                             pos_intro := i
                                        END
                                      ELSE
                                          dec(i);
                                END;
                       UNTIL (i >= longitud) or (letra ='*');
                       IF letra ='*' THEN
                          des(index);
                       WRITELN;
             END;
BEGIN {programa principal}
      ClrScr;
             WRITELN('Escribe una frase con blancos y un INTRO en medio.');
             leerfrase(frase1,conta,long);
             cont2=conta - 1; {Posicion anterior a la del INTRO}
             numblancos := 0;
             WHILE (frase1[cont2] = '')and(conts >=1) DO
                   BEGIN
                        inc(blancos);
                        dec(cont2){decrementar la posicion del array para comprobarla}
                   END;

          {Para eliminar los blancos, se copian los caracteres que ocupan las posiciones
              siguientes a las del INTRO, encima de las que ocupan los blancos hasta el
              final de la frase.}

             FOR cont2 := (conta - blancos) to (long - numblancos) DO
                 frase[cont2] := frase1[cont2 + blancos];

             WRITELN;
             WRITELN('La frase sin blancos antes del INTRO: ');
             WRITELN;
             FOR cont2 := 1 TO (long - numblancos) DO
                 IF frase1[cont2] <> #13 THEN
                    WRITE(frase1[cont2])
                 ELSE
                    WRITE('<enter>');

             WRITELN;
     Readkey
end.
```

Escribir un programa en Pascal que transforme numeros entre 0 y 999 a numeros romanos.

```
PROGRAM roma;
        Uses Crt;
             VAR contador, digitos: Integer;
             VAR num_romano, romano: String;
             VAR num: CHAR;

BEGIN
     ClrScr;
     textcolor(10);
     {7 es el color de las letras por defecto, con esta funcion cambiamos su valor}

            WRITELN('Mostraremos el equivalente en numeros romanos del numero que
                    desee.');
            WRITELN;
            WRITE('¿Cuantos digitos tiene el numero que va a introducir? ');
            READLN(digitos);
            WHILE (digitos > 3) or (digitos < 1) DO
                  BEGIN
                       WRITE('Debe estar entre 1 y 3, introduzca los digitos:');
                       READLN(digitos);
                           IF digitos = 1 THEN
                                contador := 1
                           ELSE IF digitos = 2 THEN
                                contador := 2
                           ELSE IF digitos = 3 THEN
                                contador := 3
                  END;

            WRITELN;
            WRITE('Introduzca el numero: ');


            REPEAT
                num := Readkey;
                WRITE(num);

                IF contador = 1 THEN
                   CASE num OF
                        '1': romano := 'I';
                        '2': romano := 'II';
                        '3': romano := 'III';
                        '4': romano := 'IV';
                        '5': romano := 'V';
                        '6': romano := 'VI';
                        '7': romano := 'VII';
                        '8': romano := 'VIII';
                        '9': romano := 'IX';
                        '0': romano := '';
                   END
                ELSE IF num = '.' THEN
                     CONTINUE
                ELSE IF contador = 2 THEN
                   CASE NUM OF
                        '1': romano := 'X';
                        '2': romano := 'XX';
                        '3': romano := 'XXX';
                        '4': romano := 'XL';
                        '5': romano := 'L';
                        '6': romano := 'LX';
                        '7': romano := 'LXX';
                        '8': romano := 'LXXX';
                        '9': romano := 'XC';
                        '0': romano := '';
                   END
               ELSE IF num = '.' THEN
                    CONTINUE
               ELSE IF contador = 3 THEN
                    CASE num OF
                         '1': romano := 'C';
```

```
                        '2': romano := 'CC';
                        '3': romano := 'CCC';
                        '4': romano := 'CD';
                        '5': romano := 'D';
                        '6': romano := 'DC';
                        '7': romano := 'DCC';
                        '8': romano := 'DCCC';
                        '9': romano := 'CM';
                        '0': romano := '';
                    END;

                    num_romano := num_romano + romano;
                    contador := contador - 1 ;

            UNTIL contador < 1;

            WRITELN;
            WRITELN;
            IF num_romano = '' THEN
               WRITE('Los romanos no usaban el 0!') {Es cierto, como lo harian sin el}
            ELSE
                BEGIN
                      WRITE('En numeros romanos es igual a: ');
                      textcolor(9);
                      WRITE(num_romano);
                END;

            REPEAT Until Keypressed;
END.
```

Hacer un program que lea los 3 lados de un triangulo desde el teclado y nos diga si es equilatero (3 lados iguales), isosceles (2 lados iguales) o escalano (3 lados desiguales).

```
PROGRAM triangulo;
        Uses Crt;
             VAR lado1, lado2, lado3: REAL;

BEGIN
     ClrScr;

     WRITELN('Introduzca los 3 lados de un triangulo:');
     WRITELN;

     WRITE('Lado 1: '); READLN(lado1);
     WRITE('Lado 2: '); READLN(lado2);
     WRITE('Lado 3: '); READLN(lado3);
     WRITELN;

     IF (lado1 = lado2) and (lado2 = lado3) Then
           WRITE('El triangulo es equilatero.')
     ELSE IF (lado1 = lado2) OR (lado2 = lado3) OR (lado1 = lado3) Then
           WRITE('El triangulo es isosceles.')
     ELSE
           WRITE('El triangulo es escaleno.');

     REPEAT Until Keypressed;
END.
```

Decir si una frase es o no un palíndromo, es decir, si se lee igual de derecha a a izquierda que de izquierda a derecha.

```
PROGRAM palindromo;
 USES crt;
 VAR
    cad1,cad2: STRING;
    es_pal:boolean;  {es palíndromo - lo usamos para guardar en una variable la respuesta
                      a si es un palíndromo o no}

 PROCEDURE invertir(cad1:STRING; VAR cad2:STRING);
    VAR
```

```
      i:integer;
    BEGIN
      cad2:='';
      FOR i:=length(cad1) DOWNTO 1 DO
        cad2:=cad2+copy(cad1,i,1);
    END;

  PROCEDURE comparar(cad1:string;cad2:string;VAR sw:boolean);
    VAR
      i,j:integer;
      car1,car2:string;
    BEGIN
      es_pal:=true;
      IF length(cad1) <> length(cad2) THEN  {Si la longitud de las cadenas es distinta}
        es_pal:=false                       {no puede ser un palíndromo}
      ELSE
        BEGIN
          i:=1;j:=1;
          WHILE (es_pal) AND (i <= length(cad1)) DO
            BEGIN
              car1:=copy(cad1,i,1);
              car2:=copy(cad2,j,1);
              IF car1 = ' ' THEN
                BEGIN
                  i:=i+1;
                  car1:=copy(cad1,i,1);
                END;
              IF car2 = ' ' THEN
                BEGIN
                  j:=j+1;
                  car2:=copy(cad2,j,1);
                END;
              IF car1=car2 THEN es_pal:=true
              ELSE
                es_pal:=false;
                i:=i+1;j:=j+1;
            END;
        END;
    END;

BEGIN
    ClrScr;
    gotoxy(3,3); WRITE ('ESCRIBA UNA FRASE: ');
    READLN (CAD1);
    invertir(cad1,cad2);
    comparar (cad1,cad2,es_pal);
    IF es_pal = true THEN
      BEGIN
        gotoxy(10,13); WRITELN ('LA FRASE ESCRITA ES UN PALINDROMO');
      END
    ELSE
      BEGIN
        gotoxy(10,13);  WRITELN ('LA FRASE ESCRITA NO ES UN PALINDROMO');
      END;
    REPEAT Until Keypressed;
END.
```

Escribir un programa en Pascal que obtenga los factores primos de un número introducido desde el teclado.

```
PROGRAM factoresprimos;
USES crt;
VAR
   n:INTEGER;
procedure factores(n:integer);
   VAR
      i,j:INTEGER;
      fin:BOOLEAN;
   BEGIN
      WRITELN('Los factores primos de ',n,' son: ');
      WRITELN;
      FOR i:=n DOWNTO 1 DO
        BEGIN
```

```
            IF (n MOD i)=0 THEN
               BEGIN
                  fin:=false;
                  j:=i-1;
                  WHILE (i>1) AND (NOT fin) DO
                     BEGIN
                        IF (i MOD j)=0 THEN
                           fin:=true;
                        IF j=1 THEN
                           WRITE(i:2);
                        j:=j-1;
                     END;
               END;
         END;
   END;

BEGIN {programa principal}
   REPEAT
      ClrScr;
      WRITE('Introduzca un numero entero positivo: '); READLN(n);
      WRITELN;
   UNTIL n>0;
   factores(n);
   REPEAT UNTIL Keypressed;
END.
```

Escribir un programa en Pascal que escriba, por un lado, de la A a la Z en mayúsculas y por otro de z hasta a en minúsculas.

```
PROGRAM abc;
        USES crt;
            VAR
               cont_az:byte;
               cont_za:byte;
BEGIN
    ClrScr;

    cont_az:=65;  {Tomamos el valor donde comienza la letra A y llegaremos hasta el valor
de Z}
    cont_za:=122; {Tomamos el valoz de z y llegamos hasta a}
    gotoxy(5,5);

    REPEAT
      BEGIN
       WRITE (chr(cont_az):2);
       inc(cont_az);
      END
    UNTIL cont_az=91;

    gotoxy(5,10);

    REPEAT
      BEGIN
       WRITE (chr(cont_za):2);
       dec(cont_za);
      END
    UNTIL cont_za=96;

    REPEAT UNTIL Keypressed;
END.
```

Escribir un programa en Pascal que genere 6 números aleatorios con un rango de entre 1 y 49, al igual que se hace en la lotería.

```
PROGRAM loteriaprimitiva;
        USES crt;
            TYPE
               casilla=array[1..6] OF INTEGER;
            VAR
```

```
              num,posible,cont,i:INTEGER;
              ok:boolean;
              cas:casilla;
BEGIN
   ClrScr;

   num:=1;
   cont:=1;
   ok:=true;
   posible:=0;

   REPEAT
      IF cont=1 THEN
       BEGIN
         randomize;
         cas[num]:=random(49);
         inc(num);inc(cont);
       END
      ELSE
        BEGIN
         REPEAT
            posible:=random(49);ok:=true;
            FOR i:=1 TO 6 DO
             BEGIN
               IF cas[i]=posible THEN
                  ok:=false;
             END;
            IF ok THEN
              BEGIN
                cas[num]:=posible;
                inc(num);inc(cont);
              END
         UNTIL ok
        END
   UNTIL cont=7;
   gotoxy(10,10);

   FOR i:=1 TO 6 DO
      write(cas[i]:4);

   REPEAT UNTIL Keypressed;
END.
```

Escribir un programa en Pascal que realice un juego de dados entre 2 jugadores.

```
PROGRAM dados;
        USES crt;
                VAR
                   dado1,dado2,pos:BYTE;
                   suma1,suma2,sumat1,sumat2,tirada:INTEGER;
                   jugador:INTEGER;
                   nombre:STRING;
BEGIN
   ClrScr;

   dado1:=0;
   dado2:=0;
   suma1:=0;
   suma2:=0;
   tirada:=1;
   pos:=0;

   WRITE('Introduzca su nombre: '); READLN(nombre);

   randomize;

   gotoxy(25,3); WRITE(nombre,'          JUGADOR 2');
   gotoxy(25,4); WRITE('---------        ---------');
   gotoxy(10,6); WRITE('TIRADA 1');
   gotoxy(10,8); WRITE('TIRADA 2');
   gotoxy(10,10);WRITE('TIRADA 3');

   REPEAT
```

```
      dado1 := random(6) + 1;
      dado2 := random(6) + 1;
      gotoxy(26,6+pos); WRITE(dado1,' , ',dado2);
      suma1 := dado1 + dado2;
      sumat1 := sumat1 + suma1;
      dado1 := random(6) + 1;
      dado2 := random(6) + 1;
      gotoxy(44,6+pos); WRITE(dado1,' , ',dado2);
      suma2 := dado1 + dado2;
      sumat2 := sumat2 + suma2;
      inc(tirada);
      pos := pos + 2;
   UNTIL tirada = 4;

   IF sumat1 > sumat2 THEN
      BEGIN
       gotoxy(25,15); WRITELN('Ha ganado ',nombre,'. ENHORABUENA');
      END
     ELSE
      BEGIN
       IF sumat1 < sumat2 THEN
        BEGIN
           gotoxy(25,15); WRITELN('Ha ganado el jugador 2, usted pierde');
        END
       ELSE
        BEGIN
           gotoxy(25,15); WRITE('Han empatado')
        END;
      END;

   gotoxy(10,20); WRITE('La suma del jugador 1 es: ',sumat1);
   gotoxy(10,22); WRITE('La suma del jugador 2 es: ',sumat2);

   REPEAT UNTIL Keypressed
END.
```

Se pide un programa en PASCAL que lea una temperatura en la escala Fahrenheit, la convierta en la correspondiente temperatura en la escala Celsius, y muestre las dos temperaturas justificadas a la derecha. El programa principal ha de apoyarse en una función FaC que, dado un entero (la temperatura en la escala Fahrenheit), devuelva la correspondiente temperatura en la escala Celsius (redondeando).

```
PROGRAM grados(input,output);
   Uses Crt;

    VAR f,c:integer;   {temperatura en Fahrenheit y Celsius}

    FUNCTION FaC(fahrenheit:integer):integer;
        BEGIN {FaC}
             FaC:= round(5/9 * (f - 32.0))
        END; {FaC}

BEGIN {p.p}
   ClrScr;

     WRITE('Escribe la temperatura en Fahrenheit: ');
     READLN(f);WRITELN;

     WRITELN('Conversion de temperaturas:');
     WRITELN('     Fahrenheit:',f:5);
     WRITELN('     Celsius:   ',FaC(f):5);

     REPEAT Until Keypressed;
END. {p.p}
```

Realice un programa en Pascal que, mediante una función, calcule el resultado de restar el doble de un numero a su cuadrado.

```
Program ejfun(input,output);
   Uses Crt;
```

```
      VAR a, resultado: Integer;

      FUNCTION calc(x: integer):integer;

        BEGIN {calc}
             calc := sqr(x) - (2*x);
        END; {calc}

BEGIN {p.p}
   ClrScr;

     WRITE('Introduzca un numero: '); READLN(a);
     resultado := calc(a);
     WRITELN;
     WRITELN('Resultado de sqr(x) - (2*x) = ',resultado);
     REPEAT Until Keypressed;
END. {p.p}
```

Hacer un programa que obtenga la distancia entre dos puntos que se encuentran en el plano.

```
Program vector(input,output);
    Uses Crt;
          VAR
             x1,x2,y1,y2: Real;
             resultado: Real;

          FUNCTION distancia(VAR cx1,cx2,Cy1,Cy2: Real): Real;

            BEGIN {distancia}

                distancia := sqrt(sqr(abs(cx2-cx1)) + sqr(abs(Cy2-Cy1)));

            END; {distancia}

BEGIN {p.p}
   ClrScr;

     WRITELN('Introduzca las dos coordenadas de cada punto: '); WRITELN;

     WRITE('  x1: '); READLN(x1);
     WRITE('  y1: '); READLN(y1);
     WRITELN;
     WRITE('  x2: '); READLN(x2);
     WRITE('  Y2: '); READLN(y2);
     WRITELN;

     resultado := distancia(x1,x2,y1,y2);

     WRITE('La distancia entre los dos puntos es de: ');
     Textcolor(9); WRITE(resultado:5:5);

     REPEAT Until Keypressed;
END. {p.p}
```

Escriba un programa PASCAL que calcule el máximo y el mínimo de dos números. Sin embargo,dicho programa debe apoyarse en la subprogramación. Con tal fin, proceda como sigue:
a) Escriba un procedimiento Leer que lea dos números reales.
b) Escriba una función que dados dos números reales, devuelva el máximo de ellos.
c) Escriba una función que dados dos números reales, devuelva el mínimo de ellos.
d) Escriba el programa principal que lea dos números reales, obtenga el mayor y el menor de ellos, y muestre el resultado en pantalla de la siguiente forma. (los números reales justificado a la derecha y con dos dígitos después de la coma):

Primer número --- Segundo número --- Mayor --- Menor
220.59 356.85 356.85 220.59

```
PROGRAM maxmin(input,output);
    Uses Crt;

        VAR x,y:real;

        PROCEDURE leer(VAR a,b:real);
          BEGIN {leer}
                WRITELN('Introduzca dos numeros reales: ');
                WRITE(' 1: '); READLN(a);
                WRITE(' 2: '); READLN(b);
          END; {leer}

        FUNCTION max(a,b:real):real;
          BEGIN {max}
                max := ord(a>b)*a+ord(a<=b)*b
          END; {max}

        FUNCTION min(a,b:real):real;
          BEGIN {min}
                min := -max(-a,-b);
          END; {min}

BEGIN {p.p}
   ClrScr;

      leer(x,y);
      WRITELN('Primer Numero    Segundo Numero    Mayor    Menor');
      WRITELN(x:13:2,y:18:2,max(x,y):9:2,min(x,y):9:2);

   REPEAT Until Keypressed;
END. {p.p}
```

Escriba un programa que pida al usuario introducir un carácter, y le informa si se trata de una vocal o no.
Con tal fin, escriba los siguientes subprogramas:
a) Una función EsMayuscula, que determine si un carácter es mayúscula o no.
b) Una función AMinuscula que, a partir de un carácter c, devuelva el carácter en minúscula con tal de que sea mayúscula. De lo contrario, debe devolver el mismo carácter c.
c) Una función EsVocal que determine si un carácter es una vocal o no.
El programa principal ha de apoyarse en la función EsVocal.

