Arreglos

Sitio: Agencia de Aprendizaje a lo largo de la Vida
Curso: Tecnicas de Programación 1° F
Libro: Arreglos

Imprimido por: MARIO DAVID GONZALEZ BENITEZ
Día: lunes, 7 de octubre de 2024, 23:02

Tabla de contenidos

1. Introducción

2. Arreglos

2.1. Usando un arreglo para optimizar la solución de un problema

3. Repasando lo visto y formalizando

1. Introducción

En esta semana conoceremos la forma en que podemos almacenar datos para su procesamiento. Hasta el momento hemos usado variables de tipo simple (entero, carácter, etc.). Una variable de tipo simple consiste de una sola “caja” de memoria y sólo puede contener un valor, es decir, existe una relación de uno a uno entre la variable y el número de elementos (valores) que es capaz de almacenar, por ejemplo, un contador.

Comenzaremos a formalizar el concepto de estructura de datos e incorporaremos el tipo estructurado arreglo que permite almacenar una colección de elementos del mismo tipo bajo un nombre único y acceder fácilmente a cada elemento individual.

2. Arreglos

Daremos aquí una definición formal del tipo de dato arreglo y durante el desarrollo del módulo iremos trabajando y comprendiendo más cabalmente los términos de la misma.

Como mencionamos, un arreglo es un conjunto de variables del mismo tipo que pueden ser referenciadas a través de un mismo nombre. Estas variables se almacenan en posiciones contiguas de memoria. La forma de identificar a un elemento determinado es a través de un índice en dónde la dirección más baja corresponde al primer elemento y la más alta al último, es decir que el índice especifica la posición relativa de la celda dentro del arreglo.

Un arreglo es una colección finita, homogénea y ordenada de elementos.

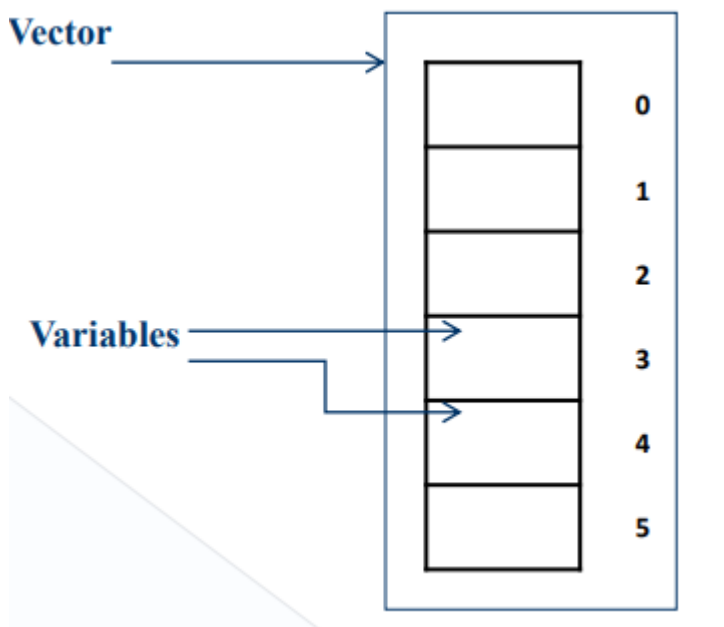
Finita: todo arreglo tiene un límite; es decir, debe determinarse cuál será el número máximo de elementos que podrán formar parte del arreglo.

Homogénea: todos los elementos del arreglo deben ser del mismo tipo.

Ordenada: existe una relación de orden; se puede determinar cuál es el primer elemento, el segundo, el tercero y el n-ésimo elemento.

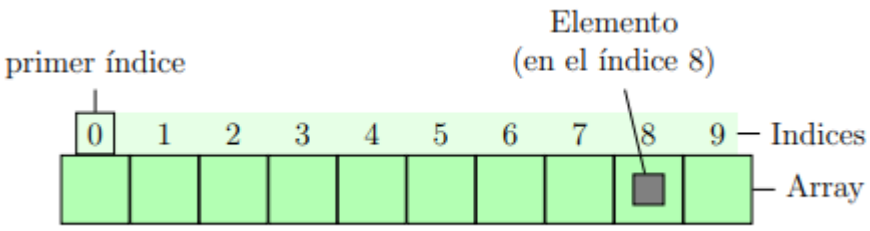
Un arreglo puede tener una o varias dimensiones. Los arreglos unidimensionales también se denominan vectores y los bidimensionales, matrices.

