

PASCAL

Nociones básicas

- Todas las sentencias deben terminar en punto y coma (;) excepto la última sentencia que termina con punto (.)
- Los programas en Pascal se componen de:
 - Cabecera:
PROGRAM identificador;
 - Declaraciones
CONST
 Iden_cte = valor_cte;
VAR
 Iden_var : tipo_var;
 - Sentencias
 - Se pueden incluir **comentarios** en cualquier parte de un programa pero deben delimitarse con llaves o con paréntesis y asterisco:
{ esto es un comentario }
(* y esto también *)

PASCAL

2

Variables y Constantes

- Los identificadores son nombres para referenciar variables, constantes y funciones. Se forman con letras (A-Z), números (0-9) y símbolo de subrayado (_) pero deben comenzar con una letra. Pueden tener cualquier longitud pero sólo se reconocen los 63 primeros caracteres
- Los tipos de datos básicos utilizados en Pascal son:
 - INTEGER: Entero
 - REAL: Real
 - CHAR: Carácter
 - BOOLEAN: Lógico/Booleano
 - STRING: Cadena de caracteres
- En Pascal, las variables deben declararse antes de ser utilizadas en una sección de declaración de variables dentro del programa. Esta sección comienza con la palabra VAR
- Las constantes también deben inicializarse en la sección de declaración de constantes. Esta sección comienza con la palabra CONST

PASCAL

3

Estructura básica de un programa en Pascal

```
PROGRAM nombre;  
CONST  
    nom_cte1 = valor1;  
    .....  
    nom_cteN = valorN;  
VAR  
    nom_var1 : tipo1;  
    .....  
    nom_varN : tipoN;  
BEGIN  
    .....  
    sentencias;  
    .....  
END.
```

PASCAL

4

Estructuras secuenciales

- Operación de asignación
var := exp ;
- Operación de Entrada/Lectura:
READ(lista parámetros);
READLN(lista parámetros);
- Operación de Salida/Escritura:

Estructuras condicionales

```
IF condición THEN  
BEGIN  
    Sentencias en caso verdadero  
END  
[ELSE  
BEGIN  
    Sentencias en caso falso  
END  
];
```

- Si el grupo 'Sentencias en caso verdadero' (y/o 'Sentencias en caso falso') está formado por una única sentencia, el par BEGIN-END no es necesario



CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENT
CALL OR WHATSAPP:689 45 44 70

Estructuras repetitivas

- Bucle FOR
FOR var: =inicio TO/DOWNTO fin DO
BEGIN
Cuerpo del bucle
END;
- Bucle WHILE
WHILE condición DO
BEGIN
Cuerpo del bucle
END;
- Bucle REPEAT-UNTIL
REPEAT
Cuerpo del bucle
UNTIL condición;
- Si 'cuerpo del bucle' (en la sentencia FOR y WHILE) está formado por una única sentencia, el par BEGIN-END no es necesario
- Nótese que la sentencia REPEAT-UNTIL no necesita un par BEGIN-END puesto que el cuerpo del bucle queda perfectamente delimitado con REPEAT al principio y UNTIL al final

PASCAL

7

Otros temas

- Para dar formato de salida a los números reales:
`WRITE(exp_real:n_col:n_dec)`
donde `n_col` es el número de columnas/posiciones que va a ocupar el número y `n_dec` el número de decimales que queremos mostrar en pantalla
- Para hacer uso de 'unidades' (librerías con funciones y procedimientos predefinidos) se utiliza la siguiente sentencia entre la cabecera y la sección de declaración del programa:

```
PROGRAM nombre;  
USES unidad;  
CONST  
.....
```

PASCAL

8

Declaración de funciones y procedimientos

```
FUNCTION ident_func (parámetros formales):tipo;  
{Declaración variables locales}  
BEGIN  
Sentencias;  
ident_func:=expresión;  
END;
```

```
PROCEDURE ident_proc (parámetros formales);  
{Declaración variables locales}  
BEGIN  
Sentencias;  
END;
```

PASCAL

9

Sección TYPE

- Se utiliza para declarar nuevos tipos de datos o renombrar tipos ya definidos

```
TYPE  
nombre-1 = tipo-1;  
nombre-2 = tipo-2;  
.....  
nombre-N = tipo-N;
```

- tipo-i es un tipo de datos predefinido o definido por el usuario

PASCAL

10

Uso de Unidades

- Las unidades se utilizan para crear librerías de funciones y procedimientos que permitan la programación modular
- Además de las unidades diseñadas por el propio programador, Pascal incluye sus propias unidades que permiten la utilización de ciertas funciones y procedimientos generales, de interés para el programador que nos facilitan el trabajo y mejoran la apariencia final de cualquier aplicación
PROGRAM nombre;