```
PROGRAM mayus(input, output);
   Uses Crt;

    VAR car : char;

    FUNCTION EsMayuscula (c : char) : boolean;
        BEGIN {EsMayuscula}
             EsMayuscula := (c >= 'A') and (c <= 'Z')
        END;  {EsMayuscula}

    FUNCTION AMinuscula (c:char):char;
        CONST offset = ord('a') - ord('A');
        BEGIN {AMinuscula}
            IF esMayuscula(c) THEN
                AMinuscula := chr(ord(c) + offset)
            ELSE
                AMinuscula := c
        END;  {AMinuscula}

    FUNCTION EsVocal(c:char):boolean;
        VAR minus:char;
        BEGIN {EsVocal}
             minus := AMinuscula(c);
             EsVocal := (minus = 'a') OR (minus = 'e') OR (minus = 'i') OR
                        (minus = 'o') OR (minus = 'u')
        END;  {EsVocal}
```

```
BEGIN {p.p}
   ClrScr;

   WRITE('Introduzca un caracter: ');
   READLN(car);
   IF EsVocal(car) THEN
      WRITELN('El caracter ''',car,''' es una vocal')
   ELSE
      writeln('El car cter ''',car,''' NO es una vocal');

   REPEAT Until Keypressed;
END. {p.p}
```

Construye un programa en Pascal que realice la conversión de moneda de dólares de USA a dólares Canadienses y pida la fecha para saber el día que se realizó.

```
PROGRAM ConvertirMoneda1( input, output );
        Uses Crt;

VAR
   MesActual,              {mes actual}
   DiaActual,              {dia actual}
   AnnoActual,             {anno actual}
   CodigoMoneda : integer; {indica el tipo de la moneda a ser convertida}


PROCEDURE MostrarInstrucciones;

   BEGIN
      WRITELN( 'Este programa convierte moneda estadounidense a' );
      WRITELN( 'canadiense y viceversa. Introduzca' );
      WRITELN( '1 para convertir moneda estadounidense a canadiense' );
      WRITE( '2 para convertir moneda canadiense a estadounidense: ' )
   END {MostrarInstrucciones};

PROCEDURE ConvertirUSACanada;

  VAR
     USACanada,      {valor de cambio}
     Dolares : real; {cantidad de dolares estadounidenses a convertir}

  BEGIN
     WRITE( 'Introduzca el valor de cambio actual EE.UU.-Canada: ' );
     READLN( USACanada );
     WRITE( 'Introduzca la cantidad en dolares estadounidenses: ');
     READLN( Dolares );
     WRITELN( 'Es equivalente a ',
              USACanada * Dolares:4:2, ' dolares canadienses.' )
  END {ConvertirUSACanada};


PROCEDURE ConvertirCanadaAUS;

   VAR
     CanadaAUS,      {valor de cambio}
     Dolares : real; {cantidad de dolares canadienses a convertir}

   BEGIN
      WRITELN;
      WRITE( 'Introduzca el valor de cambio actual Canada-EE.UU.: ' );
      READLN( CanadaAUS );
      WRITE( 'Introduzca la cantidad en dolares canadienses: ' );
      READLN( Dolares );
      WRITELN;
      WRITELN( 'Es equivalente a ',
               CanadaAUS * Dolares:4:2, ' dolares estadounidenses.' );
      WRITELN;
  END {ConvertirCanadaAUS};


BEGIN {p.p}
   ClrScr;
```

```
   WRITELN;
   WRITE( 'Introduzca el dia, mes y anno actuales: ' );
   READLN( DiaActual, MesActual, AnnoActual );
   WRITELN;
   MostrarInstrucciones;
   READLN( CodigoMoneda );
   IF CodigoMoneda = 1 THEN
      ConvertirUSACanada
   ELSE
      ConvertirCanadaAUS;
   WRITELN( '*** FECHA DE LA OPERACION: ', DiaActual:1, '-', MesActual:1,
            '-', AnnoActual:1 );

   REPEAT Until Keypressed;
END. {p.p}
```

Mejorar el anterior programa unificando los dos procedimientos en uno solo llamado convertir.

```
PROGRAM ConvertirMoneda2( input, output );
   Uses Crt;

VAR
   MesActual,              {mes actual}
   DiaActual,              {dia actual}
   AnnoActual,             {anno actual}
   CodigoMoneda : integer; {indica el tipo de la moneda a ser convertida}
   ValorCambio,            {el valor de cambio de moneda EE.UU.-Canada}
   Dinero,                 {cantidad monetaria a convertir}
   Total : real;           {total de las cantidades}

PROCEDURE MostrarInstrucciones;

   BEGIN
      WRITELN;
      WRITELN( 'Este programa convierte moneda estadounidense a' );
      WRITELN( 'canadiense y viceversa, y calcula la cantidad total.');
      WRITELN( 'Introduzca 0 para indicar que se han procesado todas',
               'las cantidades.' );
      WRITELN;
      WRITELN( 'Introduzca 1 para convertir moneda estadounidense a ',
               'canadiense' );
      WRITE( '           2 para convertir moneda canadiense a ',
             'estadounidense: ' );
   END {MostrarInstrucciones};

PROCEDURE Convertir( Codigo : integer;   {codigo de moneda}
                     Cambio,             {valor de cambio}
                     Cantidad: real );   {cantidad monetaria}

  VAR
    CantEquiv : real;  {cantidad equivalente en otro sistema monetario}

  BEGIN
     WRITE( 'Es equivalente a ' );
     IF Codigo = 1 THEN
        BEGIN
           CantEquiv := Cambio * Cantidad;
           WRITELN( CantEquiv:4:2, ' dolares canadienses' )
        END {IF}
     ELSE
        BEGIN
           CantEquiv := (1.0 / Cambio) * Cantidad;
           WRITELN( CantEquiv:4:2, ' dolares estadounidenses' )
        END {ELSE}
  END {Convertir};
BEGIN {p.p}
   ClrScr;
   WRITELN;
   WRITE( 'Introduzca el dia, mes y anno actuales: ' );
   READLN( DiaActual, MesActual, AnnoActual );
   MostrarInstrucciones;
   READLN( CodigoMoneda );
```

```
   WRITELN;
   WRITE( 'Introduzca el valor de cambio EE.UU.-Canada: ' );
   READLN( ValorCambio );
   WRITELN;
   Total := 0;

   WRITELN;
   WRITE( 'Introduzca cantidad: ' );
   READLN( Dinero );
   WRITELN;
   WHILE Dinero > 0 DO
      BEGIN
         Convertir( CodigoMoneda, ValorCambio, Dinero );
         Total := Total + Dinero;
         WRITELN;
         WRITE( 'Introduzca cantidad (0 para terminar): ' );
         READLN( Dinero );
         WRITELN;
      END {WHILE};

   WRITELN;
   WRITELN( '*** FECHA DE LA OPERACION: ', DiaActual:1, '-', MesActual:1,
            '-', AnnoActual:1 );
   WRITELN;
   WRITELN( 'La cantidad total convertida es $', Total:4:2 );

   REPEAT Until Keypressed;
END.{p.p}
```

Hacer el mismo programa que los anteriores pero este debe tener parámetros por valor y por variable.

```
PROGRAM ConvertirMoneda3( input, output );
   Uses Crt;

VAR
   MesActual,              {mes actual}
   DiaActual,              {dia actual}
   AnnoActual,             {anno actual}
   MonedaCasa,             {indica el tipo de moneda del lugar}
   CodigoMoneda : integer; {indica el tipo de la moneda a ser convertida}
   ValorCambio,            {el valor de cambio de moneda EE.UU.-Canada}
   Dinero,                 {cantidad monetaria a convertir}
   DineroConv,             {cantidad equivalente en otro sistema monetario}
   Total : real;           {total de las cantidades}

PROCEDURE MostrarInstrucciones;

   BEGIN
      Writeln;
      Writeln( 'Este programa convierte moneda estadounidense a' );
      Writeln( 'canadiense y viceversa, y calcula la cantidad total.');
      Writeln;
      Writeln( 'Introduzca 0 para indicar que se han procesado todas',
               ' las cantidades.' );
      Writeln;
      Writeln;
      Writeln( 'Introduzca 1 para convertir moneda estadounidense a ',
               'canadiense' );
      Write( '           2 para convertir moneda canadiense a ',
             'estadounidense: ' )
   END {MostrarInstrucciones};

PROCEDURE Convertir2(     Codigo : integer;   {tipo de moneda}
                          Cambio,             {valor de cambio}
                          Cantidad : real;    {cantidad a convertir}
                      VAR CantEquiv : real ); {cantidad equivalente}
  BEGIN
     Write( 'Es equivalente a ' );
     IF Codigo = 1 THEN
        BEGIN
           CantEquiv := Cambio * Cantidad;
           Writeln;
           Writeln( CantEquiv:4:2, ' dolares canadienses' )
        END {IF}
```

```
     ELSE
        BEGIN
           CantEquiv := (1.0 / Cambio) * Cantidad;
           Writeln;
           Writeln( CantEquiv:4:2, ' dolares estadounidenses' )
        END {ELSE}
  END {Convertir2};

BEGIN {p.p}
   ClrScr;

   Writeln;
   Writeln( 'Introduzca el dia, mes y anno actuales: ' );
   Write('Dia: '); Readln(DiaActual);
   Write('Mes: '); Readln(MesActual);
   Write('Año: '); Readln(AnnoActual);
   MostrarInstrucciones;
   Readln( MonedaCasa );
   Write( 'Introduzca el valor de cambio EE.UU.-Canada: ' );
   Readln( ValorCambio );
   Total := 0;

   Write( 'Introduzca tipo de moneda y cantidad (0 0 para terminar): ' );
   Readln( CodigoMoneda, Dinero );
   WHILE Dinero > 0 DO
      BEGIN
         IF CodigoMoneda <> MonedaCasa THEN
            BEGIN
               Convertir2( CodigoMoneda, ValorCambio, Dinero,
                           DineroConv );
               Total := Total + DineroConv
            END {IF}
         ELSE
            Total := Total + Dinero;
         Write( 'Introduzca tipo de moneda y cantidad ',
                '(0 0 para terminar):  ' );
         Readln( CodigoMoneda, Dinero )
      END {WHILE};

   Writeln;
   Writeln( '*** FECHA DE LA OPERACION: ', DiaActual:1, '-', MesActual:1,
            '-', AnnoActual:1 );
   Writeln( 'La cantidad total convertida es $', Total:4:2 )
END {p.p}.
```

Modificar el programa ConvertirMoneda3 de forma que la fecha de operación introducida por el usuario sea visualizada de la forma dd/mm/aa.

```
PROGRAM ConvertirMoneda4( input, output );
   Uses Crt;

VAR
   MesActual,              {mes actual}
   DiaActual,              {dia actual}
   AnnoActual,             {anno actual}
   MonedaCasa,             {indica el tipo de moneda del lugar}
   CodigoMoneda : integer; {indica el tipo de la moneda a ser convertida}
   ValorCambio,            {el valor de cambio de moneda EE.UU.-Canada}
   Dinero,                 {cantidad monetaria a convertir}
   DineroConv,             {cantidad equivalente en otro sistema
                            monetario}
   Total : real;           {total de las cantidades}

PROCEDURE MostrarInstrucciones;

   BEGIN
      Writeln;
      Writeln( 'Este programa convierte moneda estadounidense a' );
      Writeln( 'canadiense y viceversa, y calcula la cantidad total.');
      Writeln( 'Introduzca 0 para indicar que se han procesado todas',
               'las cantidades.' );
      Writeln;
      Writeln;
      Writeln( 'Introduzca 1 para convertir moneda estadounidense a ',
```

```
               'canadiense' );
      Write( '           2 para convertir moneda canadiense a ',
             'estadounidense: ' )
   END {MostrarInstrucciones};

PROCEDURE Convertir2(     Codigo : integer;   {tipo de moneda}
                          Cambio,             {valor de cambio}
                          Cantidad : real;    {cantidad a convertir}
                      VAR CantEquiv : real ); {cantidad equivalente}

  BEGIN
     Writeln;
     Write( 'Es equivalente a ' );
     IF Codigo = 1 THEN
        BEGIN
           CantEquiv := Cambio * Cantidad;
           Writeln( CantEquiv:4:2, ' dolares canadienses' )
        END {IF}
     ELSE
        BEGIN
           CantEquiv := (1.0 / Cambio) * Cantidad;
           Writeln( CantEquiv:4:2, ' dolares estadounidenses' )
        END {ELSE}
  END {Convertir2};


PROCEDURE MostrarFecha( Dia,               {el dia}
                        Mes,               {el mes}
                        Anno : integer );  {el año}


   BEGIN
      IF Dia < 10 THEN
         Write( '0' );
      Write( Dia:1, '/' );
      IF Mes < 10 THEN
         Write( '0' );
      Write( Mes:1, '/' );
      Anno := Anno MOD 100;
      IF Anno < 10 THEN
         Write( '0' );
      Writeln( Anno:1 )
   END {MostrarFecha};

BEGIN {p.p}
   ClrScr;

   Writeln( 'Introduzca el dia, mes y anno actuales: ' );
   Write('Dia: '); Readln(DiaActual);
   Write('Mes: '); Readln(MesActual);
   Write('Año: '); Readln(AnnoActual);
   MostrarInstrucciones;
   Readln( MonedaCasa );
   Write( 'Introduzca el valor de cambio EE.UU.-Canada: ' );
   Readln( ValorCambio );
   Total := 0;

   Writeln;
   Write( 'Introduzca tipo de moneda y cantidad (0 0 para terminar): ' );
   Readln( CodigoMoneda, Dinero );
   WHILE Dinero > 0 DO
      BEGIN
         IF CodigoMoneda <> MonedaCasa THEN
            BEGIN
               Convertir2( CodigoMoneda, ValorCambio, Dinero,
                           DineroConv );
               Total := Total + DineroConv
            END {IF}
         ELSE
            Total := Total + Dinero;
         Write( 'Introduzca tipo de moneda y cantidad ',
                '(0 0 para terminar):  ' );
         Readln( CodigoMoneda, Dinero )
      END {WHILE};

   Writeln;
```

```
   Writeln( '*** FECHA DE LA OPERACION: ' );
   MostrarFecha( DiaActual, MesActual, AnnoActual );
   Writeln( 'La cantidad total convertida es $', Total:4:2 )
END {p.p}.
```

Un programilla simple para amenizar el rato y hacerlo más llevadero.
Escriba un programa PASCAL que produzca las siguiente salida:

```
                         *
                        ***
                       *****
                      *******
                     *********
                    ***********
                   *************
                         *
```

```
PROGRAM arbol(output);
   Uses Crt;

BEGIN
   ClrScr;

     Textcolor(2);
     WRITELN;
     WRITELN('*':9);
     WRITELN('***':10);
     WRITELN('*****':11);
     WRITELN('*******':12);
     WRITELN('*********':13);
     WRITELN('***********':14);
     WRITELN('*************':15);
     WRITELN('*':9);

     REPEAT Until Keypressed;
END.
```

Antes de nada tengo que agradecer a todas las personas que poseeis mis ejercicios por el gran éxito que están teniendo. El día 21 de noviembre de 2001 batí el record de visitas diarias en mi página. GRACIAS A TODOS, intentaré corresponderos en la medida que mis estudios me dejen.

Empezamos con la siguiente parte:

Vamos a hacer un programa que opere con datos de tipo color y muestre un mensaje dependiendo el que se haya seleccionado.

```
Program colores (Output);
   Uses Crt;

        Type
               color = (verde, amarillo, azul, rojo, negro);
        Var
                i: color;
Begin
   ClrScr;

        i := azul; {Sobre este tipo de dato podemos operar con
                    funciones que admiten argumentos ordinales,
                    como succ y pred.
                    Hay que tener en cuenta que este tipo de dato
                    no se puede leer ni escribir. Un ejemplo es
                    que no podria hacerse: Readln(i).
                    En cambio, si podriamos hacer: i := color(3)}

        {Aumentamos la i en 1, es decir, pasamos al siguiente color}
        Case succ(i) Of
                azul: Writeln('Es el color azul, mi preferido.');
                rojo: Writeln('Rojo es el color elegido.')
```

```
        End;
End.
```

Ahora realizaremos un programa que nos diga la estación en la que nos encontramos, utilizando conjuntos y tipos.

```
Program estaciones;
   Uses Crt;

   Type meses = (enero,febrero,marzo,abril,mayo,junio,julio,
                 agosto,septiembre,octubre,noviembre,diciembre);

        estacion = Set Of meses;

   Var
     primavera,verano,otonno,invierno,cambio_est: estacion;
     mes: meses;
     i: Integer;

Begin
   ClrScr;

   {Hacemos los conjuntos de meses que forman cada estacion}
   primavera := [marzo..junio];
   verano := [junio..septiembre];
   otonno := [septiembre..diciembre];
   invierno := [diciembre,enero..marzo];

   (*
   Entre conjuntos los operadores se comportan de la siguiente forma:
    A := B + C      {A es el conjunto de la union de B y C}
    A := B * C      {A es el conjunto de la interseccion de B y C}
    A := B - C      {A es el conjunto de la diferencia de B y C}

    Tambien vamos a poder comparar conjuntos con los operadores logicos:
     cierto := A = B      {True si A y B son iguales}
     cierto := A <> B     {True si A y B son distintos}
     cierto := A <= B     {True si A es un subconjunto de B}
     cierto := A => B     {True si B es un subconjunto de A}
    *)

    {Creamos un conjunto con los meses en los que cambian las estaciones}
    cambio_est := primavera * verano + verano * otonno + otonno * invierno + invierno *
primavera;

    Repeat
       Write('Escriba un mes para decirle en que estacion se encuentra: ');
       Read(i)
    Until (i >= 1) and (i <= 12);

    mes := Meses(i - 1); {Como los conjuntos empiezan por 0, el termino
                          al que nos referimos es uno menor}

    If mes in cambio_est Then Write('En este mes cambiamos de estacion.')
    Else
       Begin
        If mes in primavera  Then Write('¡Estamos en primavera!');
        If mes in verano     Then Write('¡Ya ha llegado el verano!');
        If mes in otonno     Then Write('Este mes pertenece al otoño.');
        If mes in invierno   Then Write('Invierno, ¡A disfrutar de la nieve!')
       End;
End.
```

Vamos a realizar un programa que escriba hola dentro de un archivo

```
Program hola1;
{Este programa va a escribir "HOLA" dentro de un archivo llamado "hola.txt"}
   Uses Crt;

      Var
         prueba: File of Byte;
         num1,num2,num3,num4: Byte;
```

```
Begin {p.p}
   ClrScr;

     num1 := 72; num2 := 79;
     num3 := 76; num4 := 65;

     Assign(prueba,'hola.txt');          {Asignamos el nombre del archivo}
     Rewrite(prueba);                    {Abrimos el archivo}
     Write(prueba,num1,num2,num3,num4); {Escribimos hola en el archivo}
     Close(prueba)                       {Cerramos el archivo}
End. {p.p}
```

Ahora vemos otra versión de cómo escribir en un archivo.

