PROGRAMACION PASCAL

Como hemos visto en un algoritmo describimos un conjunto de operaciones que debe realizar el computador , para que esas instrucciones las ejecute el computador debemos utilizar un lenguaje de computación que para nuestro caso vamos a ver que requerimos para escribir instrucciones básicas en PASCAL.

- ***PARTES DE UN PROGRAMA EN PASCAL***

Un programa en PASCAL se caracteriza por tener la siguientes partes que definen su estructura:

PROGRAM <Identificador> ; { Zona de Encabezamiento }

USES <declaración de unidades>;

LABEL

<declaración de etiquetas>

CONST

<definición de constantes>

TYPE

<declaración de tipos>

VAR

<declaración de variables>

< declaración del procedimiento>

<declaración de la función>

BEGIN

.....

{ cuerpo del programa }

{ Instrucciones del Programa }

.....

END.

- ***ZONA DE ENCABEZAMIENTO :***

En ella le damos el nombre al programa. En todo programa debe haber una zona de encabeza-miento.

La zona de encabezamiento se inicia con la palabra **PROGRAM;** seguida por el nombre con que identificamos nuestro programa.

EJEMPLO:

En nuestro primer programa el encabezado termina es: **PROGRAM** EJEMPLO**;**

- ***ZONA DE DECLARACIONES:***

En ella se declaran los objetos con que un programa trabaja, en ella se pueden declarar: **variables, tipos de datos, constantes, label, procedimientos y funciones**.

Un programa en PASCAL no necesita tener todas estas partes, y al declarlas pueden ser escritas en cualquier orden; se recomienda sin embargo seguir el orden propuesto arriba.

Durante el curso veremos en detalle cada una de las partes que conforman la zona de declaraciones y su utilidad.

- ***DECLARACION DE UNIDADES:***

El Turbo PASCAL le entrega al programador unas biblioteca con funciones y procedimientos, que le permiten realizar distintos tipo de tareas, las cuales vienen en archivos llamados UNIDADES.

Cuando un usuario desea utilizar alguna función o procedimiento de alguna de esas unidades, debe declarar la unidad (biblioteca) que contiene dicha función o procedimiento antes de poder hacer uso de ella en el programa principal.

Los programadores también pueden construir sus propias unidades.

Mas adelante estudiaremos en detalle las unidades y las funciones que suministra.

Las unidades que suministra Turbo Pascal son: DOS, WINCRT, PRINTER, GRAPH, GRAPH3.

EJEMPLO: Si en un programa tiene :

USES WINCRT, DOS;

En el momento de la escritura del programa, se podran hacer uso de los procedimientos o funciones que formen dichas unidades.

- ***DECLARACION DE CONSTANTES:***

Esta zona se caracteriza por iniciar con la palabra **CONST** seguido por los identificadores de sitios en memoria que van almacenar datos CONSTANTES para ese programa, la zona de declaración de constantes es opcional.

Ejemplo: El siguiente es un ejemplo de una declaración de constantes:

CONST

Centimetros = 100 ;

Gravedad = 9.8 ;

Mensaje = 'El Sistema no tiene Solución' ;

Respuesta = 'S' ;

Gravedadina = Centimetros*100 ;

Valordolar **: REAL** = 368.9;{**Constante con Tipo**}

En el momento de encontrar estas declaraciones, el compilador de PASCAL crea en memoria sitios que se van a identificar con esos nombre y mientras se ejecute el programa tomarán el valor que se expresa a la derecha de la declaración. Ninguno de los valores asignados a una constante se pueden cambiar en un programa. **Excepto en el caso de las constantes declaradas con tipo**.

- ***DECLARACION DE VARIABLES :***

Una variable, es el objeto de un programa que puede cambiar su valor durante la ejecución.

En la realidad, una variable es una celda de memoria conformada por uno o más bytes a la cual le asignamos un **nombre para identificarla y es el lugar donde durante la ejecución de un programa** almacenamos **un dato**.

Le decimos, al computador que variables tiene el programa, declarándolas en la zona que comienza con la palabra **VAR**.

El **computador sabe cuantos bytes conforman una variable en memoria *por el tipo(dominio)* *de variable*** que se le asigna en la declaración de ella.

Todas las variables que maneja un programa deben ser declaradas.

EJEMPLO : El siguiente es un ejemplo de una zona de declaración de variables en un programa en PASCAL.

VAR

horas **:INTEGER** ;

sal_total **:REAL ;**

mensaje **:STRING[30];**

respues **:CHAR;**

Factorial **:LONGINT;**

Estado **:BOOLEAN;**

- **TIPOS DE VARIABLES EN PASCAL:**

Los tipos de variable en PASCAL le permiten al compilador reservar el espacio en bytes necesario para almacenar un determinado dato, también le sirven para determinar durante la ejecución de un programa que

datos se pueden almacenarse en dicha variable.