Unidad System

- **ABS(X)**: Valor absoluto de X
- **ARCTAN(X)**: Arcotangente de X
- **CHR(X)**: Carácter cuyo número ordinal es X
- **CONCAT(S1..Sn)**: Concatena en una única cadena todas las cadenas que toma como argumento
- **COPY(S,X,Y)**: Devuelve una subcadena de longitud Y comenzando en la posición X de la cadena S
- **COS(X)**: Coseno de X
- **EXP(X)**: Número e elevado a X
- **LENGTH(S)**: Número de caracteres de la cadena S
- **LN(X)**: Logaritmo Neperiano de X
- **ODD(X)**: Devuelve TRUE si X es impar
- **POS(S1..S2)**: Devuelve la posición del primer carácter de S1 en S2, siempre que S1 se encuentre

Cartagena99

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENT
CALL OR WHATSAPP:689 45 44 70

Ejemplos

Generación de números aleatorios

```
PROGRAM aleatorio;
VAR
R: INTEGER;
BEGIN
RANDOMIZE;
REPEAT
R:=RANDOM(10);
WRITELN (R);
UNTIL (R=0);
END.
```

Convertir a mayúsculas

```
PROGRAM mayus;
VAR
s : STRING;
i : INTEGER;
BEGIN
WRITE('Introduzca una cadena
de caracteres: ');
READ(s);
FOR i := 1 TO LENGTH(s) DO
s[i] := UPCASE(s[i]);
WRITELN('La cadena en
mayúsculas es : ',s);
END.
```

PASCAL

13

Unidad Crt

- **CLRSR**: Limpia la pantalla dejando el curso en la esquina superior izquierda de la pantalla.
- **DELAY(X)**: Espera X milisegundos antes de empezar a ejecutar la siguiente sentencia (realiza una pausa de X milisegundos).
- **GOTOXY(X,Y)**: Sitúa el cursor en la columna X y fila Y. El valor de X varía de 1 a 80 y el de Y de 1 a 24.
- **KEYPRESSED**: Devuelve TRUE si se ha pulsado alguna tecla del teclado y FALSE en otro caso.
- **HIGHVIDEO**: Selecciona el modo de alta intensidad de caracteres.
- **LOWVIDEO**: Selecciona el modo de baja intensidad de caracteres.
- **NORMVIDEO**: Selecciona el modo normal de intensidad de caracteres.
- **NOSOUND**: Desactiva un sonido realizado con SOUND.
- **SOUND(X)**: Emite un sonido de X hertzios de frecuencia. El sonido continúa hasta que se desactiva con NOSOUND.
- **TEXTBACKGROUND(X)**: Inicializa el color de fondo al color X. X es una expresión que devuelva un valor entre 0 y 7, correspondiente con los valores de las ocho primeras constantes de color.
- **TEXTCOLOR(X)**: Inicializa el color del texto al color X. X es una expresión que devuelva un valor entero.
- **READKEY**: Lee un único carácter del teclado.
- **WHEREX**: Devuelve la coordenada x de la posición actual del cursor.
- **WHEREY**: Devuelve la coordenada y de la posición actual del cursor.

PASCAL

14

Ejemplos

Colores del texto

```
PROGRAM colores;
USES Crt;
VAR i: INTEGER;
BEGIN
FOR i:=0 TO 31 DO
BEGIN
TextColor(i);
TextBackground(0);
WRITELN('Color del texto ',i);
READKEY;
END;
NORMVIDEO;
WRITELN('Vuelve a texto normal');
END.
```

Uso de Keypressed

```
PROGRAM aleatorio2;
USES Crt;
VAR
R: INTEGER;
BEGIN
RANDOMIZE;
REPEAT
R:=RANDOM(1001);
WRITELN (R);
UNTIL KEYPRESSED;
END.
```

PASCAL

15

Tipos de datos estructurados

- Arrays
 - Declaración

```
Ident_array:ARRAY[inicio..fin] OF tipo;
```
 - Ejemplos

```
Vector:ARRAY[1..50] OF INTEGER;
Matriz:ARRAY[1..4] OF ARRAY[1..6] OF REAL;
```
- Registros
 - Declaración:

```
Ident_reg: RECORD
lista de campos
END;
```
 - Ejemplos

```
empleada: RECORD
nombre:STRING[30];
sexo:CHAR;
edad:INTEGER;
salario:REAL;
casado:BOOLEAN;
END;
```
- Instrucción WITH para registros

```
WITH nombre_reg DO
BEGIN
instrucciones
END;
```

Ejemplo: la siguiente instrucción WITH WITH empleado DO

```
BEGIN
nombre:='Alma Caro Ruiz';
sexo:='M';
edad:=35;
salario:=1250;
casado:=F;
END;
```

Es equivalente a

```
empleado.nombre:='Alma Caro Ruiz';
empleado.sexo:='M';
empleado.edad:=35;
empleado.salario:=1250;
empleado.casado:=F;
```

PASCAL

16

Ficheros

- Ficheros de texto
 - Declaración

```
vfic:TEXT;
```
 - Se puede utilizar el concepto de línea (EOLN, READLN, WRITELN)
 - Los elementos del fichero son caracteres