```
Program hola2;
{Este programa va a escribir "Hola Mundo" dentro de un archivo
 llamado "hola2.txt"}

   Uses Crt;

      Var
         prueba: Text;

Begin {p.p}
   ClrScr;

     Assign(prueba,'hola2.txt'); {Asignamos el nombre del archivo}
     Rewrite(prueba);            {Abrimos el archivo}
     Write(prueba,'Hola Mundo'); {Escribimos hola en el archivo}
     Close(prueba)               {Cerramos el archivo}

End. {p.p}
```

Realiza un programa que lea el archivo creado en el ejercicio hola1 y muestre por pantalla su contenido.

```
Program hola2;
{Usamos el archivo creado en el ejercicio hola1 como entrada
 de datos y mostramos por pantalla su contenido}

   Uses Crt;

        Var
           prueba: File of Char;
           num1,num2,num3,num4: Char;

Begin {p.p}
   ClrScr;

   Assign(prueba,'hola.txt');
   Reset(prueba);
   Read(prueba,num1,num2,num3,num4);
   Close(prueba);
   Writeln(num1,num2,num3,num4);

End. {p.p}
```

Escribir un programa en Pascal que escriba la lista de caracteres ASCII dentro de un archivo de texto.

```
Program caracteresASCII;
 {Este programa va a escribir la lista de caracteres ASCII en
  un archivo con extension txt}

        Var
```

```
          ascii: Text;        {Archivo de texto}
          caracter: Char;
          i: Byte;

Begin {p.p}

      Assign(ascii,'ascii.txt');    {Damos el nombre al archivo}
      Rewrite(ascii);               {Abrimos un archivo nuevo}

      {Escribimos la lista de caracteres ASCII}
      For i := 33 To 255 Do
         Begin
           caracter := chr(ord(i));
           Writeln(ascii,caracter);
         End;

      {Por ultimo, escribimos unos mensajes dentro del archivo.
       Si quereis, probar a poner vuestros mensajes con el formato
       de salida que deseeis}

      Writeln(ascii,'');
      Writeln(ascii,'Victor Sanchez Sanchez');
      Writeln(ascii,'http://usarios.tripod.es/VictorSanchez2');
      Writeln(ascii,'');
      Writeln(ascii,'Hasta pronto, pasa un buen dia');

      Close(ascii);  {Cerramos el archivo}

      {Del 0 al 32 están los caracteres como el tabulador horizontal,
       el salto de linea, el salto de carro, el espacio... y bastantes
       numeros sin caracter ASCII asociado. Posiblemente aparezcan
       cuadraditos en blanco, esto se debe a que existen espacios en los
       que no hay asignados caracteres}

End. {p.p}
```

Ahora vamos a comenzar con las prácticas realizo en mi universidad (Universidad Autónoma de Madrid – España).
Si hay algún error o alguna forma mejor de hacerlo, por favor decírmelo para poder corregirlo:

P1. Variables
Escribir un programa en el que se declare una variable por cada uno de los siguientes tipos de datos: Integer, LongInt, ShortInt, Byte, Word, String, Char, Real, Double, Boolean.

```
Program P1;

        Var
           a: ShortInt;
           b: String;
           c: Real;
           d: Boolean;
           e: Char;
           f: Integer;
           g: LongInt;
           h: Double;
           i: Word;
           j: Byte;

Begin {p.p}

      a := -90;
      b := '9.0';
      c := 9e30;
      d := true;
      e := '9';
      f := 32700;
      g := 70000;
      h := 9e40;
      i := 65500;
      j := 90;
```

```
End. {p.p}
```

P2. Constantes

Repitan el ejercicio anterior definiendo (y asignandoles valor en la declaración) los valores anteriores como constantes.

```
Program P2; {p.p}

        Const
           a = 32700;
           b = -90;
           c = 90;
           d = 65500;
           e = '9.0';
           f = '9';
           g = 9e30;
           h = 9e40;
           i = True;
           j = 70000;

Begin {p.p}

{Programa que asigna los valores anteriores como constantes}

End. {p.p}
```

P3. Constantes con tipo.

Repetir el ejercicio anterior definiendo (y asignandoles valor en la declaración) los valores anteriores como constantes con tipo.

```
Program P3; {p.p}

        Const
           a: ShortInt = -90;
           b: String = '9.0';
           c: Real = 9e30;
           d: Boolean = true;
           e: Char = '9';
           f: Integer = 32700;
           g: LongInt = 70000;
           h: Double = 9e40;
           i: Word = 65500;
           j: Byte = 90;

Begin {p.p}

{Programa que asigna los valores anteriores como constantes con tipo}

End. {p.p}
```

P4. Búsqueda de errores.

Encuentren todos los errores (tanto de compilación como de ejecución) del siguiente programa, señalen la causa del error. Tecleen y ejecuten paso a paso si es necesario. Se debe entregar el programa tal cual indicando, dentro de comentarios Pascal, los errores que se han encontrado junto con el motivo que los produce.

```
Program Errores (OUTPUT);
 Const
      vPrec = 19;
 Var
    A, B, E : Integer;
    C, D : Shortint;
    F : Real;
```

```
 Begin
    vPrec := vPrec + 1; {No podemos sumarle 1 porque es una constante deberia
                         ser una constante con tipo para poder hacerlo}

    B := ( A + 2 ) * 2; {Deberiamos haber inicializado A anteriormente para
                         que el programa funcione con normalidad}
    writeln('El valor de B es ',B);
    D := vPrec*6;
    C := D * 100 + 18; {No podemos asignarle a C un valor mayor a su rango,
                        deberiamos declarar C como Integer}
    E := F + 1; {A una variable de tipo entero no se le puede asignar
                 un valor real. Para hacerlo E deberia ser de tipo Real}
    Writeln('El valor de E es ', E:3:3); {No podemos pedir que nos escriba
                                          E con decimales porque es un entero
                                          deberiamos haberlo declarado como
                                          un Real}
End.
```

P.5 Correción de programas

Arreglar el programa P4 para que el resultado que aparece en la pantalla sea:

El valor de B es 6
El valor de E es 12032.333

```
Program Errores (Output);
 Const
      vPrec: Integer = 19; {Es necesario definir vPrec como constante con
                            tipo para que posteriormente podamos sumarle 1}
 Var
    A, B, C : Integer;
    D : Shortint;
    F, E : Real;
    {Asignamos los tipos de datos adecuados para el correcto funcionamiento
     del programa. E lo declaramos como Real y C como Integer}

Begin {p.p}
    vPrec := vPrec + 1;
    A := 1; {Inicialiamos a con valor 1 para que B valga 6}
    B := ( A + 2 ) * 2;
    Writeln('El valor de B es ',B);
    D := vPrec*6;
    C := D * 100 + 18; {El valor de C superaba el rango, por eso se ha
                        declarado C como Integer}
    F := 2 * vPrec / 3 + C;
    E := F + 1; {E pasa a ser de tipo Real para obtener el resultado deseado}
    Writeln('El valor de E es ', E:5:3); {Ponemos 5:3 para controlar que
                                          en la salida haya 5 espacios, mas
                                          otros 3 para la parte decimal}
End. {p.p}
```

P6. Escribir un programa que pida al usuario el nombre de un artículo, su precio en Euros, la cantidad de artículos que se desean y el descuento a aplicar (en tanto por ciento, por unidad) sobre el precio inicial. Con esos datos obtener el total, haciendo el descuento. Presentar el PVP final de los productos solicitados por pantalla, en pesetas y Euros. Elijan los tipos adecuados para cada variable del programa.

```
Program P6 (INPUT,OUTPUT);

        Const
              Euro = 166.386;

        Var
              articulo  : String[40]; {Nombre del articulo}
              num_arti  : Byte;       {Numero de articulos que nos indican}
              precio    : Real;       {Precio de una unidad}
              precio_t  : Real;       {Precio Total}
```

```
            descuento : Real;       {Descuento a aplicar}
            total     : Real;       {Precio con el descuento aplicado}

Begin {p.p}

          Write('Introduzca el nombre del articulo: ':50);
          Readln(articulo);
          Writeln('');
          Write('Introduzca su precio en Euros:     ':50);
          Readln(precio);
          Writeln('');
          Write('¿Cuantos articulos desea?:         ':50);
          Readln(num_arti);
          Writeln('');
          Write('Descuento a aplicar en %:          ':50);
          Readln(descuento);
          Writeln('');

          precio_t := precio * num_arti; {obtenemos el precio total}
          descuento := (precio_t * descuento) / 100; {total del descuento}
          total := precio_t - descuento;

          Writeln('El valor total de la compra es de: ':50);
          Writeln('');
          Writeln(total:35:3,' Euros.');

          total := total * Euro; {Pasamos a pts el precio total}

          Write(total:35:3,' Pts.');

End. {p.p}
```

P7. E/S por teclado y pantalla.

Escriban un programa que lea tres números enteros introducidos por teclado en la misma línea. El programa aceptará a continuación tres nuevos números enteros escritos en líneas sucesivas, y escribirá por pantalla cuatro líneas con los siguientes datos:

1. *Los seis números introducidos, separados por blancos.*
2. *La suma de los seis números.*
3. *El resultado obtenido al dividir la suma de los tres primeros números por la suma de los tres segundos.*
4. *El cociente entero y el resto obtenidos al dividir la suma de los tres primeros números por la suma de los tres segundos.*

```
Program P7 (INPUT, OUTPUT);

        VAR
           num1, num2, num3: Integer;        {Tres primeros numeros que pedimos}
           num4, num5, num6: Integer;        {Tres ultimos numeros que pedimos}
           suma1, suma2, sumatotal: Integer; {Guardan las sumas que realizamos}
           div_ent, resto: Integer;          {División entera y su resto}
           div_real: Real;                   {División real}

Begin {p.p}

      Write('Introduzca 3 numeros enteros: ');
      Readln(num1,num2,num3); {Leemos los tres primeros datos}

      Writeln('');
      Writeln('Introduzca otros 3 numeros enteros: ');
      Write('Numero 4: '); Readln(num4);
      Write('Numero 5: '); Readln(num5);
      Write('Numero 6: '); Readln(num6);
      {Leemos los tres datos siguientes}

      Writeln('');
      Write('Los numeros introducidos son: ');
      Write(num1,' ',num2,' ',num3,' ',num4,' ',num5,' ',num6);
```

```
      {Sacamos por pantalla los numeros introducidos}

      Writeln('');
      suma1 := (num1 + num2 + num3);  {sumamos los 3 primeros numeros}
      suma2 := (num4 + num5 + num6);  {sumamos los 3 ultimos numeros}

      sumatotal := suma1 + suma2;     {hacemos la suma de todos los numeros}
      div_real  := suma1 / suma2;     {guardamos la división real}
      div_ent   := suma1 div suma2;   {obtenemos la división entera}
      resto     := suma1 mod suma2;   {calculamos el resto de la división}

      Writeln('La suma de los 6 numeros es: ',sumatotal);
      {Saca por pantalla el resultado de la suma de todos los numeros}

      Writeln('');
      Writeln('La division de los 3 primeros numeros entre los 3 ultimos es:');

      Writeln('');
      Writeln(' División real:                    ':50,div_real:6:2);
      Writeln(' División entera:                  ':50,div_ent:6);
      Writeln(' El resto de la división entera es: ':50,resto:6);
      {Mostramos por pantalla el resultado de las divisiones y el resto}

End. {p.p}
```

P8. Redireccionamiento de la E/S.

Cualquier programa E/S estándar, como el P7, escribe por defecto en la pantalla y lee del teclado. En DOS, la lectura de un programa puede ser redireccionada desde un archivo utilizando el carácter < . Si el nombre del fichero ejecutable del programa es P8.exe, con el comando

P8 < entrada.dat

se ejecutará el programa P8 tomando los datos de entrada.dat

Escriban un programa que lea los datos de un archivo y los esciba en otro. Los datos que ha de leer son los siguientes, y se encontrarán en el archivo con el siguiente formato:

Esta línea se tiene que con
catenar con esta otra.
La salida va a ocu
par sólo dos líneas.

{Estas son las lineas que debemos grabas en nuestro archivo. El archivo debe tener extension .txt o .dat}

El programa escribirá cada frase en una única línea:

Esta línea se tiene que concatenar con esta otra.
La salida va a ocupar sólo dos líneas.

```
Program P8 (INPUT,OUTPUT);

        VAR
          linea1, linea2, linea3, linea4: String[30];
          {Declaramos las 4 lineas que encontraremos
           en el archivo que vamos a introducir}

Begin {p.p}

          {Vamos a leer cada linea del archivo. Lo almacenamos en las variables}
```

```
          Readln(linea1);
          Readln(linea2);
          Readln(linea3);
          Readln(linea4);

          Writeln(linea1,linea2); {Escribimos primera y segunda linea}
          Writeln(linea3,linea4); {Unimos la tercera y cuarta linea}
          {Así obtenemos únicamente las 2 líneas que se necesitan}

          {Para el correcto funcionamiento de este programa se nos pide leer
           los datos de un archivo y pasarlo a otro que ocupe sólo 2 líneas.
           Utilizando DOS lograremos obtener los datos de entrada y de salida.
           Ej: 3C12P8.EXE < TEX_E.TXT > TEX_S.TXT}

End. {p.p}
```

P9. Salida con formato

Escriban un programa que lea del teclado el valor del lado de un cuadrado, y calcule los siguientes valores:

el área del cuadrado
el área del círculo inscrito (radio = semilado)
el área del círculo circunscrito (radio = semidiagonal);
el perímetro del círculo circunscrito

La salida del programa, para un valor de entrada igual a 2.0, será:

Area del cuadrado = 4.00
Area del círculo inscrito = 3.14
Area del círculo circunscrito = 6.28
Perímetro del círculo circunscrito = 8.99

```
Program P9 (INPUT,OUTPUT);

        CONST
             PI= 3.14159; {Declaramos como constante el valor de PI}

        VAR
           ladocuad: Real;  {Lado del cuadrado}
           a_cuad: Real;    {Area del cuadrado}
           a_cins: Real;    {Area del círculo inscrito}
           a_ccir: Real;    {Area del círculo circunscrito}
           p_ccir: Real;    {Perímetro del círculo circunscrito}

Begin {p.p}

      Write('Introduzca el lado del cuadrado: ');
      Readln(ladocuad); {Leemos del teclado el valor del lado del cuadrado}

      a_cuad := sqr(ladocuad); {Lado al cuadrado}
      a_cins := PI * (sqr(ladocuad / 2));  {PI * radio al cuadrado}
      a_ccir := PI * (sqr(sqrt(sqr(ladocuad) + sqr(ladocuad)) / 2));
      {PI * radio al cuadrado. Radio =  mitad de la diagonal del cuadrado}
      p_ccir := 2 * PI * (sqrt(sqr(ladocuad) + sqr(ladocuad)) / 2);
      {2 * PI * radio. El radio es la mitad de la diagonal del cuadrado}

      Writeln('');
      Writeln('Area del cuadrado =                 ',a_cuad:5:2);
      Writeln('Area del círculo inscrito =         ',a_cins:5:2);
      Writeln('Area del círculo circunscrito =     ',a_ccir:5:2);
      Writeln('Perímetro del círculo circunscrito = ',p_ccir:5:2);
      {Mostramos por pantalla el cálculo de las áreas y el perímetro}

End. {p.p}
```

P.10 Operaciones aritméticas

Escriban un programa que lea dos números reales positivos y devuelva la potencia entera a la que hay que elevar el primero para que de el número más próximo posible al segundo.

```
Program P10 (INPUT,OUTPUT);

        VAR
            num1, num2: Real;  {N£meros con los que calcularemos la potencia}
            n: Real;           {Potencia}

Begin {p.p}

      Write('Introduzca dos numeros reales positivos: ');
      Readln(num1,num2); {Leemos los 2 numeros reales}

      n := ln(num2) / ln(num1); {Calculamos la potencia tomando logaritmos}

      n := Round(n);
      {Redondeamos la potencia y obtenemos la potencia entera m s cercana a
      la que hay que elevar el primer numero para obtener el segundo}

      Write('La potencia entera a la que hay que elevar el numero 1 es: ');
      Write(n:3:0); {Sacamos por pantalla el valor de la potencia entera}

End. {p.p}
```

P11. Números pseudoaleatorios

Escriban un programa que genere un número entero aleatorio entre 1800 y 20000, utilizando la función Random (¡y NO Random(x)!). Considerando, además, que dicho número representa un lapso de tiempo en segundos, dar el resultado en forma de horas, minutos y segundos (formato HH:MM:SS).

```
Program P11 (OUTPUT);

      VAR
         aleatorio: Integer;
         {Va a ser el n£mero donde guardaremos el valor.
          El rango va a estar entre 1800 y 20000}
         horas, minutos, segundos: Integer; {Van a ser las horas, minutos
                                             y segundos a los que convertimos
                                             el numero aleatorio}

Begin {p.p}

      Randomize; {Inicializamos Random}

      aleatorio := 20000 - (Round(Random * 18200));
      {Como aleatorio debe estar entre 1800 y 20000, a 20000 le restamos
       un numero generado entre 0 y 18200, con lo cual, vamos a obtener
       el rango deseado:
       Si Random = 0, el valor ser  m ximo = 20000
       Si Random = 1, el valor ser  m¡nimo = 1800}

      Writeln(''); {Dejamos una linea en blanco}
      Writeln('Generamos un numero aleatorio: ',aleatorio);
      {Mostramos el valor por pantalla}

      horas    := aleatorio div 3600; {Vemos las horas que hay en total}
      segundos := aleatorio mod 3600; {El resto lo guardamos como segundos}
      minutos  := segundos div 60;    {Sacamos los minutos totales}
      segundos := segundos mod 60;    {El resto van a ser los segundos}

      Writeln('');
      Write(aleatorio,' segundos son: ');
      {Sacamos un mensaje mostrando el numero de segundos a convertir}

      Write(horas,':',minutos,':',segundos);
      {Sacamos por pantalla los datos en el formato deseado}
```

```
End. {p.p}
```

P12. Operacoines lógicas (I)

Escriban un programa que suministrado un día, mes y año calcule el siguiente día, mes y año. Tanto el formato de entrada como de salida será: dd mm aaaa.

Nota: Un año es bisiesto si es divisible por 4, pero no por 100. Una excepción son los años divisibles por 400, que son todos bisiestos.

