

PROGRAMACIÓN I - UNIDAD IV: ARREGLOS

Control de versiones:

Versión	Descripción cambios	Fecha
1.0	Inicial	13/05/2021
1.1	Correcciones ejemplos	20/05/2021

Contenido

Arreglos.....3

Arreglos Unidimensionales.....3

 Ejemplo 15

Arreglos Bidimensionales6

 Ejemplo 27

Arreglos Multidimensionales.....8

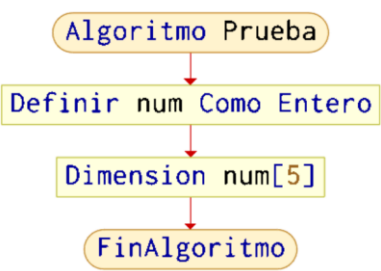
Arreglos

En el trabajo de programación es muy probable que se requiera tratar con conjuntos de datos de manera unitaria, donde el tamaño del conjunto resulta muy importante. Un arreglo es un conjunto de variables en donde cada una de ellas puede ser referenciada utilizando su posición relativa, es decir, su ubicación en relación con el primer elemento de dicho conjunto. Por ejemplo, podemos pensar en la cola en un banco, donde cada persona se ubica en una posición definida (primer lugar, segundo lugar, etc.)

Arreglos Unidimensionales

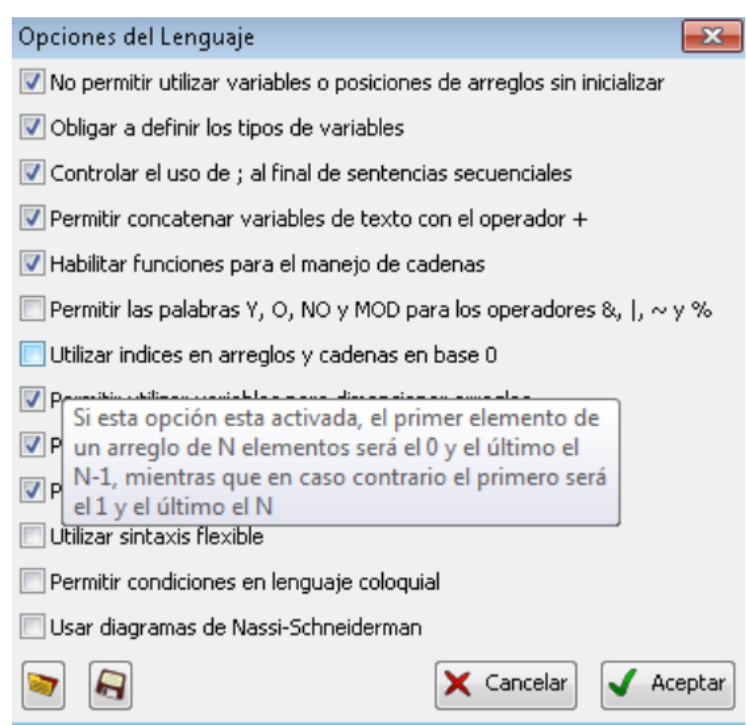
Un arreglo unidimensional posee un **nombre** (un identificador), un **tipo** y una **dimensión**. El nombre de un arreglo es un identificador del programa, el cual debe ser el único al interior del bloque de definición. El tipo, por su parte, es un tipo escalar estándar, un tipo apuntador o cualquier otro tipo definido por el programador; en este caso, se excluye el tipo *void*. La dimensión es un número entero positivo que indica el tamaño del arreglo.

Los arreglos o arrays en PseInt son estructuras que almacenan valores del mismo tipo como números o cadenas. Tiene una longitud determinada. Lo primero es declarar su dimensión, indicando el nombre del arreglo o array y su longitud dentro de [].

Pseudocódigo	Diagrama de flujo
<pre>Algoritmo Prueba Definir num como entero; Dimension num[5]; FinAlgoritmo</pre>	

Debemos definir num con el tipo de dato que queramos que sea, en este caso numérico, ya que si no al recorrer nos saltará un error.

En los arrays o arreglos empiezan desde 1 y acaban en la longitud que hayamos definido. Si has probado otros lenguajes de programación, sabrás que los arrays empiezan en 0 y acaban en la longitud -1 que hayamos definido. Si es tu caso lo podemos cambiar en las opciones de PseInt, Configurar -> Opciones del Lenguaje -> Personalizar, nos aparecerá lo siguiente:



Elegimos la opción utilizar índices en arreglos y cadenas en base 0.