- ***TIPOS DE VARIABLE NUMERICA:***

Para muchos programas se necesitan variables que sean capaces de almacenar números, PASCAL nos permite declarar distintos tipos de variables numéricas, algunos son:

- **VARIABLE TIPO INTEGER :**

Una variables de tipo INTEGER puede manipular números entre **−32768..32767** ocupando **dos bytes** para representarlos.

EJEMPLO: Un ejemplo de como se declaran variables enteras es:

VAR

dato, mes :**INTEGER**;

- **VARIABLE TIPO LONGINT :**

Una variable de tipo LONGINT puede manipular numeros entre −2,147,483,648 .. 2,147,483,647 ocupando 4 byte para almacenarlos.

EJEMPLO: Un ejemplo de como se declaran variables **LONGINT** es:

VAR

ferma, y :**LONGINT**;

- **DATOS TIPO REAL :**

Diremos que un dato es de tipo real cuando necesitamos manipular con el datos como **2.5, 0.012, 34.56** etc.

PASCAL reconoce los siguientes tipos datos como reales:

- **VARIABLES TIPO REAL:**

Puede puede manejar, datos numéricos llamados de punto flotante, que se caracterizan por tener mantisa (parte fraccionaria) multiplicada por una potencia de 10. El número de dígitos que maneja en su mantisa se llama cifras significativas una variable tipo real puede manejar 11 cifras significativas en su mantisa y el rango de exponentes entre:

10−39 .. 10+38, para su representación real ocupa 6 bytes de memoria.

El rango de números es: 2.9*****10−39 .. 1.7*****1038

- ***VARIABLES DE TIPO CARACTER:***

En muchos programas necesitamos tener sitios en memoria que puedan almacenar datos que son un solo carácter como 'A', 'a',....,'Z','z','0'...'9' o secuencias de caracteres como 'NANA'o 'Calle 40 d 30−200' para esos casos necesitamos variables que sean de tipo CHAR o STRING.

- **VARIABLE TIPO CHAR :**

Una variable de tipo **CHAR** es aquella que va almacenar datos que están en el conjunto de los caracteres ASCII de un computadora.

Una variable CHAR se gasta un byte para la representación en memoria de uno de ellos.

Utilice estos tipos de variables cuando necesite en un programa almacenar una letra del alfabeto, o un carácter como #, $, %, &, *, + etc, en general cualquier carácter de la tabla ASCII.

EJEMPLO: El siguiente es un ejemplo de la declaración de dos variables CHAR.

VAR

letra, resp :**CHAR;**

- **VARIABLE DE TIPO STRING :**

Una variable de tipo String puede almacenar una cadena de caracteres.

Se gasta tanto espacio en memoria como caracteres se quiera que el computador pueda almacenar de la cadena.

EJEMPLO: El siguiente es un ejemplo de la declaración de dos variables STRING.

VAR

nombre :**STRING[10];**

ciudad :**STRING[7];**

El diagrama muestra como estan las variables despues de una operación de lectura o de asignación como la siguiente.

nombre:=' LYNA JUAN ';

ciudad:='MEDALLO';

- **VARIABLES DE TIPO BOOLEAN :**

Una variable de tipo BOOLEAN es aquel que puede almacenar en memoria los valores **TRUE**, **FALSE**.

Una variable BOOLEAN no se puede utilizar en instrucciones de lectura como READ o READLN.

- **IDENTIFICADORES:**

Son los nombres con los cuales identificamos los objetos de un programa como :variables, constantes, funciones, tipos, procedimientos, unidades etc.

Un identificador en el PASCAL es una secuencia de caracteres que puede ser de cualquier longitud **pero solo lo primeros 63** primeros caracteres son significativos.

Se construyen a gusto del programador y siguiendo la siguiente reglas:

1. Deben comenzar con una letra de la (A a Z), mayúscula o minúscula).

2. No es permitido el carácter blanco como parte de un identificador

3. No se pueden usar palabras reservadas como: PROGRAM, USES, VAR para identificar.

4. Letras, dígitos y carácter _ subrayado son permitidos sólo después del primer carácter del identificador.

- ***COMO SE ENTRAN DATOS A UNA VARIABLE DE MEMORIA ?:***

Hasta el momento sabemos definir los espacios de memoria para almacenar información, pero no hemos dicho como podemos almacenar datos en esos sitios.

Existen dos métodos para entrar datos a una variable en memoria:

1. Usando los procedimientos READ y READLN.

2. Por medio de la operación de asignación.

- **PROCEDIMIENTOS READ, READLN :**

Estos procedimientos nos permiten entrar uno o más datos desde el teclado para ser almacenadas en alguna celdas en memoria.