```
Program P12 (INPUT,OUTPUT);
   Uses Crt;

        VAR
           dia,mes: Byte; {Dia y mes que se va a introducir}
           anno: Word;    {A¤o que introduce el usuario}
           esbisiesto,nobisiesto,esdiciembre,es30,es31: Boolean;
           {variables para chequear en que grupo de meses se encuentra}

Begin {p.p}
   ClrScr; {limpiamos la pantalla}

   {Pedimos al usuario los datos}
   Repeat
     Write('Introduzca la fecha que desee (dd mm aaaa): ');
     Read(dia,mes,anno);
   Until (dia in [1..31]) and (mes in [1..12]) and (anno > 0);
   {Solo continuamos cuando dia mes y anno sean correctos}

   Case mes Of

     2: {El mes es febrero}
       Begin
         If (anno mod 400 = 0) or ((anno mod 4 = 0) and (anno mod 100 <> 0)) Then
             esbisiesto := True {Si cumple los requisitos es bisiesto}
         Else
            Begin
             If dia > 28 Then
                Begin
                  Repeat
                     Write('Introduzca un dia que corresponda al mes: ');
                     Read(dia)
                  Until (dia >= 1) and (dia <= 30);
                End; {Pedimos un dia adecuado para es mes introducido}

              nobisiesto := True {El mes es febrero, pero no es bisiesto}
            End;
       End;

    12: {El mes es diciembre}
       esdiciembre := True;

    4,6,9,11: {El mes tiene 30 dias (abril, junio, septiembre o noviembre)}
       Begin
         If dia > 30 then {Estos meses no pueden tener 31 dias}
            Begin
               Repeat
                  Write('Introduzca un dia que corresponda al mes: ');
                  Read(dia)
               Until (dia >= 1) and (dia <= 30);
            End; {Pedimos un dia adecuado para es mes introducido}

            es30 := True {Esta en el grupo de los meses que tienen 30 dias}
       End
    Else {Esta en el grupo de los meses que tienen 31 dias}
       es31 := True
   End; {fin Case}

   dia := dia + 1; {Sumamos un dia a la fecha actual}

   {Hacemos los cambios oportunos para respetar el ciclo logico de las fechas}
   If esbisiesto and (dia > 29) Then
   {Si estamos en febrero, es bisiesto y la fecha pasa del 29...}
```

```
      Begin
           dia := 1;
           mes := 3  {Debe ser el 1 de marzo}
      End

   Else If nobisiesto and (dia > 28) Then
   {Si estamos en febrero, no es bisiesto y la fecha pasa del 28...}
      Begin
           dia := 1;
           mes := 3  {Debe ser el 1 de marzo}
      End

   Else If esdiciembre and (dia > 31) Then
   {Si es diciembre y el dia pasa del 31, debe ser el 1 de enero}
      Begin
           dia := 1;
           mes := 1;
           anno := anno + 1 {Debe ser 1 de enero y un a¤o mayor}
      End

   Else If es30 and (dia > 30) Then
   {Es un mes de 30 dias, si el dia pasa de 30, sera 1 del siguiente mes}
      Begin
           dia := 1;
           mes := mes + 1
      End

   Else If es31 and (dia > 31) Then
   {Es un mes de 31 dias, si el dia pasa de 31, sera 1 del mes siguiente}
      Begin
           dia := 1;
           mes := mes + 1
      End;

   Writeln;
   Write('La fecha introducida mas un dia es: ':50,dia,' ',mes,' ',anno);
   {Escribimos por pantalla la fecha final, a la cual le hemos sumado un dia}

End. {p.p}
```

P13. Operacoines lógicas (II)

Un rectángulo queda determinado por las coordenadas de dos vértices opuestos. Un círculo queda determinado por las coordenadas de su centro y su radio. Escriban un programa que, teniendo como datos de entrada las coordenadas del par de vértices opuestos de rectángulo, del centro del círculo y de su radio, y las coordenadas del punto problema, proporcione como salida el texto adecuado, seleccionado entre las siguientes posibilidades:

· Punto exterior al círculo y al rectángulo.
· Punto interior al rectángulo.
· Punto interior al círculo.
· Punto interior al círculo y al rectángulo.

```
Program P13 (INPUT,OUTPUT);
   Uses Crt;

        VAR
           x,y: Real;                          {Coordenadas del punto}
           x1_rec,y1_rec,x2_rec,y2_rec: Real; {Datos del rectangulo: x,y del
                                               vertice 1 y el vertice 2}
           centrox,centroy,radio: Real;       {Datos del circulo}
           distancia: Real;                   {Distancia entre el centro del
                                               circulo y el punto introducido
                                               por el usuario}
           aux: Real;                         {Variable auxiliar}

Begin {p.p}
   ClrScr;

   Writeln(' Por favor, introduzca los valores necesarios: ');
   {Pedimos los datos necesarios para realizar el programa}
```

```
   Writeln;
   Write(' Coordenadas x,y del centro del circulo: ');
   Read(centrox,centroy); {Coordenadas del centro del circulo}

   {Pedimos al usuario un radio positivo del circulo}
   Repeat
     Write(' Radio del circulo: ');
     Read(radio);
     If radio <= 0 Then
        Writeln(' El radio debe ser positivo.');
   Until radio > 0;

   Repeat
      Writeln(' Introduzca unas coordenadas x,y de vertices opuestos del rectangulo: ');
      Write(' Vertice 1: '); Read(x1_rec,y1_rec);
      Write(' Vertice 2: '); Read(x2_rec,y2_rec);
      {Con las coordenadas de los 2 vertices conoceremos el area del rectangulo}

      If (x1_rec = x2_rec) Or (y1_rec = y2_rec) Then
         Writeln(' Por favor, introduzca unos datos correctos: ');
   Until (x1_rec <> x2_rec) And (y1_rec <> y2_rec);

   Write(' Introduzca las coordenadas x,y del punto: ');
   Read(x,y); {Coordenadas del punto que introduce el usuario}
   Writeln;

   distancia := sqrt(sqr(x - centrox) + sqr(y - centroy));
   {Distancia entre el punto introducido y el centro del circulo.
    Si la distancia es mayor que el radio estara fuera del circulo}
   If distancia <= radio Then
       Writeln(' El punto esta dentro del circulo.    ')
   Else
       Writeln(' El punto no esta dentro del circulo. ');

   Writeln;

   {Tenemos que chequear los posibles casos que se pueden dar en
    los 4 cuadrantes, y que el orden en que se pueden escribir
    los vertices puede ser distinto}
   If (x1_rec > x2_rec) and (y1_rec > y2_rec) Then
      Begin
        aux := x2_rec;
        x2_rec := x1_rec;
        x1_rec := aux;

        aux := y2_rec;
        y2_rec := y1_rec;
        y1_rec := aux
      End
   Else If (x1_rec < x2_rec) and (y1_rec > y2_rec) Then
      Begin
        aux := y2_rec;
        y2_rec := y1_rec;
        y1_rec := aux
      End
   Else If (x1_rec > x2_rec) and (y1_rec < y2_rec) Then
      Begin
        aux := x2_rec;
        x2_rec := x1_rec;
        x1_rec := aux;
      End;

   {Vemos si el punto esta dentro del area del rectangulo}
   If (x >= x1_rec) and (x <= x2_rec) and (y >= y1_rec) and (y <= y2_rec) Then
       Write(' El punto esta dentro del rectangulo.   ')
   Else
       Write(' El punto no esta dentro del rectangulo.')
End. {p.p}
```

P14. Operaciones de manipulación de bits

Escriban un programa que dado un número entero en base decimal, calcule e imprima su equivalente en hexadecimal. El número entero estará en el rango 0..32767. Usar las operaciones de desplazamiento SHL y SHR.

```
Program P14 (INPUT,OUTPUT);
        Uses Crt;

        Const ascii_A = ord('A'); {Valor ASCII de la letra 'A'}

        Var
           decimal,hexadecimal,aux: Integer;{Las usamos para manipular los bits}
           hex_letra: Char;                 {Valor de la letra en hexadecimal}
           izq: Byte;                       {Bits desplazados a la izquierda}
           i,j: Byte;                       {Variables para recorrer los bucles}

Begin {p.p}
  ClrScr;

  {Pedimos al usuario un entero que este en el rango deseado}
  Repeat
        Write('Introduzca un numero entero entre 0 y 32767: ');
        Read(decimal);
  Until ((decimal >= 0) and (decimal <= 32767));

  Writeln;
  Write('El numero en hexadecimal es: ');
  {Preparamos el formato de salida para mostrarlo adecuadamente}

  izq := 0; {Inicializamos las variables que vamos a utilizar}

  Repeat
      j := 0; {Inicializamos j para que el valor ASCII de la primera letra
               sea el de 'A' al comenzar cada cuarteto}

      {Con estas dos lineas vamos a mover a izquierda y derecha los bits
       para poder pasarlo a hexadecimal. Con izq cambiaremos de cuarteto}
      aux := decimal Shl izq;
      hexadecimal := aux Shr 12;

      {Si el valor del cuarteto es mayor o igual a 10 se escribe con letra}
      If hexadecimal >= 10 Then
        Begin
          For i := 10 To 15 Do {Vamos a asignar la letra correspondiente}
            Begin
              If i = hexadecimal Then
                 hex_letra := chr(ascii_A + j);
              {El valor ASCII de la letra hexadecimal aumenta a la vez que la i,
               por lo que A se va a corresponder con 10, B con 11, C con 12,
               D con 13, E con 14 y F con 15}

              j := j + 1 {Al aumentar j, aumenta el valor ASCII
                          y obtenemos la siguiente letra}
            End;

          Write(hex_letra); {Escribimos por pantalla la letra correspondiente}
        End
      Else
        Write(hexadecimal);
       {Si el valor del cuarteto de bits es < 9 no hay que realizar cambios}

      izq := izq + 4; {Aumentamos 4 para pasar al siguiente cuarteto}

  Until izq > 12  {Cuando hallamos recorrido todos los cuartetos finalizamos}
End. {p.p}
```

P15. Sentencias de repetición (I).

El valor del número Pi se puede calcular sabiendo que la serie:

$$1 - 1/3 + 1/5 - 1/7 + 1/9 - 1/11 + 1/13 - 1/15...$$

tiende a Pi/4. Escriban un programa que calcule una aproximación al número Pi usando un número de terminos suministrado por el usuario.

```
{Este programa nos va a permitir realizar una aproximacion al
```

```
 numero pi. Un ejemplo: Para 10.000.000 de terminos, el valor
 de pi es: 3.141592553589792}

{Debe estar en modo 80x87 para compilar correctamente}

Program P15 (INPUT,OUTPUT);
   Uses Crt;

        Var
           {Vamos a declarar todas las variables de tipo Extended
            porque lo que buscamos es una gran precision}

           terminos: Extended;          {Numero de terminos a calcular}
           suma_pos,suma_neg: Extended; {Suma de los numeros positivos
                                         y negativos de la sucesion}
           suma_total: Extended;        {Suma total de la sucesion}
           i: Extended;                 {Va a ser el divisor en la sucesion}

Begin {p.p}
   ClrScr;

   Repeat
      Write('Introduzca el numero de terminos: ');
      Read(terminos)
   Until terminos > 0;

   {Inicializamos las variables}
   i := 1;
   suma_pos := 0;
   suma_neg := 0;

   While i < (terminos * 2) Do
   {Como en la sucesion i solamente toma valores impares, va dando saltos de
    2 en 2, y entonces i debe duplicar los terminos introducidos por el usuario}
      Begin
            suma_pos := suma_pos + (4 / i); {Suma de cada termino positivo}
            i := i + 2;                     {Pasamos al siguiente divisor}

            If i < (terminos * 2) Then
            {Con este if evitamos que al meter un numero de terminos
             impar se ejecute tambien esta parte}
              Begin
                suma_neg := suma_neg + (4 / i); {Suma de cada termino negativo}
                i := i + 2                      {Pasamos al siguiente divisor}
              End;
      End;

   suma_total := suma_pos - suma_neg; {Hacemos la resta de los los terminos
                                       positivos y negativos de la sucesion}

   Write('La aproximacion al numero pi es: ',suma_total:18:15)
   {Mostramos por pantalla el valor final de la aproximacion}

End. {p.p}
```

Ahora vamos a realizar el prgrama de la aproximacion al numero PI, pero vamos a utilizar un nueva función (GetTime) que nos permite saber el tiempo en cada instante.

```
Program P15_time(INPUT,OUTPUT);
   Uses Crt,Dos; {Es necesario poner las librerias correctas}

        Var
           {Vamos a declarar todas las variables de tipo Extended
            porque lo que buscamos es una gran precision}

           terminos: Extended;          {Numero de terminos a calcular}
           suma_pos,suma_neg: Extended; {Suma de los numeros positivos
                                         y negativos de la sucesion}
           suma_total: Extended;        {Suma total de la sucesion}
           i: Extended;                 {Va a ser el divisor en la sucesion}

           h1,m1,s1,c1,h2,m2,s2,c2: Word;

Begin {p.p}
```

```
   ClrScr;

   Repeat
      Write('Introduzca el numero de terminos: ');
      Read(terminos)
   Until terminos > 0;

   {Inicializamos las variables}
   i := 1;
   suma_pos := 0;
   suma_neg := 0;

   GetTime(h1,m1,s1,c1);
   Writeln(h1,':',m1,':',s1,':',c1);

   While i < (terminos * 2) Do
   {Como en la sucesion i solamente toma valores impares, va dando saltos de
    2 en 2, y entonces i debe duplicar los terminos introducidos por el usuario}
      Begin
            suma_pos := suma_pos + (4 / i); {Suma de cada termino positivo}
            i := i + 2;                     {Pasamos al siguiente divisor}

            If i < (terminos * 2) Then
            {Con este if evitamos que al meter un numero de terminos
             impar se ejecute tambien esta parte}
              Begin
                suma_neg := suma_neg + (4 / i); {Suma de cada termino negativo}
                i := i + 2                      {Pasamos al siguiente divisor}
              End;
      End;

   suma_total := suma_pos - suma_neg; {Hacemos la resta de los los terminos
                                       positivos y negativos de la sucesion}

   Write('La aproximacion al numero pi es: ',suma_total:18:15);
   {Mostramos por pantalla el valor final de la aproximacion}
   Writeln;
   GetTime(h2,m2,s2,c2);
   Writeln(h2,':',m2,':',s2,':',c2);

 {Esta nueva funcion GetTime tiene una gran utilidad. Podemos saber
  el tiempo en cada instante de nuestro programa y ver si podemos
  reducirlo perfeccionando nuestros algoritmos.

  Las pruebas que he hecho en mi ordenador son: (En un portatil Pentium III a 700 Mhz)
  Para calcular 10.000.000 de terminos, mi ordenador tardo:
       * 1 segundo y 54 centesimas.
  Para calcular 100.000.000 de terminos:
       * 15 segundos y 46 centesimas.
  Para calcular 1.000.000.000 de terminos (1 BILLON DE TERMINOS):
       * 2 minutos 14 segundos y 93 centesimas.
  La aproximacion del numero PI que he llegado a obtener es:
       * 3.141592652589835
  El numero en realidad (hasta este numero de cifras) es:
       * 3.141592653589793

  Como podeis ver hemos calculado la "barbaridad" de un BILLON de terminos
  en un tiempo muy pequeño.
  Definitivamente, esto de la informatica es increible y a la vez MARAVILLOSO.}
End. {p.p}
```

P16. Sentencias de repetición(II).

Escriban un programa que dado un número nos diga si este es primo. Los números primos son enteros mayores que 1 sin divisores enteros positivos, exceptuando el 1 y ellos mismos. Una observación interesante es que todos los números primos son impares excepto el 2. Otra observación es que sólo necesitamos comprobar la divisibilidad por números superiores a la raíz cuadrada del número.

```
Program P16 (INPUT,OUTPUT);
     Uses Crt;
```

```
 {Los numeros primos son enteros mayores que 1 sin divisores enteros positivos,
  exceptuando el 1 y ellos mismos. Todos los primos son impares, excepto el 2.
  Solo es necesario comprobar la divisibilidad por numeros superiores a la raiz
  cuadrada del numero.}

     Var
        num: Word;          {Numero introducido por el usuario}
        raiznum: Integer;   {Valor de la raiz redondeada del numero}
        par: Boolean;       {Nos sirve para marcar los numeros que son pares}
        i: Integer;         {Variable que usamos dentro del bucle}

Begin {p.p}
     ClrScr;

     {Pedimos un numero y no lo aceptamos hasta que sea > 1}
     Repeat
        Write('Introduzca un numero entero para ver si es primo: ');
        Read(num);
        Writeln;
     Until num > 1;

     par := num mod 2 = 0; {par va a ser True cuando el numero sea par}

     If num = 2 Then
        Write('El 2 es primo, es el unico numero par que lo es.')
     Else if par Then
        Write('Todos los numeros pares no son primos, excepto el 2.')
     Else
        Begin
           raiznum := round(sqrt(num));
           {Almacenamos la raiz del numero, redondeada}

           {Comprobamos la divisibilidad, desde el valor de la raiz
           cuadrada del numero introducido hasta uno menor al numero.
           Si algun numero le divide, noesprimo tomara el valor true.}
           For i := raiznum To (num - 1) Do
             Begin
                If (num mod i) = 0 Then  {Vemos si tiene divisores}
                   Begin
                      Writeln(num,' no es un numero primo.');
                      Write('Hemos encontrado un divisor: ',i,' le divide.');
                       {Le decimos al usuario un divisor del numero
                        para que pueda certificarlo por si mismo}
                      Exit; {Si no es primo, hemos terminado}
                   End;
             End;

           Writeln(num,' es un numero primo.');
           {Si no se ha encontrado ningun divisor del numero, este sera primo}
        End;
End. {p.p}
```

P17. Cadenas

Escriban un programa que tome como entrada un número natural de 20 cifras a lo sumo, lo lea como cadena de caracteres y obtenga como salida el número con sus cifras en orden inverso (ej., el número 91417 debe pasarse a 71419). Impriman, además, un mensaje si el número es capícua, y la cifra menor del número junto con el número de veces que aparece (para el ejemplo, debería indicar que la cifra menor es el 1 y aparece 2 veces).

