5.1. LAZOS

Hasta el momento los contenidos desarrollados en los capítulos anteriores nos han permitido conocer dos de los tres tipos de estructuras fundamentales en cualquier tipo de lenguaje de programación que son:

ESTRUCTURA 1 (SECUENCIA): Grupo de sentencias que se van ejecutando de manera secuencial.

ESTRUCTURA 2 (SELECCION): Bloque de bifurcación que permite ejecutar diferentes grupos de sentencias.

Ahora se describirá el tercer tipo fundamental de estructura de cualquier programa, el LAZO o BUCLE.

ESTRUCTURA 3 (LAZO o BUCLE): Permite la repetición de cualquiera de las dos estructuras anteriores de manera cíclica.

Este tipo de construcciones nos permites realizar una operación o grupo de sentencias un numero de veces que se fijara a través de una variable entera. Su construcción básica se muestra a continuación:

ESTRUCTURA	SIGNIFICADO
DO i = u ₁ , u ₂ , u ₃ Grupo de sentencias ENDDO	 DO: Sentencia FORTRAN de comienzo de lazo. i : Indice entero del lazo DO, también se llama variable del lazo DO u₁: Numero entero o real que indica el valor inicial para el LAZO u₂: Numero entero o real que indica el valor final para el LAZO u₃: Numero entero o real que indica el paso o incremento para el LAZO, si se omite el incremento será por defecto u₃ = 1 ENDDO : Sentencia de fin de lazo.

La estructura de LAZO o BUCLE anterior se ejecuta del siguiente modo:

Cuando el programa llega a una sentencia DO comienza una ejecución cíclica en la que la variable i irá tomando valores que dependerán de las variables u_1 , u_2 , u_3 . Veamos esto para cada ciclo de ejecución:

 1^{o} Ciclo: Nada mas iniciarse este ciclo la variable i tomará el valor u_{1}

2º Ciclo: Al iniciarse el segundo ciclo la variable i tomará el valor u₁+ u₃

 3° Ciclo: Al iniciarse el tercer ciclo la variable i tomará el valor $u_1 + 2u_3$

Nº Ciclo: Este será el último ciclo, el cual se produce cuando i toma el valor u₂.

En caso de que la suma de incrementos no concuerde con el valor de \mathbf{u}_2 , el lazo finalizará con el valor de \mathbf{i} previo al que sobrepase el valor de \mathbf{u}_2 . Veamos algunos ejemplos sencillos de aplicación:

Ejemplo

LAZO	SALIDA POR PANTALLA
DO i=4,10,2 PRINT*,i ENDDO	4 6 8 10
DO i=30,15,-5 PRINT*,i ENDDO	30 25 20 15

DO i=2,9,2 PRINT*,i 4 ENDDO 6

Veamos ahora un ejemplo de un programa completo haciendo el desarrollo del mismo desde el principio:

Ejemplo:

PROBLEMA

Programa que calcule la media de un conjunto de números reales.

SOLUCION

Realicemos un planteamiento del problema antes de comenzar la codificación.

Problema: Calcular la media de un conjunto de N números reales

Descripción de Entradas y Salidas:

- El usuario deberá introducir el numero N de valores y los valores del conjunto de números
- La salida deberá presentar la media del conjunto de números.

Algoritmo:

- 1. Lectura de N y de los N números
- 2. Inicializar el valor de la suma
- 3. sumar los N valores
- 4. dividir la suma entre los N valores
- 5. Presentar la media como resultado

CODIFICACION

					Р	R	О	G	R	A	M			М	Е	D	I	A	S									
						IX	0	0	IX	71	171			141	L	Ъ		71	, S									
					I	М	P	L	I	С	I	Т			N	О	N	Е										
					R	Е	A	L			D	A	Т	О	S	,	S	U	M	Α	,	R	Е	S				
					I	N	Т	Е	G	Е	R		N															
С	L	Е	Е	M	0	S		Е	L		N	U	M	Е	R	О		D	Е		V	Α	L	О	R	Е	S	
					Р	R	I	N	Т	*		,	N	U	M	Е	R	0		D	Е		D	Α	Т	0	S	,
					R	Е	A	D	*		N																	
С	I	N	I	С	I	A	L	I	Z	O		L	Α		S	U	M	A										
					S	U	M	Α	=	0		0																
С	С	0	M	I	E	N	Z	0		Е	L		L	Α	Z	0												
					D	0		i	=	1	,	N																
С	Р	I	D	0		U	N		D	A	T	0		Y		L	0		L	Е	0							
					Р	R	I	N	Т	*		,	D	A	M	Е		U	N		D	Α	Т	О	,			
					R	Е	A	D	*		D	Α	Т	0		_		Ü	.,					Ü				
С	R	Е	A	L	I	Z	0		L	A		S	U	М	A		Α	С	U	M	U	L	Α	Т	I	V	A	
		_		_	S	U	М	Α	=	S	U	М	A	+	D	Α	Т	0	Ü		Ü	_			-			
					Е	N	D	О			Ū							Ü										