Nosotros lo dejaremos por defecto, es decir, que empiecen en 1.

Veamos cómo se asignan valores al arreglo:

Pseudocódigo	Diagrama de flujo
<p>Proceso Prueba</p> <p>Definir num como entero;</p> <p>Dimension num[5];</p> <p>num[1]=5;</p> <p>num[2]=10;</p> <p>num[3]=15;</p> <p>num[4]=20;</p> <p>num[5]=25;</p> <p>FinProceso</p>	<pre>graph TD; A([Algoritmo Prueba]) --> B[Definir num Como Entero]; B --> C[Dimension num[5]]; C --> D[num[1] ← 5]; D --> E[num[2] ← 10]; E --> F[num[3] ← 15]; F --> G[num[4] ← 20]; G --> H[num[5] ← 25]; H --> I([FinAlgoritmo]);</pre>

Como vemos, simplemente debemos indicar el nombre del array, el índice y su valor. Deben estar dentro de la longitud creada, es decir, que en este caso no podríamos asignar un valor al índice 6.

Para recorrer un array, deberemos usar una estructura repetitiva, lo más recomendable es un Para o Desde donde la variable que declaremos se insertara en el índice haciendo que en cada repetición varié. Veamos un ejemplo:

Pseudocódigo	Diagrama de flujo
<p>Algoritmo Prueba</p> <p>Definir num, i como entero;</p> <p>Dimension num[5];</p> <p>num[1]=5;</p> <p>num[2]=10;</p> <p>num[3]=15;</p> <p>num[4]=20;</p> <p>num[5]=25;</p> <p>Para i<-1 Hasta 5 Con Paso 1 Hacer</p> <p> escribir num[i];</p> <p>Fin Para</p> <p>FinAlgoritmo</p>	<pre>graph TD; A([Algoritmo Prueba]) --> B[Definir num Como Entero]; B --> C[Dimension num[5]]; C --> D[num[1] ← 5]; D --> E[num[2] ← 10]; E --> F[num[3] ← 15]; F --> G[num[4] ← 20]; G --> H[num[5] ← 25]; H --> I((i 1 1 5)); I --> J[/num[i]/]; J --> I; I --> K([FinAlgoritmo]);</pre>

Además de mostrar los valores, también podemos introducirlos por teclado con leer nombre_array[indice] si queremos introducir valores en todos los índices, para ello usaremos un bucle. Veamos un ejemplo:

Pseudocódigo	Diagrama de flujo
<p>Algoritmo Prueba</p> <p>Definir num, i como entero;</p> <p>Dimension num[5];</p> <p>Para i<-1 Hasta 5 Con Paso 1 Hacer</p> <p> leer num[i];</p> <p>Fin Para</p> <p>Para i<-1 Hasta 5 Con Paso 1 Hacer</p> <p> escribir num[i];</p> <p>Fin Para</p> <p>FinAlgoritmo</p>	<pre>graph TD Start([Algoritmo Prueba]) --> Def[Definir num, i Como Entero] Def --> Dim[Dimension num[5]] Dim --> Loop1(()) Loop1 --> Read[/leer num[i]/] Read --> Counter1((i: 1 1 5)) Counter1 --> Loop1 Loop1 --> Loop2(()) Loop2 --> Write[/escribir num[i]/] Write --> Counter2((i: 1 1 5)) Counter2 --> Loop2 Loop2 --> End([FinAlgoritmo])</pre>

Ejemplo 1

Desarrollar un algoritmo que permita ingresar 5 datos numéricos en un arreglo y que luego los sume a todos y muestre el resultado en pantalla.