```
Program P17 (INPUT,OUTPUT);
   Uses Crt;

   Type
      StrMax = String[20];
   Var
      numero: StrMax;   {Numero introducido por el usuario. Tomado como cadena}
      menor: Char;      {Cifra menor del numero}
      i,j: Byte;        {Variables usadas en los bucles}
      veces :Byte;      {Numero de veces que aparece la cifra menor}
      capicua: Boolean; {Veremos si es capicua}
      resp: Char;       {Para poder introducir otro numero y comenzar de nuevo}
```

```
  Procedure LeerDatos;
   {Leemos el numero introducido por el usuario}
      Var correcto: Boolean;
          c: String;
    Procedure CompruebaNumero;
     {Comprobamos el numero que hemos obtenido}
     Begin
       For i := 1 To length(c) Do
         If (ord(c[i]) < 48) Or (ord(c[i]) > 57) Then
            correcto := False;
         {Chequeamos que el numero solo contenga caracteres numericos}

          If length(c) > 20 Then
            correcto := False;
         {El numero no puede contener mas de 20 digitos}
     End; {CompruebaNumero}

   Begin {LeerDatos}
    Repeat
      correcto := True; {Inicializamos correcto}
      Write('Introduzca un numero natural de 20 cifras como maximo: ');
      Readln(c);
      If c = '' Then  {Evitamos que no se introduzca nada}
        Begin
          Writeln('Por favor, introduzca un numero.');
          correcto := False
        End;
      CompruebaNumero; {Comprobamos que todos los caracteres sean numeros}
      Writeln;
    Until correcto;
    numero := c {Asignamos a numero el valor de c porque es un numero correcto}
   End; {LeerDatos}
Begin {p.p}
```

P18. Funciones y cadenas

Escriban un programa que acepte como entradas dos números octales introducidos por el usuario, y muestre por pantalla su suma en octal. Los números introducidos por el usuario podrán tener un número de cifras entre 1 y 32, y deberán ser leidos por el programa como cadenas de carácteres. Obviamente, el número de cifras de ambos números no tendrá porque coincidir en general.
El programa debera contener una función o procedimiento para leer un número como cadena y asegurar que es un número octal, y otra función o procedimiento que los sume y escriba por pantalla el resultado.

```
Program P18 (INPUT,OUTPUT);
   Uses Crt;

   Type
       digitos = Set of 0..7;  {Digitos que se permiten en octal}
       str32 = String[32];
       bit = 0..1; {Es el tipo del acarreo. Nunca va a poder ser
                    mayor de 1 porque, solo sumamos dos numeros}
   Var
      octal1,octal2: str32;             {Valores octales que leemos como cadena}
      sumaoctal: Array [1..32] Of Byte; {Suma total de los 2 numeros octales}
      acarreo: bit;                     {Valdra 1 o 0}
      i,terminos: Byte;      {Terminos nos ayudara para el formato de salida}
      resp: Char;            {El usuario decidira si quiere realizar otra suma}

      Procedure LeerDatos;
      {Leemos los dos numeros octales y a la vez los comprobamos}
        Var
          correcto: Boolean; {Para chequear los digitos de entrada}
          uno,dos: String;   {Numeros octales}
          octales: digitos;  {Numeros permitidos, van del 0 al 7}
      Begin
        Repeat
          Write('Introduzca el primer numero octal:  '); Readln(uno);
          Write('Introduzca el segundo numero octal: '); Readln(dos);

          correcto := True;  {Inicializamos correcto}
          octales := [0..7]; {Digitos permitidos en octal}
```

```
    If (uno = '') Or (dos = '') Then
      correcto := False;
    If (length(uno) > 32) Or (length(dos) > 32) Then
      correcto := False;

    {Vemos si los caracteres introducidos pertenecen a numeros octales}
    For i := 1 To length(uno) Do
      If Not((ord(uno[i]) - 48) In octales) Then
         correcto := False;
    For i := 1 To length(dos) Do
      If Not((ord(dos[i]) - 48) In octales) Then
         correcto := False;
  Until correcto;
  octal1 := uno; {Cuando sean valores correctos los acumulamos}
  octal2 := dos;
End; {leerdatos}

Procedure suma;
{Este procedimiento realiza la suma de dos numeros octales}
   Var
      oct1,oct2: Byte;             {Octales que pasamos a enteros}
      columna: Byte;               {Valor de cada columna}
      fin_oct1, fin_oct2: Boolean; {Final de cada numero octal}
      j,k: Byte;
Begin
  columna := 0;
  acarreo := 0;
  i := 1;
  j := 1;
  k := 1;
  fin_oct1 := False;
  fin_oct2 := False;
  Repeat
     {Continuamos con el siguiente digito hasta el final del octal}
     If Not fin_oct1 Then
        oct1 := ord(octal1[i]) - 48 {Obtenemos el numero natural}
     Else
        oct1 := 0; {Las cadenas pueden ser distintas. Le damos el valor
                    del elemento neutro, el 0, para poder seguir sumando}
     If Not fin_oct2 Then
        oct2 := ord(octal2[j]) - 48
     Else
        oct2 := 0;

     {Obtemos el numero octal de cada columna mirando modulo 8, es decir,
      calculamos el resto que dejan. Este sera el valor de la columna}
     columna := (oct1 + oct2 + acarreo) mod 8;
     {Si no hemos terminado con los 2 octales calculamos el acarreo}
     If Not (fin_oct1 and fin_oct2 and (oct1 = 0) and (oct2 = 0)) Then
        acarreo := (oct1 + oct2 + acarreo) div 8;
     sumaoctal[k] := columna;
     {Vamos acumulando los resultados de la suma total}

     If i = length(octal1) Then
        fin_oct1 := True  {Hemos llegado al final del primer octal}
     Else
        i := i + 1;       {Pasamos al siguiente digito}
     If j = length(octal2) Then
        fin_oct2 := True  {Hemos llegado al final del segundo octal}
     Else
        j := j + 1;       {Pasamos al siguiente digito}

     k := k + 1;          {Siguiente posicion de sumaoctal}
  Until fin_oct1 and fin_oct2 and (oct1 = 0) and (oct2 = 0);
End; {suma}

Procedure invierte(Var octal:str32);
{Invertimos la cadena del numero octal}
  Var
     aux: str32;
     j: Byte;
Begin
  For i := length(octal) Downto 1 Do
     aux[i] := octal[i];  {Copiamos el numero octal}
  j := 1;
  For i := length(octal) Downto 1 Do
```

```
          Begin
            octal[j] := aux[i]; {Le invertimos para leer de izquierda a derecha}
            j := j + 1;
          End;
      End; {invierte}

      Function EscribeOctal(octal: str32):str32;
      {Funcion para sacar el formato de la suma de los octales por pantalla}
      Begin
        For i := (length(octal) + 1) To 32 Do
          octal[i] := ' ';
          {Limpiamos todo lo que no es el numero octal y evitamos que aparezcan
           caracteres extraños. Lo queremos rellenar con blancos}
        Writeln;
        Write(' ':20);
        For i := terminos Downto 1 Do
           Write(octal[i]); {Escribimos el numero con las posiciones necesarias}
      End; {escribeoctal}

      Procedure SumaOctales;
      {Mostramos la suma de los numeros octales}
       Begin
        Writeln;
        Write(' ':19); {Si hay acarreo dejamos un espacio menos en blanco}
        Case acarreo of
          1:
             For i := (terminos + 1) Downto 1 Do
                Write(sumaoctal[i]);
          0:
             Begin
               Write(' '); {No hay acarreo, necesitamos un espacio mas}
               For i := terminos Downto 1 Do
                  Write(sumaoctal[i])
             End
        End; {case}
        Writeln
      End;


Begin {p.p}
   ClrScr;
 Repeat
   {Inicializamos los datos}
   For i := 1 to 32 Do
    Begin
     sumaoctal[i] := 0;
     octal1[i] := ' ';
     octal2[i] := ' ';
    End;

   LeerDatos; {Pedimos al usuarios los datos}
   Writeln;
   Invierte(octal1); {Invertimos las cadenas de octales}
   Invierte(octal2);
   suma; {Sumamos los dos numeros}

   {Vemos que cadena tiene mayor numero de terminos, lo utilizaremos
    para mostrar un formato de salida por pantalla uniforme}
   If length(octal1) > length(octal2) Then
      terminos := length(octal1)
   Else
      terminos := length(octal2);

   {Sacamos por pantalla el formato adecuado para mostrar la suma}
   Writeln('La suma de los numeros octales es: ');
   EscribeOctal(octal1);
   EscribeOctal(octal2);
   Writeln;
   Write(' ':20); {Vamos a escribir la linea de separacion}
   For i := terminos Downto 1 Do
     Write('-');

   SumaOctales; {Mostramos por pantalla el resultado de sumar los 2 numeros}

   {Damos la posibilidad al usuario de poder realizar mas sumas}
   Writeln;
```

```
   Writeln('¿Desea realizar otra suma?');
   Write('Escriba "N" para terminar o cualquier caracter para continuar: ');
   Readln(resp);
   Writeln
 Until (resp = 'n') Or (resp = 'N')
End. {p.p}
```

P19. Recursividad

Escriban un programa para calcular el máximo común divisor (mcd) de dos números naturales que se le suministren. Utilizar el algoritmo clásico de Euclides, que se basa en la siguiente recurrencia:

mcd(0,n) = n

mcd(n,0) = n

mcd(m,n) = mcd(max(n,m) mod min(n,m), min(n,m)), para m>0, n>0.

```
Program P19 (Input,Output);
 {Este programa va calcular el Mcd de (n,m) mediante el Algoritmo de Euclides}
  Uses Crt;

  Var
    a,b: LongInt;
    max,min: LongInt;

  Function Mcd (n,m: LongInt) :LongInt;
    Begin
      {Vemos que numero es mayor y cual menor}
      If n > m Then
        Begin
          max := n;
          min := m
        End
      Else
        Begin
          max := m;
          min := n
        End;
      {Si alguno de los valores es 0, el Mcd sera el otro numero,
       si los 2 son 0, entonces el Mcd valdra 0}
      If n = 0 Then
         Mcd := m
      Else If m = 0 Then
         Mcd := n
      Else Mcd := Mcd(max mod min,min); {Calculamos el Mcd}
    End; {Mcd}

Begin {p.p}
  ClrScr;

   {Pedimos al usuario los datos}
   Repeat
     Write('Calculo del Mcd de (n,m). Introduzca dos numeros naturales: ');
     Read(a,b);
   Until (a >= 0 ) and (b >= 0);

   Writeln;
   Writeln('El Maximo comun divisor de ',a,' y ',b,' es ', Mcd(a,b))

End.{p.p}
```

P20. Conjuntos

Escriban un programa que acepte una cadena de caracteres, introducida por teclado, y determine: a) el número total de letras; b) el número de letras minúsculas; c) el número de dígitos pares; d) el número de signos de puntuación (',', ';', ':', '.', '?','¿', '!', '¡'); e) el número de blancos; f) el número de caracteres de la cadena que aparezcan también en una cadena constante. Utilícense conjuntos allí donde sea apropiado. En particular, el programa deberá generar para f) un conjunto a partir de los caracteres de la cadena constante.

```
Program P20 (Input,Output);
   Uses Crt;
```

```
   Const
       {Nuestra frase constante}
       CadenaCte: String = '~> Victor Sanchez2 y Jose Maria18!!!!!';
       pares:  Set of Char = ['0','2','4','6','8'];
       signos: Set of Char = [',',';',':','.','¿','?','','!'];
       mayus:  Set of Char = ['A'..'Z'];
       minus:  Set of Char = ['a'..'z'];

   Type
      ConjCar = Set of Char; {Conjunto de caracteres}

   Var
      i: Byte;                {Variable para los bucles}
      ConjCte: ConjCar;       {Le creamos a partir de nuestra cadena constante}
      caracteres: String;     {Cadena introducida por el usuario}
      ConjUsuario: ConjCar;   {Creado a partir de la cadena del usuario}
      interseccion: ConjCar;  {Caracteres comunes entre ConjCte y ConjUsuario}
      num_letras,num_minus,num_pares,num_signos,num_blancos: Byte;
      {Estadisticas que se nos pide mostrar}

   Procedure CreaConjunto(cadena: String;
                          Var conjunto: ConjCar;
                          longitud: Byte);
   {Creamos un conjunto a partir de una cadena de caracteres dada}
   Begin
    Write('[ '); {Corchete de inicio del conjunto}
    For i := 1 To longitud Do
      Begin
      {Incorporamos los caracteres al conjunto y le vamos formando}
        If Not (cadena[i] In conjunto) Then
          Begin
            conjunto := conjunto + [cadena[i]];
            Write(cadena[i],', ') {Escribimos el caracter}
          End;
      End;
    Write(']'); {Corchete de fin del conjunto}
    Writeln;
   End;

Begin {p.p}
  ClrScr;

  num_letras := 0;
  num_minus := 0;
  num_pares := 0;
  num_signos := 0;
  num_blancos := 0;

  Write('Introduzca una cadena de caracteres: ');
  Readln(caracteres);
  Writeln;

  {Comprobamos caracter por caracter y vemos a que grupo pertenece}
  For i := 1 To length(caracteres) Do
    Begin
      If (caracteres[i] In mayus) Or (caracteres[i] In minus) Then
        num_letras := num_letras + 1;

      If (caracteres[i] In minus) Then
        num_minus := num_minus + 1
      Else If (caracteres[i] In pares) Then
        num_pares := num_pares + 1
      Else If (caracteres[i] In signos) Then
        num_signos := num_signos + 1
      Else If caracteres[i] = ' ' Then
        num_blancos := num_blancos + 1
    End;

  Writeln;
  Writeln('Numero de letras:------> ',num_letras:2);
  Writeln('Numero de minusculas:--> ',num_minus:2);
  Writeln('Numero de pares:-------> ',num_pares:2);
  Writeln('Numero de signos:------> ',num_signos:2);
  Writeln('Numero de blancos:-----> ',num_blancos:2);
  Writeln;
```

```
  {Mostramos por pantalla nuestra cadena constante}
  Write('Nuestra cadena constante es: ');
  For i := 1 To length(CadenaCte) Do
    Write(CadenaCte[i]);
  Writeln;

  {Inicializamos los conjuntos y procedemos a crearlos}
  ConjCte := [];
  ConjUsuario := [];
  Writeln;
  Writeln('Conjunto creado a partir de nuestra cadena constante: ');
  CreaConjunto(CadenaCte,ConjCte,length(CadenaCte));
  Writeln;
  Writeln('Conjunto creado a partir de la cadena del usuario: ');
  CreaConjunto(caracteres,ConjUsuario,length(caracteres));

  {La interseccion son los caracteres comunes a los dos conjuntos}
  interseccion := ConjCte * ConjUsuario;

  Writeln; {Mostramos por pantalla los caracteres comunes a los conjuntos}
  Writeln('Comparando la cadena del usuario con nuestra cadena');
  Writeln('constante, los caracteres que se repiten son:');
  i := 1;
  Write('[ '); {Corchete de inicio del conjunto}
  Repeat
    If caracteres[i] In interseccion Then
       Write(caracteres[i],', ');
       interseccion := interseccion - [caracteres[i]];
       {Vaciamos el conjunto para evitar que haya caracteres repetidos}
       i := i + 1
  Until (interseccion = []) Or (i = length(caracteres));
  Write(']')   {Corchete de fin del conjunto}
End. {p.p}
```

P21. Arrays.

Escribid un programa para verificar propiedades de las matrices. El programa deberá:

a) *Mediante un procedimiento leer DOS matrices cuadradas de orden 5 de un fichero como el ejemplo matriz.*
b) Efectuar control de errores sobre el orden de la matriz (deben ser cuadradas).
c) Mediante una función comprobar si la matriz A posee la propiedad antisimétrica, es decir que A' = -A
d) Demostrar que (A·B)' = B' · A'. El producto de matrices se implementará como un procedimiento.

Observación: Se recomienda implementar un procedimiento de impresión de matrices en pantalla con formato

Matriz A
0 1 2 3 4
-1 0 5 6 7
-2 -5 0 8 9
-3 -6 -8 0 0
-4 -7 -9 0 0
Matriz B
1 0 0 0 0
0 1 0 0 0
0 0 1 0 0
0 0 0 1 0
0 0 0 0 1

```
Program P21 (INPUT,OUTPUT);
  Uses Crt;

   Const
        MaxDim = 5;  {Aqui podremos declarar el tamaño de las matrices}
        T = Chr(39); {Simbolo ' de las matrices traspuestas}

   Type
      matriz = Array[1..MaxDim,1..MaxDim] Of Integer;
      {Tipo para matrices cuadradas de una dimension constante}

   Var
      fich: Text;    {Fichero al que le asignaremos el nombre correspondiente}
      i,j: Byte;     {Variables para usar dentro de los bucles}
      A,B,C: matriz;             {Matrices}
      nombre: String;            {Nombre del archivo donde estan las matrices}
      TrasA,TrasB,TrasC: matriz; {Matrices traspuestas}
      iguales: Boolean;   {Para chequear igualdades entre matrices}
      error: Boolean;     {Chequeamos que las matrices sean cuadradas}
      resp: String;       {Respuestas del usuario a nuestras preguntas}