Según la definición, el vector es un conjunto de “contenedores” o variables llamadas celdas. Toda esta estructura tendrá un nombre que la identifique. Esto significa que cada una de las variables se llama igual. Su nombre es el mismo que el del vector, pero se diferencian mediante una segunda identificación, en este caso numérica, llamada índice.



Un array es una colección de datos que cumple con las siguientes características: todos los elementos del array son del mismo tipo; tienen una estructura secuencial, cada elemento se ubica en una posición a la que se accede por un índice; el índice de la primera posición es siempre; el índice se incrementa de uno en uno; su tamaño es estático, es decir, no puede variar la cantidad de celdas una vez definido su tamaño, no se puede agrandar ni reducir.

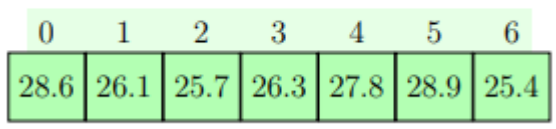
```
>
```



Por ejemplo, imaginemos que queremos guardar las temperaturas máximas de los 7 días de la semana. Podríamos crear 7 variables (una para cada día) y guardar la temperatura máxima de cada día en una de ellas:

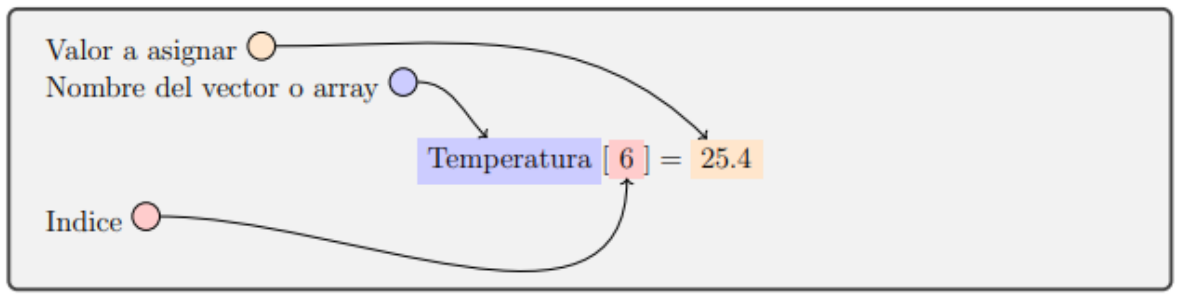
- tempLun = 28.6
- tempMar = 26.1
- tempMie = 25.7
- tempJue = 26.3
- tempVie = 27.8
- tempSab = 28.9
- tempDom = 25.4

Pero si luego queremos ampliar el rango de días a todo el mes, esto no parece muy eficiente ya que estaríamos obligados a crear muchísimas variables y se volvería muy engorroso. Una mejor alternativa es utilizar un array de 7 posiciones, al que llamaremos:



Para acceder cada una de estas temperaturas tenemos que acceder al array con el índice de cada posición:

- temperaturas[0] = 28.6
- temperaturas[1] = 26.1
- temperaturas[2] = 25.7
- temperaturas[3] = 26.3
- temperaturas[4] = 27.8
- temperaturas[5] = 28.9
- temperaturas[6] = 25.4


Valor a asignar
Nombre del vector o array
Temperatura [6] = 25.4
Indice

2.1. Usando un arreglo para optimizar la solución de un problema

Supongamos que se nos pide calcular el promedio total de las cantidades vendidas de cada uno de los 200 artículos que se comercializan en un negocio de venta de electrodomésticos y determinar la cantidad de artículos cuyas ventas superan el promedio