Ejemplo en pseudocódigo

```
Algoritmo suma_arreglo

Definir arreglo, i como Entero;
Definir acum Como Real

Dimension arreglo[5];
acum = 0;

Para i<-1 Hasta 5 Con Paso 1 Hacer
    Escribir "Digite el número"
    Leer arreglo[i];
    acum = acum + arreglo[i];
Fin Para

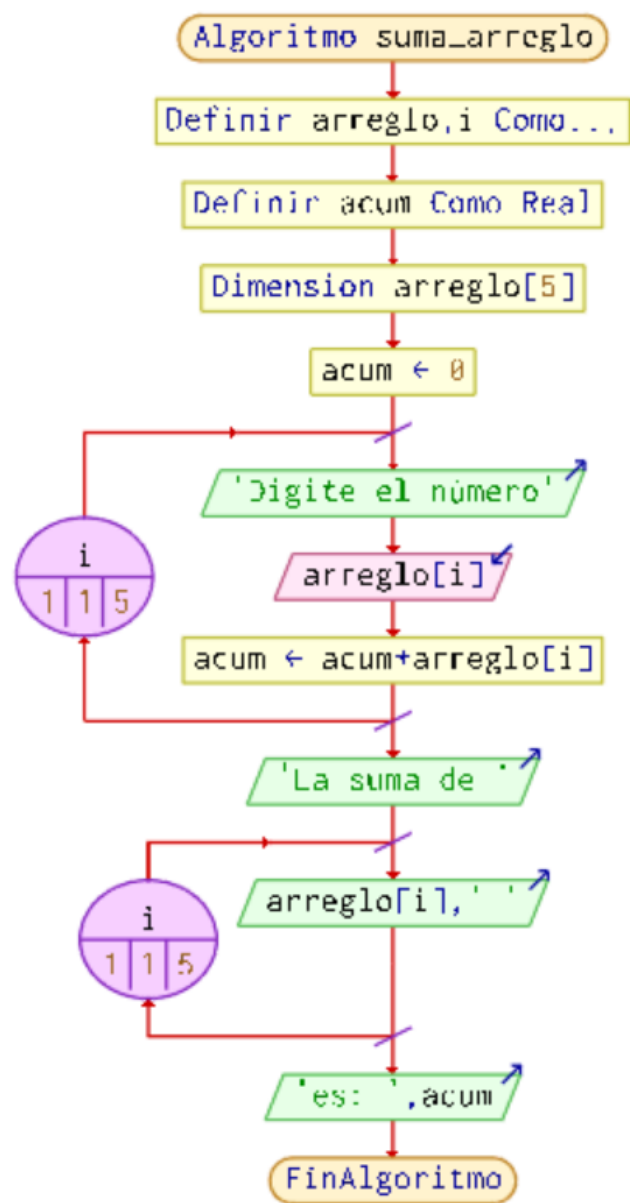
Escribir Sin Saltar "La suma de ";

Para i<-1 Hasta 5 Con Paso 1 Hacer
    Escribir Sin Saltar arreglo[i], " ";
Fin Para

Escribir "es: ", acum;

FinAlgoritmo
```

Ejemplo en diagrama de flujo



Arreglos Bidimensionales

Los arreglos bidimensionales o matrices son estructuras de datos que organizan su información en forma de tablas, es decir, los elementos que la conforman están dispuestos bajo dos conceptos de clasificación (fila y columna). Para poder indicar el lugar donde se encuentra un determinado elemento, es necesario utilizar dos índices: uno para indicar el renglón o fila y otro para indicar la columna. Por ejemplo, si tenemos un arreglo llamado matriz de dos dimensiones como se muestra a continuación:

Dimension matriz[3, 6];

Gráficamente podemos representar al arreglo bidimensional como se muestra a continuación:

	0	1	2	3	4	5
0						
1						
2						

Un arreglo bidimensional N * M tiene N filas y M columnas; por lo tanto, tiene N * M elementos dispuestos interiormente en memoria en forma sucesiva.

Por ejemplo, podemos pensar en las ventas de las cajas de dos locales. Si cada local tiene rendidos 4 caja, podríamos pensar el monto de las ventas en el siguiente arreglo $\{\{900,700,683,104\}, \{673,745,687,924\}\}$:

Pseudocódigo	Diagrama de flujo
<p>Algoritmo Prueba</p> <p>Definir venta como real;</p> <p>Definir i, j como entero;</p> <p>Dimension venta[2,4];</p> <p>venta[1,1]=900;</p> <p>venta[1,2]=700;</p> <p>venta[1,3]=683;</p> <p>venta[1,4]=104;</p> <p>venta[2,1]=673;</p> <p>venta[2,2]=745;</p> <p>venta[2,3]=687;</p> <p>venta[2,4]=924;</p> <p>Escribir "Ventas"</p> <p>Para i<-1 Hasta 2 Con Paso 1 Hacer</p> <p> Para j<-1 Hasta 4 Con Paso 1 Hacer</p> <p> Escribir "Sucursal " i " Caja " j ": " venta[i,j];</p> <p> Fin Para</p> <p>Fin Para</p> <p>FinAlgoritmo</p>	

Ejemplo 2

Realizar un programa que permita almacenar números reales en una matriz de orden 4x3, 4 filas y tres columnas, y que al final se imprima la suma de cada uno de los números ingresados en la misma.