   Procedure errores;
   {Las matrices deben ser cuadradas. Buscamos posibles fallos en el orden}
    Var
       etiqueta: String;  {Lee "MatrizA" y "MatrizB" si el archivo es correcto}
       digitos: String;   {Digitos leidos en cada fila del archivo}
       blancos,num_blancos: Byte; {Conociendo el numero de blancos por linea
                                   sabemos el numero de digitos de la matriz
                                   en cada fila y averiguamos si es cuadrada.}
       LeeLinea: Byte; {Lineas donde deben estar colocadas las etiquetas
                        "MatrizA" y "MatrizB". Si en el archivo no han sido
                        introducidas correctamente aparece un mensaje de error}
    Begin
     Reset(fich); {Abrimos el fichero para leer las matrices}
     {Inicializamos los datos}
     error := False;
     digitos := '';
     j := 1;
     num_blancos := 0;
     {En la primera linea debe aparecer la etiqueta de la primera matriz}
     LeeLinea := 1;
     Repeat
       blancos := 0; {Ponemos a 0 el contador de blancos}
       {Leemos las lineas donde deben estar situadas
        las etiquetas de las matrices}
       If (j = 1) Or (j = LeeLinea) Then
          Readln(fich,etiqueta);

       Read(fich,digitos); {Digitos de la fila de la matriz}

       {Si es la primera linea con digitos acumulamos el numero de blancos}
       If j = 1 Then
         For i := 1 To length(digitos) Do
            If digitos[i] = ' ' Then
               num_blancos := num_blancos + 1;

       {Contamos el numero de blancos de cada fila para saber si es cuadrada}
       For i := 1 To length(digitos) Do
         If digitos[i] = ' ' Then
            blancos := blancos + 1;
       {Todas las lineas deben tener el mismo numero de blancos, de lo contrario
        no sera una matriz cuadrada o el usuario ha generado un archivo erroneo}
       If blancos <> num_blancos Then
         error := True;
       {Si terminamos la linea pasamos a la siguiente}
       If Eoln(fich) Then
         Begin
           Readln(fich);
           Writeln
         End;
      j := j + 1;
      {Si es una matriz cuadrada, LeeLinea sera la linea donde
       se encuentra la etiqueta de la matriz}
      LeeLinea := num_blancos + 2;
      Until Eof(fich) Or error; {Hasta que termine el fichero o haya un error}
```

```
      {Si no hay errores, pero la matriz no es del orden que pedimos mostramos
       un mensaje de error}
      If Not error And (num_blancos + 1 <> MaxDim) Then
        Begin
          Writeln('Las matrices son cuadradas, pero no son ',MaxDim,'x',MaxDim);
          error := True
        End;
      Close(fich) {Cerramos el fichero}
    End; {errores}

   Procedure LeerMatriz(Var m: matriz);
   {Procedimiento que lee una matriz de un archivo dado y la guarda.
    m es la matriz con sus digitos correspondientes}
    Var
       num: Integer; {Cada numero de la matriz que vamos leyendo del archivo}
       e: String;    {etiqueda de la matriz del archivo}
    Begin
      Readln(fich,e); {Leemos la etiqueta de la matriz}
      {Acumulamos la matriz leida del archivo}
      For i := 1 To MaxDim Do
        For j := 1 To MaxDim Do
          Begin
            Read(fich,num);
            m[i,j] := num;
          End;
      Readln(fich);
      Writeln;
    End; {LeerMatriz}

    Procedure multiplica(M1,M2: matriz; Var Mresul: matriz);
    {Multiplicacion de dos matrices que se guarda en otra matriz resultado.
     M1 y M2 son las matrices que recibimos, las multiplicamos y guardamos
     el resultado en Mresul}
     Var
        x: Integer; {Para acumular la suma al multiplicar fila por columna}
        k: Byte;    {Para recorrer el bucle}
     Begin
       For i := 1 To MaxDim Do
         For j := 1 To MaxDim Do
           Begin
             x := 0;
             For k := 1 To MaxDim Do
               x := x + M1[i,k] * M2[k,j];
             Mresul[i,j] := x; {Resultado de multiplicar fila por columna,
                                lo acumulamos en su correspondiente casilla}
           End;
     End; {multiplica}

    Procedure traspuesta(m: matriz; Var traspuesta: matriz);
    {Recibimos una matriz (m) y el lugar donde lo guardamos (traspuesta).
     Con la matriz m y generamos su matriz traspuesta, la cual sera
     guardada en traspuesta}
     Begin
       For i := 1 To MaxDim Do
         For j := 1 To MaxDim Do
           traspuesta[j,i] := m[i,j];  {Obtenemos la matriz traspuesta}
     End; {traspuesta}

    Procedure MuestraMatriz(m,n: matriz);
    {Mostramos por pantalla, con el formato adecuado,
     parejas de matrices recibidas (m y n)}
     Begin
       Writeln;
       For i := 1 To MaxDim Do
         Begin
           For j := 1 To MaxDim Do
             Write(m[i,j]:3);      {Muestra la primera matriz}
           Write(' ':5);           {5 espacios de separacion entre matrices}
           For j := 1 To MaxDim Do
             Write(n[i,j]:3);      {Muestra la segunda matriz}
           Writeln                 {Pasamos a escribir en la siguiente linea}
         End;
     End; {MuestraMatriz}

Begin {p.p}
```

```
ClrScr;
{Pedimos el nombre del fichero donde se encuentran los datos}
Write('Introduzca el nombre del archivo con su ruta adecuada: ');
Readln(nombre);
Assign(fich,nombre); {Asignamos el nombre del fichero a fich}
errores;             {Chequeamos las matrices en busca de errores}
resp := ' ';         {Inicializamos la respuesta}
If error Then  {Si hay algun error se lo comunicamos al usuario y terminamos}
  Repeat
    Writeln('Los datos suministrados son incorrectos.');
    Writeln('Asegurese de que el archivo contiene las matrices adecuadas.');
    Write('Por favor, pulse F/f para terminar: ');
    Readln(resp);
  Until (resp = 'F') Or (resp = 'f');

Reset(fich);
{Si no hay errores continuamos con el programa}
If (resp <> 'F') And (resp <> 'f') Then
  Begin
    LeerMatriz(A);             {Leemos la primera matriz}
    LeerMatriz(B);             {Leemos la segunda matriz}
    traspuesta(A,TrasA);       {Traspuesta de la matriz A}
    traspuesta(B,TrasB);       {Traspuesta de la matriz B}
    multiplica(A,B,C);         {  (A.B)   -> Lo acumulamos en C}
    traspuesta(C,TrasC);       {  (A.B)'  -> Lo acumulamos en TrasC}
    multiplica(TrasB,TrasA,C); {   B'.A'  -> Lo acumulamos en C}
    ClrScr;

    {Empezamos a mostrar por pantalla las matrices obtenidas}
    Writeln('Estas son las matrices leidas del archivo ',nombre,':');
    MuestraMatriz(A,B);
    Writeln;
    Writeln('Y estas sus traspuestas: ');
    MuestraMatriz(TrasA,TrasB);
    Writeln;
    {Mensaje al usuario para continuar con la siguiente pantalla}
    Writeln('Continuamos con las propiedades de las matrices...');
    Writeln('Pulse "enter" por favor.');
    Readln(resp);
    ClrScr;
    {Vamos a comprobar si: (A.B)' = B'.A' }
    iguales := True; {Inicializamos la variable iguales}
    For i := 1 To MaxDim Do
      For j := 1 To MaxDim Do
        If TrasC[i,j] <> C[i,j] Then
          iguales := False; {Si las matrices son distintas iguales := false}
    If iguales Then
      Begin
        {Se cumple la propiedad. Lo mostramos acompañado de las matrices}
        Writeln('Se cumple que: (A.B)',T,' = B',T,'.A',T); {(A.B)' = B'.A'}
        Writeln;
        Writeln('(A.B)':11,T,'B':15,T,'.A',T);             {(A.B)' = B'.A'}
        MuestraMatriz(TrasC,C);
      End
    Else  {No se cumple (A.B)' = B'.A'}
      Writeln('No se cumple: (A.B)',T,' = B',T,'.A',T);

    Writeln;
    Writeln('Pasemos a ver si la matriz es antisimetrica.');
    Writeln('Vuelva a pulsar "enter" por favor.');
    Read(resp);
    {Vemos si es antisimetrica: A' = -A }
    iguales := True;
    For i := 1 To MaxDim Do
      For j := 1 To MaxDim Do
        If TrasA[i,j] <> -A[i,j] Then
          iguales := False
          {Si las matrices son distintas toma el valor false}
        Else
          C[i,j] := -A[i,j]; {Usamos C para acumular -A}
    If iguales Then
      Begin  {A' = -A}
        {Se cumple la propiedad antisimetrica. Mostramos las matrices}
        Writeln('La matriz A posee la propiedad antisimetrica: A',T,' = -A');
        Writeln;
        Writeln('A':9,T,'-A':19); {A' = -A}
```

```
          MuestraMatriz(TrasA,C);
        End
      Else {No cumple la antisimetria}
        Begin
          Writeln('La matriz A no posee la propiedad antisimetrica: A` = -A');
          MuestraMatriz(TrasA,C);
        End;
    End;
    Close(fich)  {Cerramos el fichero}
End. {p,p}
```

P22. Registros

Un número complejo tiene la forma a + bi, donde a y b son números reales. Las operaciones básicas con números complejos son:

Suma: (a + bi) + (c + di) = (a + c) + (b + d)i
Resta: (a + bi) - (c + di) = (a - c) + (b - d)i
*Producto: (a + bi) * (c + di) = (a·c - b·d) + (a·d + b·c)i*
Cociente: (a+bi) / (c+di) = ((ac + bd) / (c² + d²)) + ((bc –ad) / (c ² + d²))

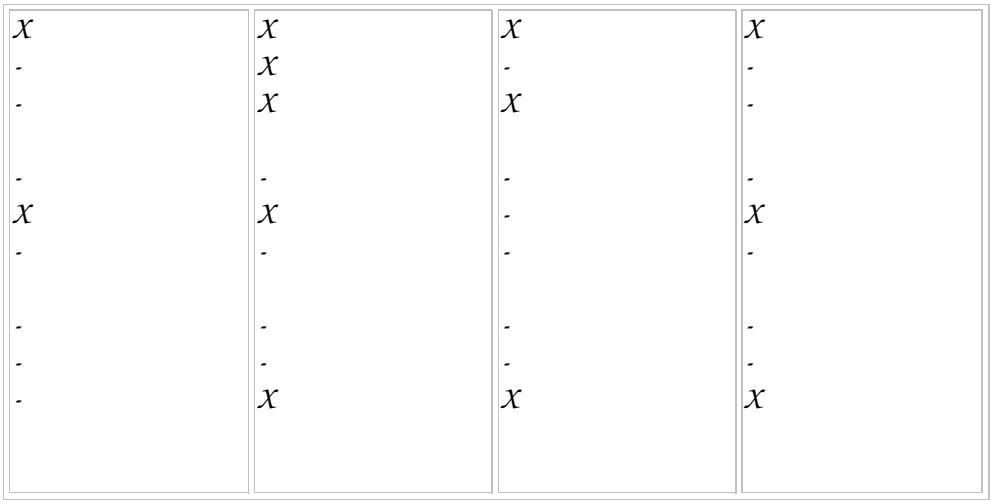
Escribir un programa que lea dos números complejos y el símbolo correspondiente a la operación y ejecute la operación indicada.
Utilizar un registro para representar números complejos y procedimientos para realizar las operaciones.

```
Program P22 (Input,Output);
  Uses Crt;

  Type
      numero =
       Record
         real: Real; {Parte real del numero}
         imag: Real; {Parte imaginaria del numero}
       End;

  Var
     n1,n2: numero; {Guardaremos la parte real e imaginaria de cada numero}
     resp: Char;    {El usuario elegira si desea realizar otra operacion}

Procedure LeerDatos(Var n: numero);
{El usuario introduce el valor de los numeros complejos. La variable
 n se pasa por referencia y va a acumular el numero complejo en n1 o n2}
 Begin
   With n Do  {Obtenemos los datos del numero complejo}
     Begin
       Writeln;
       Writeln('Introduzca el numero complejo:');
       Write('Parte real: ');
       Readln(real);
       Write('Imaginaria: ');
       Readln(imag);
       Writeln;
       Write('El numero complejo es:  ');
       If imag >= 0 Then
       {Con este If vemos si es positivo o negativo y ponemos el signo correcto
        en la salida por pantalla para que quede bien centrado}
          Writeln(real:5:2,' + ',imag:2:2,'i')
       Else
          Writeln(real:5:2,' - ',abs(imag):2:2,'i');
          {Usamos el valor absoluto para poder centrar el signo}
       Writeln;
     End;
End; {LeerDatos}

Procedure suma;
```

```
{Realizamos la suma de los numeros complejos}
Begin
   Write('El resultado de sumar los numeros complejos es: ');
   If (n1.imag + n2.imag) >= 0 Then {Vamos a colocar el signo correcto}
      Writeln((n1.real + n2.real):2:2 ,' + ',(n1.imag + n2.imag):2:2,'i')
   Else
      Writeln((n1.real + n2.real):2:2 ,' - ',abs(n1.imag + n2.imag):2:2,'i')
      {Ponemos el valor absoluto para sacar el
       signo - centrado y no pegado al numero}
 End; {suma}

Procedure resta;
{Realizamos la resta de los numeros complejos}
 Begin
   Write('El resultado es: ');
   If (n1.imag - n2.imag) >= 0 Then
     Writeln((n1.real - n2.real):2:2,' + ',(n1.imag - n2.imag):2:2,'i')
   Else
     Writeln((n1.real - n2.real):2:2,' - ',abs(n1.imag - n2.imag):2:2,'i')
End; {resta}

Procedure producto;
{Se hace la multiplicacion de los numeros complejos}
 Begin
   Write('El producto de los numeros complejos es: ');
   If ((n1.real * n2.imag) + (n1.imag * n2.real)) >= 0 Then
     Begin
       Write(((n1.real * n2.real) - (n1.imag * n2.imag)):2:2);
       Write(' + ');
       Writeln((n1.real * n2.imag) + (n1.imag * n2.real):2:2,'i')
     End
   Else
     Begin
       Write(((n1.real * n2.real) - (n1.imag * n2.imag)):2:2);
       Write(' - ');
       Writeln(abs(((n1.real * n2.imag) + (n1.imag * n2.real))):2:2,'i')
     End
End; {producto}

Procedure cociente;
{Se hace la division de los numeros complejos}
 Var
    DividendoR, DividendoI: Real; {Dividendo Real e Imaginario}
    divisor: Real;                {Es el mismo para los dos}
 Begin
   {Para hacer la division, el divisor no puede ser 0. La unica forma
    de que el cuadrado de n2.real mas el cuadrado de n2.imag sea igual
    a 0 es que los 2 sean 0}
   If (n2.real = 0) And (n2.imag = 0) Then
     Begin
       Writeln('La suma de los cuadrados de la parte real mas la imaginaria ');
       Writeln('del segundo numero debe ser distinto de cero.')
     End
   Else
     Begin
       {Hallamos los dividendos y divisores de la operacion}
       DividendoR := (n1.real * n2.real) + (n1.imag * n2.imag);
       divisor := sqr(n2.real) + sqr(n2.imag);
       DividendoI := (n1.imag * n2.real) - (n1.real * n2.imag);
       {Mostramos por pantalla el resultado de realizar el cociente}
       Write('El cociente de los 2 numeros es: ');
       Write(DividendoR / divisor:2:2);
       If ((DividendoI / divisor) > 0) Then
         Writeln(' + ',DividendoI / divisor:2:2,'i')
       Else
         Writeln(' - ',abs(DividendoI) / divisor:2:2,'i');
     End
 End; {cociente}

 Procedure Operacion;
  {Recibimos un caracter del usuario que interpretaremos
   como la operacion que debemos realizar con los numeros complejos}
  Var
     oper: Char;
     {Veremos si el usuario quiere sumar, restar, multiplicar o dividir}
  Begin
```

```
    Write('Escriba la operacion que desea realizar (+,-,*,/): ');
    Readln(oper);
    Writeln;
    Case oper Of  {Realizamos la operacion correspondiente}
        '+':suma;
        '-':resta;
        '*':producto;
        '/':cociente
     Else
        Writeln('Ha introducido un operador incorrecto.');
    End; {case}
  End; {Operacion}

Begin {p.p}
  ClrScr;