Con lo que conocemos hasta ahora deberíamos definir y trabajar con 200 variables….

```
Algoritmo introArray
    // Declaramos 200 variables enteras, una para cada artículo
    Definir articulo1,articulo2,articulo3,articulo4,articulo5,articulo6 Como Entero
    Definir articulo7,articulo8,articulo9,articulo10,articulo11,articulo12 como Entero
    Definir articulo13,articulo14,articulo15,articulo16,articulo17,articulo18 como Entero
    Definir articulo19,articulo20,articulo21,articulo22,articulo23.....articulo200 como Entero
    Definir promedio Como Real
    Definir cantidad Como Entero

    // Ahora debemos ingresar la venta de cada artículo. Para este ejemplo,
    // asumimos que los datos que se ingresan son válidos
    Escribir "Ingresa la cantidad del articulo 1 "
    Leer articulo1
    Escribir "Ingresa la cantidad del articulo 2 "
    Leer articulo2
    Escribir "Ingresa la cantidad del articulo 3 "
    Leer articulo3
    Escribir "Ingresa la cantidad del articulo 4 "
    Leer articulo4
    .......
    Escribir "Ingresa la cantidad del articulo 1 "
    Leer articulo200

    //Calculamos el promedio
    promedio = (articulo1 + articulo2 + articulo3 + .... articulo200) /200
    Escribir "El promedio de articulos es de: ", promedio

    // Ahora contamos cuántos artículos tienen ventas superiores al promedio
    cantidad = 0
    Si articulo1 > promedio Entonces
        cantidad = cantidad + 1
    FinSi
    Si articulo2 > promedio Entonces
        cantidad = cantidad + 1
    FinSi
    Si articulo3 > promedio Entonces
        cantidad = cantidad + 1
    FinSi
    ......
    Si articulo200 > promedio Entonces
        cantidad = cantidad + 1
    FinSi
    Escribir "La cantiadd superior al promedio es: ", cantidad
FinAlgoritmo
```

Aclaración: los puntos suspensivos los hemos puesto para no escribir todo y que no ocupe tanto espacio, ¡no se pueden usar en un programa!

Observemos que estamos procesando tan sólo 200 artículos, imaginemos la cantidad de variables a declarar y la extensión del programa para procesar todos los productos de una cadena de supermercados.

Vayamos por partes….

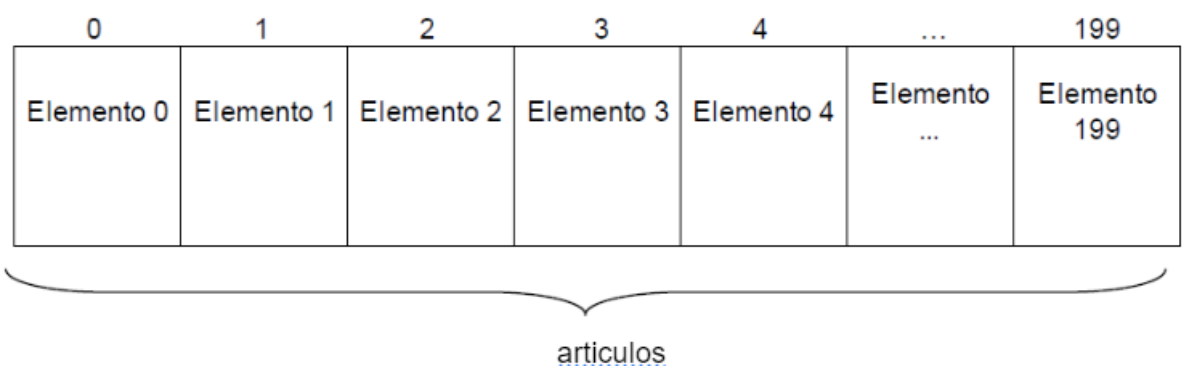
Podríamos utilizar un arreglo unidimensional para almacenar las ventas de cada uno de los 200 artículos. Como toda variable, en primer lugar debemos definirla.

Definir articulos como Entero

Y luego, le indicamos que de esa variable reserve 200 lugares:

Dimension articulos[200]

La declaración anterior instruye al compilador a que reserve una cantidad de memoria suficiente para almacenar 200 variables enteras en posiciones consecutivas y referirse a esa estructura a través del identificador artículos.