SINTAXIS :

READ(<lista de variables>**);**

READLN(<lista de variables>**);**

Donde la lista de variables está conformada por el nombre de una o mas variables separadas por comas, para las cuales deseamos entrar datos usando el teclado.

Para cuales variables debemos entrar los datos por el teclado ? para todas aquellas que en el proceso de análisis del problema, veamos que son indispensables para la realización de la taraea que el computador va a realizar y el no puede conocerlos por medios de cálculos o lectura de algun dispostivo de almacenamiento .

EJEMPLOS: Si supone, que quiere entrar los valores a las variables declaradas en un programa podría hacer lo siguiente :

READ(a,b);

READLN(a,b,d,e);

READLN(f);

En un READ o READLN, no se puede leer datos para variables lógicas.

- **INSTRUCCION DE ASIGNACION :**

Es la operación mediante la cual se le asigna un valor determinado a una variable en memoria.

SINTAXIS:

<Identificador> := <expresión>

Con toda expresión de la forma anterior le estaremos indicando al computador que: evalue la expresón y el resultado lo almacene en la variable que se identifica por el identificador.

Donde una expresión puede ser un valor constante, o fórmula matemática.

La sintaxis expresa que: **el valor de la expresión debe ser almacenado en el sitio de memoria identificado, por el identificador**.

EJEMPLO : Los siguientes son instrucciones de asignación validas en PASCAL.

Almacene en la variable **horas** el valor 30, se escribe en PASCAL así:

Horas :=30;

Almacene el valor 5000 en la variable que en memoria se llama **salario.**

Salario := 5000;

Agreguele 1000 al valor que existe en **salario** y lo que le de almacenelo en **salario**

salario := salario + 1000;

Después de la acción anterior la variable **salario tiene un valor de 6000 y el valor 5000 se perdió.**

Multiplique por 2 el valor que se encuentra en **salario**, y almacene el resultado en **sal_total**

sal_total **:=** salario*****2 **;**

Después de la asignación anterior en la variable **sal_total se almacena el valor de 12000**

Almacene True en la variable **estado que debe ser de tipo BOOLEAN**

estado**:=** TRUE**;**

Almacene la cadena 'NACIONAL LE GANA A EL DIM' en la variable **mensaje** que debe ser **declarada de tipo STRING**:

mensaje **:=** 'NACIONAL LE GANA A EL DIM' **;**

De los ejemplos anteriores se debe tener presente los siguientes aspectos:

Cuando vamos asignar valores a una variable numérica no se encierra entre comillas el valor que define la expresión, ejemplo:

Salario := 5000;

Cuando vamos asignar valores a una variable de tipo char o string se encierra entre comillas el valor que define la expresión, ejemplo:

mensaje **:=** 'NACIONAL LE GANA A EL DIM' **;**

Nunca! se escribe l a expresión a la izquierda del operador de asignación.

Ejemplo:

'**NACIONAL LE GANA A EL DIM' :=mensaje;**

salario*****2 **:=**sal_total**;**

- **PROCEDIMIENTOS DE SALIDA (WRITE, WRITELN):**

Permite escribir mensajes y/o valores en pantalla. Los valores deben estar almacenados en memoria en alguna variable.

SINTAXIS :

WRITE(<item(s)> **);**

WRITELN(<item(s) > **);**

Donde el item puede ser:

El identificador de una variable : en cuyo caso el computador escribe en pantalla el valor que contenga la variable en el momento de la ejecutar la instrucción.

Ejemplo:

WRITELN(a);

Una cadena de caracteres encerrada entre comillas simples : en cuyo caso el computador escribe todo lo que encuentre entre las comillas, textualmente en la pantalla.

WRITELN(´La Vida es una ruleta ´ **) ;**

Una expresión matemática : escribirá el resultado de la evaluación de la expresión en la pantalla.

WRITELN(a+b) ;

Una combinacón de los anteriores : Se escriben cada uno de los items, siguiendo el comportamiento descritos.

WRITELN(´La X =´, a) ;

- **FORMATO DE SALIDA:**

Cuando PASCAL presenta resultados en pantalla usando los procedimientos WRITE o WRITELN sigue las siguientes reglas por omisión para escribir en la salida cada tipo de dato.

INTEGER: Un número entero se manda a la salida sin espacios en blancos anteriores o posteriores.

REAL: El número real se manda a la salida en un campo de **18 caracteres de ancho**, con un formato de punto flotante (**exponencial**)

CHAR: Un carácter se manda a salida sin espacios anteriores o posteriores. Los caracteres de literales se escriben sin comillas sencillas o apóstrofes.

BOOLENA: Los valores TRUE o FALSE se escribe sin espacios en blanco anteriores o posteriores.

STRING: Una cadena de caracteres se escribe sin agregar espacios anteriores o posteriores.