 Repeat
   {Leemos los datos del programa}
   Writeln('Por favor, escriba dos numeros complejos.');
   Writeln('Primero la parte real, y despues la imaginaria (sin i): ');
   LeerDatos(n1); {Primer numero complejo}
   LeerDatos(n2); {Segundo numero complejo}
   Writeln;
   Operacion;  {Realizamos la operacion que nos indique el usuario}
   Writeln;
   Writeln;
   Write('¿Desea realizar otra operacion? (N/n para terminar): ');
   Readln(resp);
   Writeln;
 Until (resp = 'N') Or (resp = 'n');
 ClrScr;
End.{p.p}
```

Se pide implementar el juego de la vida, propuesto por el matemático americano John Conway. El juego utiliza un tablero que, para esta implementación, tendrá una dimensión de 15 filas por 20 columnas. Cada celda de este tablero podrá estar ocupada por una célula o podrá estar vacía. En el juego se parte de una configuración inicial de células (suministrada por el usuario o generada al azar) y se deja evolucionar esta población de acuerdo a las siguientes reglas definidas por Conway:
Una célula presente en la generación t desaparecerá en la generación t+1 si:
- Se encontraba rodeada por menos de 2 células (muerte por soledad)
- Se encontraba rodeada por mas de 3 células (muerte por superpoblación)
Una célula nacerá (aparecerá) en la generación t+1 en una celda vacía si esa celda se encontraba rodeada por exactamente tres células vivas en la generación anterior.
En cualquier otro caso, una celda no variará su estado al pasar de la generación t a la t+1.
Ejemplos:



La célula central desaparecerá en la siguiente generación por soledad (1 vecino)	*la célula central morirá en la siguiente generación por superpoblación (4 vecinos)*	*Una nueva célula nacerá en la siguiente generación en la casilla central (3 vecinos)*	*La celda central seguirá conteniendo una célula viva en la siguiente generación (2 vecinos)*

Algunas configuraciones iniciales interesantes:

x x - x - x - x x	- - - x x x - - -	- x - - x x x - x	- x x x x - - x -
patrón estático("barco")	*patrón oscilante ("faro")*	*patrón "deslizador"*	*patrón simple de evolución muy compleja (R-pentomino)*

Notas Importantes

a) A efectos de aplicación de las reglas, una celda del tablero se considerará rodeada por ocho celdas vecinas (celdas a izquierda-derecha, arriba-abajo y las cuatro celdas contiguas en diagonal).

b) La supervivencia, nacimiento o muerte de las células deberá determinarse considerando sus vecinos en esa generación. Es decir, no deberán realizarse cambios sobre el tablero hasta que no se haya determinado el destino de todas las celdas en la siguiente generación.

Implementación

El programa incorporará las características que se describen a continuación.

El programa aceptará o generará una configuración inicial de células sobre el tablero definido. El programa permitirá tanto calcular la evolución de esta población de generación en generación como calcular de una vez el resultado tras N generaciones (parámetro suministrado por el usuario). Tras mostrar por pantalla la población obtenida (en la generación siguiente en el primer caso o tras N generaciones en el segundo), el programa permitirá:

a) *Terminar el juego.*
b) *Guardar la configuración obtenida a fichero.*
c) *Editar de forma manual el tablero (eliminando células y/o creando otras nuevas).*
d) *Continuar calculando nuevas generaciones.*

El programa permitirá definir la configuración inicial por cualquiera de estos procedimientos:

a) *Leyendo datos suministrados por teclado.*
b) *Generando aleatoriamente la configuración inicial. En este modo, el usuario fijará por teclado el número de celdas ocupadas por células, y el programa distribuirá estas células de forma aleatoria sobre el tablero.*
c) *Leyendo datos de un fichero.*

El programa permitirá configurar el comportamiento del juego con células que se encuentren en los bordes del tablero. En particular, será posible seleccionar cualquiera de estas posibilidades:

a) Tablero plano. Una celda en el borde del tablero evolucionará como si todos sus vecinos no accesibles fueran celdas vacías.
b) Tablero cilíndrico. En este modo, se tomarán la primera y última columnas del tablero como contiguas.
c) Tablero toroidal. En este modo, se considerarán como contiguas tanto la primera y última columnas del tablero como la primera y última fila.
d)

Evaluación
No se evaluará el aspecto gráfico más allá de la presentación básica del juego.

Se evaluará positivamente :
a) Modularidad y estructuración del cáligo.
b) Cantidad y calidad de los comentarios.
c) Claridad en el cáligo.
d) Calidad de la documentación asociada, memoria y análisis.
e) Uso de estructuras de datos adecuadas al problema.
f) Nombre de las variables y constantes acordes a la función que desempeñan en el programa.

Se penalizará fuertemente :
a) Uso de programación no estructurada, es decir uso de goto, exit y halt.
b) Uso de variables globales dentro de funciones o procedimientos.
c) Funciones o procedimientos no documentados. Cada función o procedimiento debe tener unos comentarios de cabecera con información:
1) qué es lo que realiza la función.
2) qué parámetros de entrada recibe, y cuál es la función de cada uno de ellos.
3) qué parámetros de salida recibe, y cuál es la función de cada uno de ellos.
4) qué valor devuelve.
5) requisitos, o precondiciones en los valores de los parámetros en la llamada a la función, y qué condiciones de error se manejan dentro de cada función.
d) Paso de parámetros innecesarios a funciones y procedimientos.
e) Programas que funcionen incorrectamente.

Se deberá entregar, en un disquete libre de virus :
a) Programas fuentes, debidamente documentados.
b) Memoria, en formato txt, con información sobre :
1) Las estructuras de datos utilizadas.
2) Análisis de cómo se ha implementado el juego, módulos principales, comunicación entre ellos, etc.
3) Qué pruebas se han efectuado, y cuáles han sido los resultados.
4) Qué ficheros son necesarios para la compilación y ejecución del juego.
c) Programas ejecutables y otros archivos necesarios para la ejecución correcta del programa.

```
(*************************************************************
   PROYECTO MTPI (Febrero 2002):
   EL JUEGO DE LA VIDA (Del matematico americano: JOHN CONWAY)
```

```
   Autor: Victor Sanchez Sanchez

   Para el juego se utiliza un tablero de 15 filas por 20 columnas.
   Cada celda del tablero puede estar ocupada por una celula o estar
   vacia.
   La configuracion inicial del juego puede ser aleatoria o ser
   suministrada por el usuario.
   Las celulas pueden evolucionar de acuerdo con las leyes de Conway:
   Una celula presente en la generacion t desaparecera en la generacion
   t+1 si:
   - Se encontraba rodeada por menos de 2 celulas (muerte por soledad)
   - Se encontraba rodeada por mas de 3 celulas (muerte por superpoblacion)
   - Una celula nacera (aparecera) en la generacion t+1 en una celda
     vacia si esa celda se encontraba rodeada por exactamente tres celulas
     vivas en la generacion anterior.

   Una celda se considera rodeada por 8 celulas vecinas:
                    ---
                    -@-
                    ---

   Los cambios sobre el tablero no se realizan hasta que se hayan
   evaluado todas las celulas
*****************************************************************)

Program Conway (Input,Output);
  Uses Crt;

Const
     DimH = 15;  {Numero de filas}
     DimV = 20;  {Numero de columnas}
     respuestas: Set Of Char = ['G','g','E','e','T','t'];
     {Posibles respuestas que puede introducir el usuario}
Type
    tab = Array [1..DimH,1..DimV] Of Char; {Forma del tablero}
    RegTablero =
    {Registro que contiene el tablero y su forma,las generaciones y
     la forma de pasar de generacion (si es de una en una o si son
     varias).}
      Record
        tablero: tab;          {Tablero de 15 filas x 20 columnas}
        FormaTab: Char;        {Forma: Plano, Cilindrico, Toroidal}
        ModoGeneracion: Char;  {Calcular generaciones de una en una o
                                un numero determinado de generaciones}
        NGen: Integer;         {Numero de generaciones a calcular}
        gen: Integer;          {Muestra por pantalla el numero de generaciones}
        cont: Byte;            {Para contar las celulas de alrededor}
      End;
Var
    ConfigTab: RegTablero;  {Registro que contiene la configuracion del tablero}
    TabAux: tab;            {Tablero auxiliar para guardar la generacion}
    i,j: Byte;              {Variables para los bucles}
    resp: String;           {Respuesta del usuario a las diferentes opciones}

{-------------------------------------------------------------------------------------}

Procedure LeerTablero(Var t: RegTablero);
{
 Entrada: t: Tipo del tablero que vamos a leer.

 Funcion: Obtener el tablero desde un fichero introducido por el usuario.

 Salida: Tablero con sus celulas correspondientes. Se guarda en la variable
         tablero perteneciente al registro dado.
}

 Var
    nombre: String; {Nombre del fichero}
    fich: Text;     {Fichero al que asignaremos el nombre}
    casilla: Char;  {Cada casilla de la matriz de celulas}
 Begin
   Writeln;
   Write(' Introduzca el nombre y la ruta del fichero donde esta el tablero: ');
   Readln(nombre);            {Leemos el nombre del fichero}
   Writeln;
   Assign(fich,nombre);       {Le asignamos el nombre a fich}
```

```
   Reset(fich);               {Abrimos el fichero}
   For i := 1 To DimH Do      {Recorremos la matriz [DimH x DimV]}
     Begin
       For j := 1 To DimV Do
         Begin
           Read(fich,casilla);            {Leemos el caracter del fichero}
           If (casilla In ['@','-']) Then
             t.tablero[i,j] := casilla    {Si es correcto lo acumulamos}
           Else                           {Sino mostramos un mensaje de error}
             Begin
               Writeln;
               Writeln(' El archivo introducido contiene fallos, por favor,');
               Writeln(' compruebelo y vuelva a introducir un fichero nuevo.');
               Writeln;
               LeerTablero(t)
             End;
         End;
        {Si no hemos llegado al final del fichero pasamos a la linea siguiente}
        If Not Eof(fich) Then
          Readln(fich)
     End;
     Close(fich);
  End; {LeerTablero}

{-------------------------------------------------------------------------------------}

Function NumCorrecto(Var num: Integer): Integer;
{
 Entrada: Dato en el que vamos a acumular el valor si es un numero.

 Funcion: Procedimiento para saber si el usuario ha introducido un digito o
          un caracter que no corresponde a ningun numero.
          Controlar que el valor que introduce el usuario es un numero.

 Salida: La variable num toma el valor del numero introducido por el usuario.
}
 Var
    correcto: Boolean;
    n: String;

 Function EsNumero: Boolean;
 {
  Funcion: Asegurarnos de que se ha introducido un caracter numerico.

  Salida: La Funcion EsNumero toma el valor true si es un caracter numerico.
 }
  Begin
    EsNumero := False;                              {Inicializamos EsNumero}
    If ((Ord(n[i]) >= 48) And (Ord(n[i]) <= 57)) Then
      EsNumero := True;
  End; {EsNumero}
 Begin {NumCorrecto}
  correcto := True;
  Readln(n);                    {Leemos el caracter introducido por el usuario}
  For i := 1 To length(n) Do
    Begin
      If (length(n) > 1) And (i > 1) Then  {S hay mas de un caracter...}
        Begin
          If EsNumero Then
            num := (num * 10) + (Ord(n[i]) - 48)
            {Le sumamos las decenas necesarias cada vez que se ejecuta el bucle
             y asi vamos obteniendo el numero total}
          Else
            correcto := False      {No se ha introducido un numero}
        End
      Else                         {Si solo ha introducido un caracter...}
        Begin
          If EsNumero Then
            num := Ord(n[i]) - 48  {Obtenemos el valor numerico}
          Else
            correcto := False      {No se ha introducido un numero}
        End
    End; {For}

  If Not correcto Then
    num := -1 {No vamos a realizar ninguna generacion, el numero no es correcto}
```

```
 End; {NumCorrecto}

{-------------------------------------------------------------------------------------}

Procedure TabAleatorio(Var Reg: RegTablero);
{
 Entrada: Registro donde guardaremos el tablero.

 Funcion: Generar una matriz aleatoria de 15x20 con un numero de celulas dado.

 Salida: Tablero aleatorio con su numero de celulas correspondientes.
}
  Var
     ocupadas: Integer;  {Casillas que debe contener el tablero}
     relleno : Integer;  {Numero de casillas que vamos rellenando}
 Begin
  With Reg Do Begin
    Repeat
      Writeln;
      Write('Casillas ocupadas por celulas (Max = 300): ');
      i := NumCorrecto(ocupadas) {Leemos el numero de casillas, chequeamos
                                  que sea correcto y lo acumulamos en ocupadas}
    Until (ocupadas >= 0) And (ocupadas <= 300);
    {Una matriz 15x20 contiene 300 casillas}
   Randomize;
   relleno := 0;
   Repeat
   {Generamos una fila y una columna aleatorias}
      i := Random(DimH) + 1; {Las casillas con valor 0 no nos sirven}
      j := Random(DimV) + 1; {Generamos casillas a partir del 1}
      If (tablero[i,j] <> '@') Then  {Si la casilla no esta ocupada...}
        Begin
          tablero[i,j] := '@';       {Acumulamos una celula}
          relleno := relleno + 1     {Contamos las casillas que rellenamos}
        End
   {Hasta que hallamos rellenado las casillas necesarias}
   Until relleno = ocupadas;

   {Rellenamos los espacios del tablero que queden en blanco}
   For i := 1 To DimH Do
     For j := 1 To DimV Do
       If tablero[i,j] <> '@' Then {Si no hay una celula debe estar vacio}
          tablero[i,j] := '-'
  End {Reg}
 End; {TabAleatorio}

{-------------------------------------------------------------------------------------}

Procedure LimpiaLinea;
{
 Funcion: Limpia la linea de la pantalla la cual necesitemos.
}
 Begin
     Write(' ':48);
 End; {LimpiaLinea}

{-------------------------------------------------------------------------------------}

Procedure DatosManual(Var Reg: RegTablero);
{
 Entrada: Registro donde vamos a guardar el tablero que editamos manualmente.

 Funcion: Permitir al usuario editar manualmente un fichero.

 Salida:  Tablero con sus celulas correspondientes.
}

 Procedure LeeCaracter(Var r: String);
 {
  Entrada: Caracteres que introduce el usuario (variable r).

  Funcion: Leer un caracter para la casilla del tablero.

  Salida:  Se guarda el caracter en el tablero.
 }
  Begin
```

```
   With Reg Do
    Begin
      Readln(r);  {Leemos la casilla}
      If (r = '*') Or (r = '@') Then       {Quiere que aparezca una celula}
        tablero[i,j] := '@'
      Else If (r = '-') Or (r = '') Then   {Quiere que sea una casilla vacia}
        tablero[i,j] := '-'
      Else
        Begin
          Repeat
            Gotoxy(2,23);
            Write('El caracter introducido no es valido, escriba otro.');
            Gotoxy(23,22);  {Limpiamos la respuesta anterior}
            LimpiaLinea;
            Gotoxy(23,22);
            LeeCaracter(r);
            Gotoxy(2,23);   {Limpiamos: "El caracter introducido..."}
            Write(' ':51)
          Until (r = '@') Or (r = '-') Or (r = '') Or (r = '*');
        End;
    End; {Reg}
  End; {LeeCaracter}

 Var x: Byte; {Para mostrar por pantalla los datos}

 Begin {DatosManual}
  With Reg Do
   Begin
    ClrScr;
    {Mostramos el tablero vacio (solo mostramos las coordenadas)}
    Gotoxy(3,1);
          Write('ABCDEFGHIJKLMNOPQRST');
          For x := 1 To DimH Do
            Begin
              Gotoxy(1,x+1);
              Write(x)
            End;
    {Pedimos que vaya rellenando el tablero}
    Gotoxy(1,18);
    Writeln;
    Writeln(' Introduzca los datos del tablero:');
    Writeln(' Si desea que aparezca una celula introduzca "*" o "@".');
    Writeln(' Pulse "intro" o escriba "-" para indicar que no hay celula.');
    Writeln(' Coordenadas: ');
    For i := 1 To DimH Do
      For j := 1 To DimV Do
        Begin
          Gotoxy(15,22);
          Write('(',i,',',j,'):  '); { (i,j): }
          Gotoxy(23,22);
          LimpiaLinea;
          Gotoxy(23,22);
          LeeCaracter(resp);
          Gotoxy(j+2,i+1);     {Posicion en la pantalla de cada casilla}
          Write(tablero[i,j]); {Respuesta del usuario, si es o no una celula}
        End;
  End; {Reg}
  Gotoxy(2,24);
  Write('Pulse "enter" para continuar: ');
  Readln(resp)
 End; {DatosManual}

{-------------------------------------------------------------------------------------}

Procedure VaciaTablero(Var Reg: RegTablero);
{
 Entrada: Registro donde se encuentra el tablero.

 Funcion: Dejar el tablero sin ningun caracter.

 Salida:  Tablero del registro dado, vacio.
}
 Begin
  For i := 1 To DimH Do
      For j := 1 To DimV Do
          Reg.tablero[i,j] := '-';  {Vaciamos la casilla}
```

```
 End; {VaciaTablero}

{-------------------------------------------------------------------------------------}

Procedure MuestraTablero(Reg: RegTablero);
{
 Entrada: Registro donde se encuentra el tablero.

 Funcion: Mostrar el tablero por pantalla con sus correspondientes celulas.

 Salida:  Se muestra el tablero por pantalla.
}
 Begin
   With Reg Do
    Begin   {Mostramos una linea encima del tablero indicando de que tipo es}
       Gotoxy(5,3);
       Case FormaTab Of
         '1': Write('Tablero Plano');       {Si el tablero es plano...}
         '2': Write('Tablero Cilindrico');  {Si el tablero es cilindrico...}
         '3': Write('Tablero Toroidal');    {Si el tablero es toroidal...}
       End; {case}

     Gotoxy(5,5);                     {Vamos a mostrar el tablero}
     Write('ABCDEFGHIJKLMNOPQRST');   {Letras que nos valen como coordenadas}
     For i := 1 To DimH Do
       Begin
         Gotoxy(3,5+i);               {Escribimos las coordenadas de las filas}
         Write(i);                    {Valor de i que nos vale como coordenada}
         Gotoxy(5,i+5);
         For j := 1 To DimV Do        {Escribe las casillas del tablero}
           Write(tablero[i,j]);
         Writeln;
       End;
    End; {RegisTab}
 End; {MuestraTablero}

{-------------------------------------------------------------------------------------}

Procedure Opciones(generaciones: Word);
 {
  Salida: Mostramos las opciones del tablero:

  Funcion:
  Se muestran las siguientes opciones por pantalla al lado el tablero:
  El usuario elegira como entrada de datos las letras correspondientes:
   E/e: Editar el tablero manualmente.
   G/g: Guardar la configuarcion a fichero.
   "enter": Calcular nuevas generaciones.
   T/t: Terminar el programa.

  Entrada: Letra (opcion) elegida por el usuario.
 }
 Begin
   Gotoxy(30,3);
   Write('Opciones del juego:');
   Gotoxy(27,5);
   Write('* Editar de forma manual el tablero: (E/e)');
   Gotoxy(27,6);
   Write('* Pulse "enter" para calcular las generaciones.');
   Gotoxy(27,7);
   Write('* Guardar la configuracion en un fichero: (G/g)');
   Gotoxy(27,8);
   Write('* Terminar el juego. (T/t)');
   Gotoxy(27,9);
   {Vamos mostrando cuantas generaciones se han calculado}
   If generaciones = 1 Then
     Write('* Se ha calculado 1 generacion.       ')             {Singular}
   Else
     Write('* Se han calculado ',generaciones,' generaciones.'); {Plural}
 End; {Opciones}

{-------------------------------------------------------------------------------------}

Procedure Plano(Var Reg: RegTablero);
{
 Entrada: Registro donde se encuentra el tablero.
```

```
 Funcion: Obtener el numero de celulas que rodean a cada celula tomando un
          tablero plano.