Operaciones sobre ficheros

- Asignación `Assign(vfic,cadena);`
- Apertura para lectura `Reset(vfic);`
- Creación para escritura `Rewrite(vfic);`
- Apertura para escritura `Append(vfic);` (solo ficheros de texto)
- Lectura de datos `Read(vfic,l-parámetros);`
- Escritura de datos `Write(vfic,l-parámetros);`

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENT
CALL OR WHATSAPP:689 45 44 70



Ejemplo de uso de ficheros de texto

```
PROGRAM BorradoEspacio;
CONST
  Espacio = ' ';
VAR
  ArchEntrada, ArchSalida : text;
  Car : char;
  PrevioNoEspacio : boolean;
  ActualNoEspacio : boolean;
BEGIN
  assign(ArchEntrada,'entrada.txt');
  assign(ArchSalida,'salida.txt');
  reset (ArchEntrada);
  rewrite(ArchSalida);
  WHILE NOT eof ( ArchEntrada ) DO
    BEGIN
      PrevioNoEspacio := false;
      WHILE NOT eoln( ArchEntrada ) DO
        BEGIN
          read( ArchEntrada, Car );
          ActualNoEspacio := Car <> Espacio;
          IF ActualNoEspacio OR
             PrevioNoEspacio THEN
            write ( ArchSalida, Car );
          PrevioNoEspacio := ActualNoEspacio;
        END ( WHILE NOT eoln);
      END ( WHILE NOT eof);
      close(ArchSalida);
    END.
  END.
```

PASCAL

19

Ejemplo de uso de ficheros binarios

```
PROGRAM crea_fic :
TYPE
  TEmpleado= RECORD
    Codigo:Integer;
    Nombre:string(30);
    Edad:Integer;
    sueldo:real;
  END;
  TF_emp=FILE OF TEmpleado;
VAR
  emp:TEmpleado;
  F_emp:TF_emp;
BEGIN
  assign(F_emp,'empleado.dat');
  rewrite(F_emp);
  WITH emp DO
  BEGIN
    WRITE('Introduzca código de
    empleado (0 para terminar)');
    READLN(codigo);
    WHILE codigo >0 DO
      BEGIN
        WRITE('Introduzca nombre del empleado');
        READLN(nombre);
        REPEAT
          WRITE('Introduzca edad del empleado');
          READLN(edad);
        UNTIL (edad>18) AND (edad<70);
        WRITE('Introduzca sueldo del empleado');
        READLN(sueldo);
        WRITE(F_emp,emp);
        WRITE('Introduzca código de empleado
        (0 para terminar)');
        READLN(codigo);
      END;
    END: (* Fin del WITH *)
    WRITELN('FICHERO DE EMPLEADOS CREADO');
    CLOSE(F_emp);
  END.
```

PASCAL

20

Control de Errores de Entrada/Salida

- Directiva {\$I-} y {\$I+}
- Por defecto la directiva Input/Output Checking está activada lo que cualquier error en un procedimiento de entrada/salida causa la terminación inmediata del programa visualizando el error cometido
- Se desactiva para pasar el control de errores de entrada/salida al usuario. En este caso es el usuario el encargado de gestionar los errores producidos por los procedimientos de I/O
- Para ello, se consulta el valor de la función IOResult que devuelve el error cometido o cero si no se ha producido error

PASCAL

21

Ejemplo de uso de directivas I/O

```
PROGRAM consulta_fic :
uses crt;
TYPE
  TEmpleado= RECORD
    Codigo:Integer;
    Nombre:string(30);
    Edad:Integer;
    sueldo:real;
  END;
  TF_emp=FILE OF TEmpleado;
VAR
  emp:TEmpleado;
  F_emp:TF_emp;
BEGIN
  clrscr;
  assign(F_emp,'empleado.dat');
  {$I-}
  reset(F_emp);
  IF IOResult <> 0 THEN
    writeln ('ERROR AL ABRIR FICHERO')
  ELSE
    BEGIN
      WHILE NOT EOF(F_emp) DO
        BEGIN
          READ(F_emp, emp);
          WRITELN('Código: ',emp.codigo);
          WRITELN('Nombre: ',emp.nombre);
          WRITELN('Edad: ',emp.edad);
          WRITELN('Sueldo: ',emp.sueldo:0:2);
          WRITELN;
          readkey;
        END;
      WRITELN('* FIN DE LA CONSULTA*');
      CLOSE(F_emp);
    END;
  {$I+}
  END.
```

PASCAL

22

Tipos numéricos

TIPOS ENTEROS

TIPO	RANGO	BYTES
Shortint	-128..127	8bits, con signo
Integer	-32768..32767	16 bits, con signo
Longint	-2147483648..2147483647	32 bit, con signo
Byte	0..255	8 bits, sin signo
Word	0..65535	16 bits, con signo

TIPOS REALES

CLASES PARTICULARES, TUTORÍAS TÉCNICAS ONLINE
 LLAMA O ENVÍA WHATSAPP: 689 45 44 70

ONLINE PRIVATE LESSONS FOR SCIENCE STUDENT
 CALL OR WHATSAPP:689 45 44 70