Como hemos comentado, cuando declaramos una variable, lo que estamos haciendo es reservar una zona de la memoria para ella. Cuando declaramos un arreglo, lo que hacemos (en este ejemplo) es reservar espacio en su memoria para 200 variables del tipo Entero. Para definir el tamaño del arreglo (200) utilizamos corchetes. Esta es la parte de la definición que dice: un arreglo es un conjunto de variables del mismo tipo que tienen el mismo nombre.

La parte final de la definición dice: el índice especifica la posición relativa de cada elemento dentro del arreglo.

En el ejemplo, recorremos el vector mediante un bucle For y vamos dando valores a los distintos elementos del mismo. Para indicar a qué elemento nos referimos, usamos un número entre corchetes (en este caso la variable indiceArticulo), este número es lo que se denomina índice. El primer elemento del arreglo tiene el índice 0, el segundo tiene el 1 y así sucesivamente. De modo que si queremos asignar el valor 100 al elemento 4 (índice 3) haremos:

```
articulos[3] = 100
```

Recordar que siempre el índice de un arreglo es de tipo entero y, en los lenguajes modernos, su valor varía entre 0 ya la cantidad de elementos menos 1.

En el caso del arreglo artículos, el índice varía entre 0 y 199.

Importante: tener presente que en la declaración del array el número entre corchetes es la cantidad de elementos que tendrá el arreglo; en cambio, cuando ya usamos el arreglo, accediendo a alguno de sus elementos, el número entre corchetes indica a qué variable dentro del arreglo nos referimos (el índice).

Presentamos la solución del problema usando un arreglo unidimensional. Usaremos nuestros conocimientos de bucles For.

```
Algoritmo introArrayV2
    //Definimos la variable articulos para indicarle el tipo de dato
    Definir articulos Como Entero
    //Indicamos que serán 200 articulos
    Dimension articulos[200]
    //Definimos una variable que nos permite ir moviendonos entre los articulos
    Definir indiceArticulo Como Entero
    //Definimos variables para realizar las operaciones
    Definir promedio Como Real
    Definir cantidad Como Entero

    // Vamos a ingresar la venta de cada artículo
    Para indiceArticulo = 0 Hasta 199 Hacer
        Escribir "Ingrese cantidad vendida del articulo ", indiceArticulo
        Leer articulos[indiceArticulo]
    FinPara

    //Calculamos el promedio de todas las ventas
    promedio = 0
    Para indiceArticulo = 0 Hasta 199 Hacer
        promedio = promedio + articulos[indiceArticulo]
    FinPara
    promedio = promedio / 200
    Escribir "El promedio es: ",promedio

    //Contamos la cantidad de artículos que superan el promedio
    cantidad = 0
    Para indiceArticulo = 0 Hasta 199 Hacer
        Si articulos[indiceArticulo] > promedio Entonces
            cantidad = cantidad + 1
        FinSi
    FinPara
    Escribir "La cantidad superior al promedio es: ",cantidad
FinAlgoritmo
```

Como se puede observar, es un programa más rápido de escribir, es menos aburrido hacerlo, más cómodo para el/la programador/a y mucho más fácil de modificar que el anterior.

¿Qué ocurre si ahora en lugar de 200 artículos debemos procesar 500? Debemos modificar nuestro diagrama en al menos 4 lugares. Analicemos el ejemplo anterior e intentemos ver qué se debería modificar.

Dado que a menudo el número máximo de elementos en un arreglo suele ser utilizado en distintos lugares de un programa, ya se verá más adelante, es buen estilo de programación el referenciarlo a través de una constante simbólica.