 Salida: Contador (cont) con el numero de celulas que rodean a la celula.
}
 Begin
   With Reg Do
    Begin
     {Si esta entre las coordenadas del tablero plano}
     If (i <> 0) And (i <= DimH) And (j <> 0) And (j <= DimV) Then
       If tablero[i,j] = '@' Then
         cont := cont + 1 {Contamos las celulas que tiene alrededor}
    End;
 End; {Plano}

{-------------------------------------------------------------------------------------}

Procedure Cilindrico(Var Reg: RegTablero);
{
 Entrada: Registro donde se encuentra el tablero.

 Funcion: Obtener el numero de celulas que rodean a cada celula tomando un
          tablero cilindrico.
          Al ser cilindrico, la columna 1 esta comunicada con la ultima columna.
          Aqui las columnas que se comunican entre si van a tomar el valor 0 y
          DimV+1

 Salida: Numero de celulas que rodean a la celula.
}
 Begin
  With Reg Do
   Begin
    If (i > 0) And (i <= DimH) Then {Si esta entre las filas del tablero...}
      Begin
        If j = 0 Then {Columna 1 comunidada con Columna DimV}
          Begin
            If tablero[i,DimV] = '@' Then {ultima columna}
               cont := cont + 1
          End
        Else If j = (DimV + 1) Then {Columna DimV comunicada con Columna1}
          Begin
            If tablero[i,1] = '@' Then  {columna 1}
              cont := cont + 1
          End
        Else {No es ninguna casilla de los bordes del tablero}
          Plano(ConfigTab) {Lo tratamos como un tablero plano}
      End {If}
   End; {Reg}
 End; {Cilindrico}

{-------------------------------------------------------------------------------------}

Procedure Toroidal(Var Reg: RegTablero);
{
 Entrada: Registro donde se acumula el tablero

 Funcion: Obtener el numero de celulas que rodean a cada celula tomando un
          tablero toroidal.

 Salida: La variable cont toma el numero de celulas que rodean a la celula.
}
 Begin
   With Reg Do
    Begin
      If i = 0 Then        {Comunicamos la primera fila y la ultima}
        Begin
          If j = 0 Then
            Begin
              If tablero[DimH,DimV] = '@' Then  {Casilla inferior derecha}
                cont := cont + 1
            End
          Else if j = (DimV + 1) Then           {Columna derecha}
            Begin
              If tablero[DimH,1] = '@' Then     {Casilla inferior izquierda}
                cont := cont + 1
```

```
          End
        Else
          Begin
            If tablero[DimH,j] = '@' Then     {Fila inferior}
              cont := cont + 1
          End
      End
    Else If i = (DimH + 1) Then               {Comunica ultima fila y primera}
      Begin
        If j = 0 Then                         {Columna Inferior izquierda}
          Begin
            If tablero[1,DimV] = '@' Then     {Casilla superior derecha}
              cont := cont + 1
          End
        Else If j = (DimV + 1) Then           {Columna derecha}
          Begin
            If tablero[1,1] = '@' Then        {Casilla (1,1)}
              cont := cont + 1
            End
        Else
          Begin
            If tablero[1,j] = '@' Then        {Fila 1}
              cont := cont + 1
          End
      End
    Else
      Cilindrico(ConfigTab)
  End {Reg}
End; {Toroidal}

{-------------------------------------------------------------------------------------}

Procedure Celulas(Var Reg: RegTablero);
{
 Entrada: Registro donde se encuentra el tablero.

 Funcion: Se obtiene el estado de cada celula, contando el numero
          de celulas que la rodean. Se aplicaran las reglas de Conway
          a cada casilla.

 Salida: Generamos un nuevo tablero con la generacion siguiente.
}
 Var
    f,c: Byte; {Fila y columna del tablero}
 Begin
  With Reg Do
   Begin
     For f := 1 To DimH Do
       Begin
         For c := 1 To DimV Do
           Begin
             cont := 0;
             {Recorremos los casillas de alrededor en busca de celulas}
             For i := (f-1) To (f+1) Do    {De fila anterior a posterior}
               For j := (c-1) To (c+1) Do  {De columna anterior a posterior}
               {No contamos la casilla en la cual estamos mirando su estado}
                 If Not ((i = f) And (c = j)) Then
                   Begin
                     Case Formatab Of
                       '1': Plano(ConfigTab);       {Si es un tablero plano...}
                       '2': Cilindrico(ConfigTab);  {Si es cilindrico...}
                       '3': Toroidal(ConfigTab);    {Si es toroidal...}
                     End;
                   End;
           {Si la casilla que estamos analizando contiene una celula}
           If tablero[f,c] = '@' Then
             Begin
               If (cont > 3) Or (cont < 2) Then
                 TabAux[f,c] := '-'     {Muerte por superpoblacion o soledad}
               Else
                 TabAux[f,c] := '@'     {Dejamos la celula}
             End
           Else                         {Si la casilla NO contiene una celula}
             Begin
               If cont = 3 Then         {Si esta rodeada por 3 celulas...}
                 TabAux[f,c] := '@'     {Nace una celula}
```

```
              Else
                TabAux[f,c] := '-'     {Dejamos la celula}
            End
          End {For c}
        End {For f}
   End {Reg}
 End; {Celulas}

{-------------------------------------------------------------------------------------}

Procedure GuardaAFichero(Reg: RegTablero);
{
 Entrada: Registro donde se encuentra la configuracion del tablero

 Funcion: Guardar en un fichero la configuracion obtenida.

 Salida: Fichero con el nombre que elige el usuario.
}

 Var
    FichConfig: Text;  {Fichero en el que guardamos la configuracion}
    nombre: String;    {Nombre que asignamos al fichero}
 Begin
  With Reg Do
   Begin
     Gotoxy(27,13);
     Write('Introduzca el nombre del fichero.');
     Repeat
       Gotoxy(27,14);
       Write('Escriba la ruta si es necesario: ');
       Readln(nombre)                          {Nombre del fichero}
     Until nombre <> ''; {Es necesario que se introduzca algun fichero}
     Assign(FichConfig,nombre);
     Rewrite(FichConfig);
     {Recorremos el tablero y lo vamos escribiendo en el fichero}
     For i := 1 To DimH Do
       Begin
         For j := 1 To DimV Do
           Write(FichConfig,tablero[i,j]);
         Writeln(FichConfig)                {Pasamos a la siguiente fila}
       End;
     Gotoxy(27,15); {Mostramos un mensaje de que fue guardado con exito}
     Write('La configuracion fue guardada en: ',nombre);
     Close(FichConfig)
   End {Reg}
 End; {GuardaAFichero}

{-------------------------------------------------------------------------------}

Procedure EditaManual(Var Reg: RegTablero);
{
 Entrada: Recibimos el registro donde vamos a guardar los datos.

 Funcion: Que el usuario pueda cambiar las casillas del tablero.

 Salida:  Tablero con la nueva casilla modificada.
}

 Var
    columna: Char; {Columna del tablero, esta representada con un letra
                    entre la A y la correspondiente a DimV}
    n: String;     {Para comprobar que se introducen datos correctos}
    fila: String;  {Fila del tablero, representada por un numero entre 1 y DimH}
 Procedure MuestraFallo;
 {
  Funcion: Informar al usuario de que ha introducido unos datos incorrectos.
 }
  Begin
    Gotoxy(27,22);
    Write('Recuerde: 15x20!');
    Gotoxy(27,23);
    Write('No corresponde a ninguna posicion del tablero');
    Gotoxy(27,24);
    Write('Numeros para las filas, letras para las columnas');
    j := 0; {No es un valor valido, con j := 0 hacemos que se repita el bucle}
  End; {MuestraFallo}
```

```
 Begin {EditaManual}
  With Reg Do
   Begin
    Gotoxy(27,15);
    Write('Si selecciona una casilla con celula, esta morira,');
    Gotoxy(27,16);
    Write('si selecciona una vacia, nacera una celula.');
    Gotoxy(27,18);
    Write('Introduzca las coordenadas: ');
    Repeat
      {Limpiamos el espacio de pantalla que no nos sirve}
      Gotoxy(33,19);
      LimpiaLinea;
      Gotoxy(36,20);
      LimpiaLinea;
      {Mostramos las opciones por pantalla}
      Repeat
        Gotoxy(27,19);
        Write('Fila: ');      {Fila que en la que se encuentra la casilla}
        Readln(fila)
      Until fila <> '';       {Chequeamos que no se deje en blanco}
      Repeat
        Gotoxy(27,20);
        Write('Columna: ');   {Columna en la que se encuentra la casilla}
        Readln(n);            {Leemos la columna}
        If Ord(n[0]) = 1 Then {Si solo hay una letra}
          columna := n[1]     {La acumulamos}
        Else                  {Limpiamos el valor incorrecto}
          Begin
            Gotoxy(27,20);
            LimpiaLinea;
            MuestraFallo
          End
      Until (Ord(n[0]) = 1);  {No puede haber mas de una letra}

     If (length(fila) = 2) Then {Si la fila leida se encuentra entre 10 y 15..}
       i := (Ord(fila[2]) - 48) + ((Ord(fila[1]) - 48) * 10)    {La acumulamos}
     Else If (length(fila) = 1) Then  {La fila se encuentra entre 1 y 9...}
       i := (Ord(fila[1]) - 48)       {La acumulamos en i}
     Else
       i := 0; {No se ha introducido un valor correcto}

     If ((Ord(columna) >= Ord('A')) And (Ord(columna) <= Ord('T'))) Then
         j := Ord(columna) - 64 {Para las mayusculas. Convertimos a numero}
     Else If (Ord(columna) >= Ord('a')) And (Ord(columna) <= Ord('t')) Then
         j := Ord(columna) - 96 {Para las minusculas. Convertimos a numero}
     Else {Se han introducido unos datos erroneos}
       j := 0; {No se ha introducido un valor correcto}

     If Not ((i >= 1) And (i <= DimH) And (j >= 1) And (j <= DimV)) Then
     {Si no esta entre el rango de filas y columnas que pedimos lo decimos}
       MuestraFallo;
   {Hasta que este entre las coordenadas del tablero}
   Until (i >= 1) And (i <= DimH) And (j >= 1) And (j <= DimV);

   {Realizamos el cambio. Si habia una celula morira, y si estaba vacia
    nacera una celula nueva}
   If tablero[i,j] = '-' Then {Estaba vacia, nace una celula}
      tablero[i,j] := '@'
   Else
      tablero[i,j] := '-';    {Habia una celula, esta morira}

   {Limpiamos las lineas del tablero que no nos sirven}
   Gotoxy(35,11);
   LimpiaLinea;
   For i := 22 To 24 Do
     Begin
       Gotoxy(27,i);
       LimpiaLinea
     End;
  End {Reg}
 End; {EditaManual}

{-------------------------------------------------------------------------------}

Procedure MuestraOpcion;
```

```
{
 Funcion:  Pedir al usuario que introduzca su opcion.
           Puede calcular nuevas generaciones, guardar en un archivo,
           editar el tablero manualmente o terminar el programa.

 Salida:   Muestra por pantalla la palabra "opcion" y se dice si se ha
           introducido una respuesta correcta.
           La variable resp toma como valor la opcion elegida por el
           usuario (Calcular generacion, Grabar, Editar o Terminar).
}

 Procedure CompruebaResp (r: String);
 {
  Entrada: r: Variable de tipo string (es la respuesta del usuario).

  Funcion: Comprobar si la respuesta es valida. Solo puede ser:
           G/g, E/e, T/t o haberse pulsado unicamente enter.

  Salida: Si la respuesta es valida retornamos el control al programa, sino
          pedimos otra respuesta.
 }
  Begin
    If length(resp) > 1 Then {La respuesta no es valida}
      Begin
        Gotoxy(27,13);
        Write('La respuesta introducida no es valida.');
        Gotoxy(35,11);
        LimpiaLinea;
        Gotoxy(35,11);
        Readln(resp)         {Volvemos a pedir la respuesta}
      End
  End; {CompruebaResp}
 Begin {MuestraOpcion}
   Repeat
     Gotoxy(27,11);
     Write('Opcion:');
     LimpiaLinea;
     Gotoxy(35,11);
     Readln(resp);
     CompruebaResp(resp); {Comprobamos que no tenga mas de un caracter}
     If Not ((resp[1] In respuestas) Or (resp = '')) Then {Si no es correcta}
       Begin
         Gotoxy(27,13);
         Write('La respuesta introducida no es valida.')
       End;
   {Hasta que sea G/g, E/e, T/t o "enter"}
   Until (resp[1] In respuestas) Or (resp = '');
 End; {MuestraOpcion}

{-------------------------------------------------------------------------------------}

Procedure LimpiaOpcion;
{
 Funcion: Limpiar la pantalla de las opciones que no son necesarias
}
 Begin
   For i := 13 To 24 Do
        Begin
          Gotoxy(27,i);
          Write(' ':50);
        End;
 End; {LimpiaOpcion}

{-------------------------------------------------------------------------------------}

Begin {********************** PROGRAMA PRINCIPAL ************************** p.p}
  VaciaTablero(ConfigTab);  {Dejamos el tablero sin ninguna celula viva}

 Repeat   {Pedimos que nos diga como desea obtener los datos del tablero}
   ClrScr;
   Writeln;
   Writeln(' Elija su configuracion inicial: ');
   Writeln;
   Writeln(' 1) Leer los datos de un fichero.');           {Desde fichero}
   Writeln(' 2) Generar una configuracion aleatoria.');    {Aleatorio}
   Writeln(' 3) Leer los datos desde teclado.');           {Desde teclado}
```

```
   Write(' Introduzca su opcion: ');
   Readln(resp);
 Until (resp = '1') Or (resp = '2') Or (resp = '3');

  With ConfigTab Do
   Begin
    Repeat {Obtenemos como se quieren calcular las generaciones}
      Writeln;
      Writeln(' ¿Como desea que se calculen las generaciones?');
      Writeln(' 1 - Calcular generacion tras generacion.');
      Writeln(' 2 - Calcular un numero determinado de generaciones.');
      Write(' Elija 1 o 2, por favor: ');
      Readln(ModoGeneracion);
      If ModoGeneracion = '2' Then  {Si quiere calcular varias generaciones}
        Begin
          Repeat
            Write(' Introduzca el numero de generaciones a calcular: ');
            gen := NumCorrecto(Ngen);  {Chequeamos que introduzca un numero}
            gen := Ngen  {Guardamos la generacion en una variable para
                          mostrarlo cuando vayan pasando las generaciones}
          Until Ngen > 0
       End
     Else          {Para mostrar por pantalla el inicio de las generaciones}
       gen := 0;
     ClrScr
    Until ModoGeneracion In ['1','2']
   End; {ConfigTab}

  Case resp[1] Of
   '1': LeerTablero(ConfigTab);       {Leemos el tablero del fichero}
   '2': TabAleatorio(ConfigTab);      {Generamos un tablero aleatorio}
   '3': DatosManual(ConfigTab)        {El tablero se genera manualmente}
   {Solo puede valer 1, 2 o 3 porque ya lo hemos chequeado antes}
  End;

  Repeat {Forma del Tablero}
    ClrScr;
    Writeln(' Elija el numero del tipo de tablero que prefiere: ');
    Writeln(' 1 - Plano.');
    Writeln(' 2 - Cilindrico.');
    Writeln(' 3 - Toroidal.');
    Write(' Introduzca su respuesta: ');
    Readln(ConfigTab.FormaTab)
  Until ConfigTab.FormaTab In ['1','2','3'];

  ClrScr;
  ConfigTab.gen := 0; {No hemos calculado ninguna generacion, lo inicializamos}
  resp := '';
  MuestraTablero(ConfigTab);   {Mostramos el tablero inicial}

 Repeat
  With ConfigTab Do
   Begin
    If ModoGeneracion = '2' Then
    {Se ha pedido calcular un numero determinado de generaciones}
      Begin
        Gotoxy(27,5);
        Write('Pulse enter para calcular las generaciones: ');
        MuestraOpcion;
        gen := Ngen;
        For Ngen := Ngen Downto 1 Do
          Begin
            Celulas(ConfigTab);    {Calculamos la siguiente generacion}
            tablero := TabAux      {Acumulamos la generacion}
          End;
        If (resp = 'G') Or (resp = 'g') Then
          GuardaAFichero(ConfigTab)
        Else If (resp = 'E') Or (resp = 'e') Then
          EditaManual(ConfigTab);

        MuestraTablero(ConfigTab);
        Opciones(gen);
        ModoGeneracion := '1';    {Para calcular las generaciones de una en una}
      End; {If}

    Opciones(gen);                {Menu de opciones del programa}
```

```
    MuestraOpcion;

    If ModoGeneracion = '1' Then  {Calcular generacion tras generacion}
      Begin
        LimpiaOpcion;             {Limpiamos las opciones que no son necesarias}
        If resp = '' Then         {Se quiere calcular una nueva generacion}
          Begin
            celulas(ConfigTab);   {Calculamos la siguiente generacion}
            tablero := TabAux;    {Acumulamos la generacion}
            gen := gen + 1        {Aumentamos la generacion}
          End
        Else
          Begin
            If (resp = 'G') Or (resp = 'g') Then
              GuardaAFichero(ConfigTab)
            Else If (resp = 'E') Or (resp = 'e') Then
              EditaManual(ConfigTab)
          End;
        MuestraTablero(ConfigTab)
      End {If}
   End; {ConfigTab}
 Until (resp = 'T') Or (resp = 't')
End. {********************* PROGRAMA PRINCIPAL ************************** p.p}
```

Las prácticas y el proyecto no están corregidos integramente por un profesor, lo que sí está garantizado es que tienen una calificación de notable. Si pueden hacer algún comentario que sirva para mejorar los ejercicios, por favor, háganlo.

Hemos llegado al final, sdo me gustaría pediros una cosa, si distribuís estos ejercicios entre vuestros amigos o a través de Internet, por favor, dejar mi nombre, me ha costado un gran trabajo.

Está permitida su distribución para fines educativos, nunca lucrativos.

Por favor, escribir ante cualquier duda o comentario que sirva para mejorar los ejercicios.

¡¡¡ Ánimo !!!

202080
MADRID (ESPAÑA)