Ejemplo en pseudocódigo

```
Algoritmo Prueba

Definir i,j como Entero
Definir M,n,suma Como Real

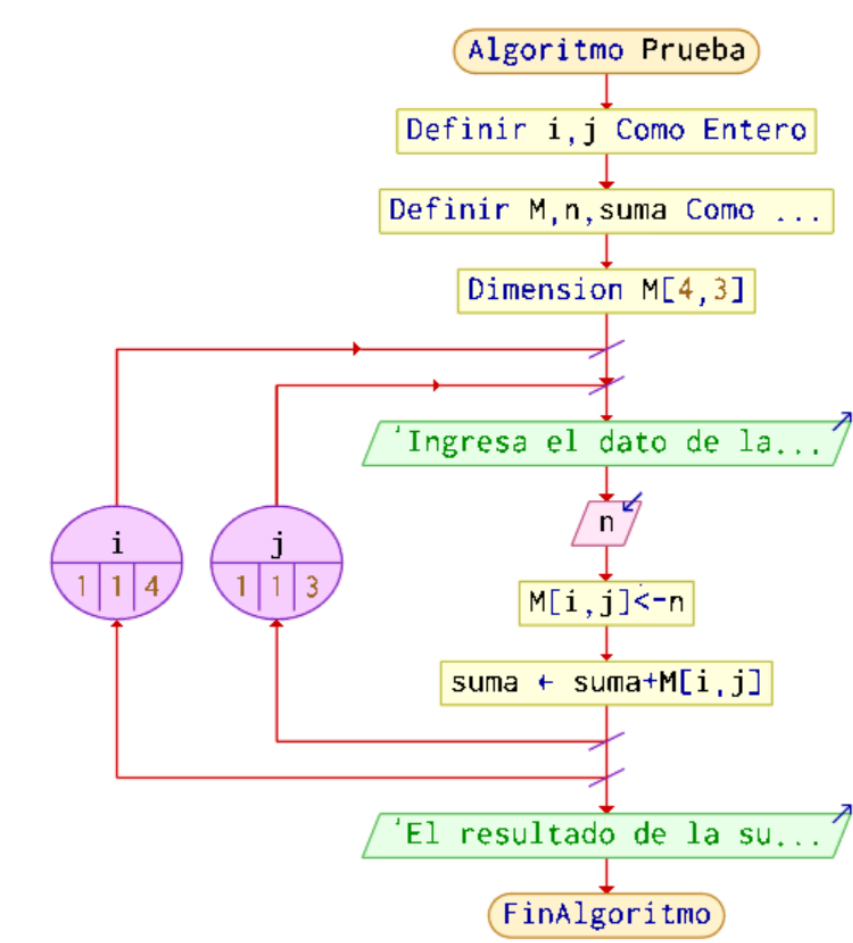
Dimension M[4,3];

Para i<-1 Hasta 4 Con Paso 1 Hacer
    Para j<-1 Hasta 3 Con Paso 1 Hacer
        Escribir Sin Saltar "Ingresa el dato de la posición [" , i, ", ", j, "];
        Leer n;
        M[i,j]<-n;
        suma<-suma+M[i,j];
    Fin Para
Fin Para

Escribir "El resultado de la suma de todos los elementos de la matriz es :",suma;

FinAlgoritmo
```


Ejemplo en diagrama de flujo



Arreglos Multidimensionales

Al declarar un arreglo de más de dos dimensiones o multidimensional es todo aquel que posee más de dos dimensiones. Por ejemplo, podemos declarar un arreglo de tres dimensiones llamado matriz_multi de la siguiente forma:

```
Dimension matriz_multi[2][4][3];
```

A continuación, se presenta un ejemplo gráfico de un arreglo multidimensional de NxMx3.

