

Estructuras de datos

Es una colección de datos que se caracteriza por su organización y las operaciones que se definen en ella. Los datos de tipo estándar pueden ser organizados en diferentes estructuras de datos: estáticas y dinámicas.

Las estructuras de datos estáticas son aquellas en las que el espacio ocupado en memoria se define en tiempo de compilación y no puede ser modificado durante la ejecución del programa; por el contrario, en las estructuras de datos dinámicas el espacio ocupado en memoria puede ser modificado en tiempo de ejecución. Estructuras de datos estáticas son arrays (arreglos) y las estructuras dinámicas son listas, árboles y grafos (estas estructuras no son soportadas en todos los lenguajes).

Arrays (arreglos)

Un array o arreglo es una colección de datos del mismo tipo, que se almacenan en posiciones consecutivas de memoria y reciben un nombre común. Para referirnos a una posición o elemento en particular del arreglo, especificamos el nombre del arreglo y el número de la posición de ese elemento en el arreglo.

En la siguiente figura se muestra un arreglo de enteros llamado c. Este arreglo contiene doce elementos. Nos podemos referir a cualquiera de estos elementos dando el nombre del arreglo seguido del número de posición del elemento específico encerrado en corchetes (paréntesis cuadrados []).

Nombre del arreglo (observe que todos los elementos de este arreglo tienen el mismo nombre, c)



c[0]	-45
c[1]	6
c[2]	0
c[3]	72
c[4]	1543
c[5]	-89
c[6]	0
c[7]	62
c[8]	-3
c[9]	1
c[10]	6453
c[11]	78



Número de la posición del elemento dentro del arreglo c

La posición del primer elemento de cualquier arreglo es la posición número cero. Así, nos referimos al primer elemento del arreglo c con c[0], al segundo elemento del arreglo c con c[1], al séptimo elemento del arreglo c con c[6] y, en general al i-ésimo elemento del arreglo c con c[i-1]. Los nombres de arreglo obedecen las mismas convenciones que cualquier otro tipo de variable.

El número de posición en corchetes recibe el nombre más formal de subíndice, Un subíndice debe ser un entero o una expresión entera. Si un programa utiliza un expresión como subíndice, la expresión se evaluara para determinar el subíndice. Por ejemplo, si suponemos que la variable a es igual a 5 y la variable b es igual a 6, entonces el enunciado:

`c[a+b]+=2;`

Suma 2 al elemento c[11] del arreglo.

Si quisiéramos calcular la suma en los valores contenidos en los tres primeros elementos del arreglo c y almacenar su suma en la variable suma, escribiríamos:

```
suma=c[0]+ c[1]+ c[2];
```

Si quisiéramos dividir el valor del séptimo elemento del arreglo c entre 2 y asignar el resultado a la variable x, escribiríamos:

```
x= c[6] / 2;
```

Arrays en PHP

En **PHP** existen dos tipos de principales de matrices según la manera de acceder a su contenido.

- **INDEXADAS:** Como su nombre lo indican, cada elemento de la matriz está referenciado por un índice cuyo valor es un número entero.
- **ASOCIATIVAS:** En ellas el elemento está determinado por una dupla clave => valor. El acceso a un elemento específico se hace a través de su valor.

Creación de matrices

Como en cualquier problema computacional, es importante antes de la creación de una matriz entender bien el problema, para definir cual es el tipo de matriz mas adecuado al mismo.

Independientemente de si la matriz es indexada o asociativa, se puede crear de dos formas diferentes: la implícita y la explícita.

Forma implícita:

Se crean simplemente asignándole valores a los diferentes elementos de la matriz.

Ejemplo para matriz indexada:

```
$marca[0] = 'CHEVROLET';  
$marca[1] = 'RENAULT';  
$marca[3] = 'MAZDA';  
$marca[] = 'TOYOTA';
```

Como se puede apreciar, no es necesario colocar el índice en el último valor, puesto que PHP asume el valor final.

Ejemplo para matriz asociativa:

```
$universidad["Nombre"] = 'UNIVERSIDAD DE CALDAS';  
$universidad["Direccion"] = 'CALLE 65 26-10';  
$universidad["Teléfono"] = '8861250';
```

Forma explícita:

En este caso se acude a funciones en lugar de asignar directamente los valores. En este aparte se presentará la función array().

Matriz indexada:

En este caso se asignan envían los valores como parámetros y la función se encarga de ubicarlos en el mismo orden dentro de la matriz. Por ejemplo:

```
$marca = array('CHEVROLET', 'RENAULT', 'MAZDA', 'TOYOTA');
```

Matriz asociativa:

Se hace el llamado a la función asignando clave => valor. Por ejemplo:

```
$universidad = array('Nombre' => 'UNIVERSIDAD DE CALDAS',  
                   'Direccion' => 'CALLE 65 26-10',  
                   'Teléfono' => '8861250');
```

Recorrido de una matriz:

Las matrices se recorren según su tipo de manera particular.

Recorrido de una matriz indexada:

PHP proporciona la información de una matriz por medio de la función count() y el recorrido se hace a través de su índice por medio de un ciclo.

Ejemplo:

```
for ($i=0; $i<Count($marca); $i++) {  
    echo $marca[$i].'<br>';  
}
```

Ejercicio:

Realizar un script en PHP que dados dos vectores de 6 elementos, realice su respectiva suma y resta.

Programación para sistemas en red II

Profesor: Julio César Meza Ramírez