EJEMPLO: Supongamos que en un programa de PASCAL se tiene:

```
PROGRAM Formatos;

VAR

a :INTEGER;

b :REAL;

c :CHAR;

d :BOOLEAN;

e :STRING[15];

BEGIN

a:= 45;

b:= 1.33;

c:='X';

d:=false;

e:='Turbo';

WRITELN(a,b,c,d,e);

END.
```

Escribe en la pantalla los resultados así:

45 1.3300000000E+00XFALSETurbo

Es claro que la salida anterior es desagradable y muy poco fácil de leer.

Normalmente en los programas se desea hacer una presentación más legible, para lo cual se debe especificar en el WRITELN los formatos de salida.

Para cambiar el formato de salida por omisión para los datos a escribir con un WRITE / WRITELN, especifique mediante un número entero el ancho para el campo de salida.

Para hacer eso escriba dos puntos **(:)** y un entero (**para el ancho del campo**) a cada uno de los datos a escribir. Para todos los tipos de datos, salvo los reales, el ancho del campo que se especifique debe ser mayor al ancho por omisión para que se vea algún efecto.

Para el caso de los números de reales se debe especificar cuanto espacio del ancho del campo se quiere utilizar para presentar cifras decimales como se muestra a continuación.

EJEMPLO: Modifique el programa del ejemplo anterior en el WRITELN para que quede así:

WRITELN(a**:4**,b**:7:2**,c**:2**,d**:7**,e**:7**);

y analice lo que escribe el computador en la pantalla:

lo que usted verá es así:

En la línea anterior se muestra que el computador está escribiendo cada un de los datos en el WRITELN, en un espacio igual al especificado en el formato y lo escribe ajustado a la derecha.

- **OPERADORES ARITMETICOS :**

Los operadores aritméticos son:

+ SUMA.

− RESTA.

***** MULTIPLICACION.

/ DIVISION DE REALES.

MOD OBTIENE EL RESIDUO DE UNA DIVISION.

DIV DIVISION ENTERA.

POTENCIACION NO EXITE Y HAY QUE IMPLEMENTARLA

- ***OPERADOR MOD :***

Dicho operador permite conocer el residuo de una división entre dos números enteros y lo puede guardar en una variable en memoria tipo entera.

EJEMPLO:

Escriba un programa que lea dos números enteros a, b y muestre el residuo de dividir a por b.

PROGRAM Ejemplo;

VAR

x,w,

y **:INTEGER;**

BEGIN

WRITE('Entre dos numero'**)**

READLN(x**,**y**);**

w **:=** x **Mod** y;

WRITELN('El residuo de Dividir'**,** x :10**,** 'entre' **,** y:10 **,** 'es =' **,**w:10**);**

END.

- ***OPERADOR DIV:***

Dicho operador nos permite conocer el cociente entero de dividir dos números enteros.

PROGRAM Ejemplo;

VAR

x,w, y :INTEGER;

BEGIN

WRITE('Entre dos numero'**)**

READLN(x**,**y**);**

W **:=** X **DIV** Y;

WRITELN(' El cociente de Dividir ', x:10 **, 'entre' ,**y:10 **,'** es = ' **,**W:10**);**

END.

Si en la variable **X** se almacena un 5 y en la variable **Y** un 2 en la variable **W** se almacena 2 que es el cociente entero de dividir 5 entre 2. **NUNCA !** use el operador **DIV y MOD** con variables (x,y) de tipo **REAL**. **La variable donde va almacenar el resultado puede ser REAL.**

- ***JERARQUIA DE LOS OPERADORES ARITMETICOS :***

Cuando el computador evalúa una expresión aritmética sigue los siguientes criterios para obtener el resultado.

1. En una operación aritmética que incluya varios operadores aritméticos, los operadores ***, /, DIV, MOD** son los operadores aritméticos que tienen mayor prioridad, esto significa que primero se realizan las operaciones que estén asociadas con estos operadores aritméticos.

2. En una expresión aritmética compuesta, las operaciones que están asociadas con los operadores **+, −** se ejecutan después de haberse ejecutado todos los operadores aritméticos enunciados en la primera regla.

3. Si en una expresión existen varios operadores aritméticos que tengan la misma prioridad, estos se resuelven de izquierda a derecha.

EJEMPLOS: Cuando vamos a escribir expresiones algebraicas con PASCAL debemos tener cuidado al escribirlas, teniendo en cuenta las prioridades de los operadores.

Suponga que tiene la expresión algebraica, y la escribimos en un programa en PASCAL así:

Z **:=** Z + B + C/D;

La expresión anterior el computador la evalúa así:

Primero calcula C/D luego Z + B y por último suma los resultados anteriores.

Supongamos ahora que algebraicamente se tiene :