С	С	A	L	С	U	L	О		L	A		M	Е	D	I	A										
					R	Е	S		S	U	M	A	/	R	Е	A	L	(N)						
С	P	R	Е	s	Е	N	Т	О		L	О	S		R	Е	S	U	L	Т	A	D	О	s			
					P	R	I	N	Т	*	,	,	M	Е	D	I	Α	=	,	,	R	Е	S			
С	F	I	N		D	Е	L		P	R	О	G	R	A	M	A										
					Е	N	D																			

El programa anterior representa un ejemplo claro de como utilizar un lazo DO de manera eficiente, por ello se recomienda escribir el programa, compilarlo y ejecutarlo con el fin de entender como funciona. Por otro lado es indispensable entender los siguientes aspecto del código:

- 1. La parte sombreada constituye la parte del programa en la que se desarrolla el lazo, por lo tanto esa parte se repetirá N veces.
- 2. N es un numero entero puesto que indica el numero de valores que introduciremos para hallar la media.
- 3. La petición de N y su lectura se realiza <u>fuera del lazo</u> DO.
- 4. La lectura de los N valores se realiza <u>dentro del lazo</u> puesto que el lazo tiene N ciclos y permite su lectura de manera continuada tal y como se habrá podido comprobar al ejecutar el programa.
- 5. La variable i ira tomando los siguientes valores:

6. Obsérvese que cuando i=1 entonces:

SUMA=0.0 y DATO = valor del primer 1° dato, luego al final del primer ciclo SUMA=0.0+ DATO, ósea SUMA= valor del primer 1° dato.

En el segundo ciclo cuando i=2, SUMA= (valor del primer 1º dato) + (valor del primer 2º dato) y así sucesivamente hasta N ciclos. Como se observa la variable SUMA va acumulando las sumas parciales.

7. Por ultimo resaltar que a la hora de calcular la media se ha aplico la siguiente operación:

RES=SUMA/REAL(N)

Es decir se ha convertido el denominador en un numero real con el fin de que numerador y denominador sean ambos reales y evitar problemas.

EJERCICIO 5.1

• Realizar un programa que calcule la suma de todos los números enteros comprendidos entre 0 y N, siendo N un numero entero introducido por teclado.

EJERCICIO 5.2

Realizar un programa que lea un numero entero por pantalla y calcule su factorial.

EJERCICIO 5.3

Supongamos que tenemos la siguiente función:

$$F(x) = 2 x^3 + 4x^2 + 2$$
 con x 0 [-4, +4]

Y deseamos obtener una serie de puntos para realizar una gráfica para un intervalo [A,B]. Se pide realizar un programa que pregunte al usuario los valores del intervalo A y B y un incremento para que el programa pueda establecer un numero finito de valores de la variable X y así obtener los valores de F(x) correspondientes. Además el programa deberá presentar los resultados en dos columnas como se muestra a continuación:

Χ	f(x)
Valor 1	Resultado 1
Valor 2	Resultado 2
Valor 3	Resultado 3
Valor 4	Resultado 4

EJERCICIO 5.4

• Realizar un programa que presente como resultados la equivalencia entre grados y radianes de 0 a 2pi de grado en grado.

5.2.- ANIDAMIENTO DE LAZOS

Al igual que ocurría con la sentencia **IF-ELSE-ENDIF** para la cual estaba permitido ir anidando los bloques de manera continua , siempre y cuando mantuviéramos las reglas de escritura correspondientes, para el caso del bloque **DO-ENDDO** ocurre lo mismo, a continuación se muestra este tipo de estructura.

```
TIPICO ANIDAMIENTO

DO i=1,3,1
Sentencias FORTRAN 1° nivel
DO j=1,2,1
Sentencias FORTRAN 2° nivel
DO k=1,2,1
Sentencias FORTRAN 3° nivel
ENDDO
ENDDO
ENDDO
ENDDO
```

Respecto al anidamiento anterior es preciso observar los siguientes aspectos:

- Es muy conveniente realizar un sangrado regular para cada uno de los bloques **DO-ENDDO**, esto facilita la comprensión de la estructura.
- Al estar anidados tres lazos DO aparecen tres variables de lazo **i,j** y **k.** Todas ellas deberán estar declaradas como enteras en la sección de declaración de variables.
- Para entender el comportamiento de este tipo de estructuras anidadas es preciso entender que:

```
Las sentencias FORTRAN 1º nivel serán ejecutadas 3 veces
Las sentencias FORTRAN 2º nivel serán ejecutadas 3x2=6 veces
Las sentencias FORTRAN 3º nivel serán ejecutadas 3x2x2=12 veces
```