Nuestro algoritmo queda de la siguiente manera:

```
Algoritmo introArrayV3
    Definir CANTIDAD_DE_ARTICULOS Como Entero
    CANTIDAD_DE_ARTICULOS = 200
    //Definimos la variable articulos para indicarle el tipo de dato
    Definir articulos Como Entero
    //Indicamos que serán 200 articulos
    Dimension articulos[CANTIDAD_DE_ARTICULOS]
    //Definimos una variable que nos permite ir moviendonos entre los articulos
    Definir indiceArticulo Como Entero
    //Definimos variables para realizar las operaciones
    Definir promedio Como Real
    Definir cantidad Como Entero

    // Vamos a ingresar la venta de cada artículo
    Para indiceArticulo = 0 Hasta CANTIDAD_DE_ARTICULOS-1 Hacer
        Escribir "Ingrese cantidad vendida del articulo ", indiceArticulo
        Leer articulos[indiceArticulo]
    FinPara

    //Calculamos el promedio de todas las ventas
    promedio = 0
    Para indiceArticulo = 0 Hasta CANTIDAD_DE_ARTICULOS-1 Hacer
        promedio = promedio + articulos[indiceArticulo]
    FinPara
    promedio = promedio / CANTIDAD_DE_ARTICULOS
    Escribir "El promedio es: ",promedio

    //Contamos la cantidad de artículos que superan el promedio
    cantidad = 0
    Para indiceArticulo = 0 Hasta CANTIDAD_DE_ARTICULOS-1 Hacer
        Si articulos[indiceArticulo] > promedio Entonces
            cantidad = cantidad + 1
        FinSi
    FinPara
    Escribir "La cantidad superior al promedio es: ",cantidad
FinAlgoritmo
```

Observemos que en el diagrama anterior, hemos reemplazado la constante 200 por:

CANTIDAD_DE_ARTICULOS y 199 por CANTIDAD_DE_ARTICULOS-1.

¿Qué ocurre si ahora en lugar de 200 artículos se deben procesar 500?

Releé el ejemplo anterior para ver qué deberías modificar.

3. Repasando lo visto y formalizando

Como vimos en el ejemplo, en PSeInt la sintaxis para declarar un arreglo unidimensional o vector es:

```
Definir nombreArreglo como TipoDeDato

Dimensión nombreArreglo [CANTIDAD_DE_ELEMENTOS]
```

Repasemos viendo algunos ejemplos.

Para declarar un arreglo de enteros llamado listaNumeros con diez elementos escribimos:

```
Definir listaNumeros como Entero

Dimensión listaNumeros [10]
```

Recordar utilizar una constante como valor tope del arreglo. De tal modo, quedaría:

```
Definir TOPE como Entero //Se declara una constante para el valor del tope

TOPE = 10

Definir listaNumeros como Entero

Dimension listaNumeros [TOPE] //Se dimensiona un arreglo de 10 elementos
```

Todos los arreglos usan cero (0) como índice para el primer elemento. Por lo tanto, en el ejemplo anterior los elementos serán listaNumeros[0] hasta listaNumeros[9]; o en forma más general listaNumeros[0] hasta listaNumeros[TOPE-1].

Para acceder a un elemento de un arreglo, debemos indicar el nombre del arreglo y luego entre [ ] la ubicación del elemento a acceder. Por ejemplo:

```
// Asigna el valor 15 al 3er elemento del arreglo listaNumeros

listaNumeros[2] = 15

// Asigna el contenido del 6to elemento a la variable num

num = listaNumeros[5]

//Comprueba si el valor almacenado en el 1er elemento es mayor que el valor almacenado en el 2do.

Si listaNumeros[0] > listaNumeros[1] Entonces

.....

FinSi
```

Importante: el lenguaje NO realiza comprobación de contornos en los arreglos; es decir, no verifica que el índice con el que accedemos a los elementos de un arreglo tenga un valor comprendido entre cero (0) y la cantidad máxima de elementos del arreglo menos 1 (por ejemplo TOPE-1).

En el caso de que sobrepase el final durante una operación de asignación, entonces se asignarán valores a otra variable o a un trozo del código o lo que se encuentre en memoria en ese momento en ese lugar que no lo habíamos reservado para nuestro arreglo. Esto es, si se dimensiona un arreglo de tamaño TOPE, permite referenciar el arreglo por encima de TOPE sin emitir ningún mensaje de error en tiempo de compilación, aunque probablemente se provoque el fallo del programa en tiempo de ejecución.

El/la programador/a es el responsable de establecer el tamaño adecuado del arreglo en función de lo que se deba almacenar y de controlar siempre que el índice esté dentro del rango del arreglo.