• El orden en que los ciclos se van produciendo puede entenderse observando los valores que van tomando de **i,j** y **k** a medida que la estructura se ejecuta.

i	j	k
1	1	1
1	1	2
1	2	1
1	2	2
2	1 2 2 1	1
2	1	2
2	1 2 2 1	1
2	2	2
3	1	1
1 1 1 1 2 2 2 2 2 3 3 3 3	1 2 2	1 2 1 2 1 2 1 2 1 2 1 2 1 2
3	2	1
3	2	2

Como puede observarse los lazos mas profundos dentro del anidamiento son los que mas repeticiones realizan en función de los lazos menos profundos. Es decir, para que el índice \mathbf{j} de un paso ha de completarse el numero total de ciclos establecidos para \mathbf{K} y de forma equivalente para que \mathbf{i} de un paso habrán de completarse los ciclos establecidos para \mathbf{j}

EJERCICIO 5.5

 Realizar un programa con cuatro lazos DO-ENDDO anidados y que liste por pantalla cada una de las variables de lazo establecidas, los resultados deberán ser mostrados de forma equivalente a los a los que muestra la tabla anterior.

EJERCICIO 5.6

 Calcular el valor de la serie de Taylor para e^x y para un numero de términos N introducido por teclado.

$$e^{x} = 1 + x + (x^{2}/2!) + (x^{3}/3!) + \dots$$

EJERCICIO 5.7 (opcional)

 Calcular el valor de la serie de Taylor para sen x y para un numero de términos N introducido por teclado.

sen x = x -
$$(x^3/5!)$$
 + $(x^5/5!)$ - $(x^7/7!)$ +

5.3.- SENTENCIA DO WHILE

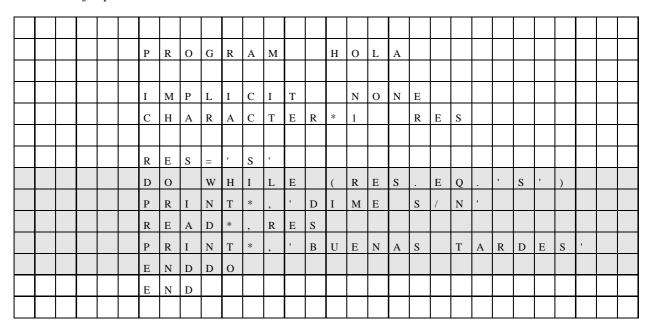
Un caso especial de utilización de la sentencia DO es el bloque constituido por las sentencia **DO WHILE-ENDDO**. Este tipo de sentencia tiene por objeto establecer un lazo de ejecución que una vez iniciado solamente finalizara en caso de cambiar de valor la expresión lógica que lo controla. La estructura básica de un lazo DO WHILE-ENDDO se muestra a continuación:

DO WHILE (expresión lógica)

Grupo sentencias ejecutadas de manera cíclica hasta que la expresión lógica tome el valor .false.

ENDDO

Veamos un ejemplo:



Del ejemplo anterior es preciso resaltar los siguientes aspectos:

- Obsérvese que se ha inicializado **RES** = 'S', con el fin de que la expresión lógica (**RES.EQ.'S'**) nada mas comenzar la ejecucion, de como resultado el valor .true. De este modo se garantiza el comienzo del lazo DO WHILE.
- La región sombreada del programa será ejecutada de manera cíclica hasta que la expresión lógica (RES.EQ.'S') sea .false.
- Dentro del lazo DO WHILE se lee a través de la sentencia **READ*, RES** la variable RES.
- Si el usuario introduce 'S' la expresión lógica mantendrá su valor .true. y por lo tanto el programa escribirá en la pantalla HOLA y nuevamente esperara un valor para RES.
- Cuando el usuario introduzca valores para RES diferentes de 'S' el programa finalizara.

EJERCICIO 5.8

• En el ejercicio 4.4 se calcularon las raíces de la ecuación de 2º grado. El programa permitía resolver la ecuación una vez por ejecución, ahora con la sentencia DO WHILE podemos establecer una estructura que permita al usuario volver a introducir los coeficientes de la ecuación A,B y C de una nueva ecuación y calcular las nuevas raíces. Este proceso podrá continuar hasta que el usuario decida lo contrario. Realizar las modificaciones adecuadas en el programa del ejercicio 4.4 para que lograr lo anteriormente expuesto.

EJERCICIO 5.9

 Realizar un programa que pida un numero real por pantalla y lo divida por dos de manera consecutiva hasta que su valor sea inferior a 0.01. El programa deberá presentar como resultado el valor final de las sucesivas divisiones y el numero de divisiones realizadas. Emplear la sentencia DO WHILE.

EJERCICIO 5.10

 Realizar el ejercicio anterior empleando una combinación de las sentencias DO-ENDDO e IF-ELSE-ENDIF.