



ESTRUCTURA DE DATOS I: ARREGLOS Y MATRICES

Autores:

Colectivo de Entrenadores ACM-ICPC UM

Febrero / 2019

ENTRENAMIENTO PARA CONCURSANTES
ACM-ICPC DE LA UNIVERSIDAD DE
MATANZAS

Objetivos

- ▶ Familiarizar a los estudiantes con estructuras de datos simples.

Contenidos

Objetivos

Contenidos

Bibliografía

Arreglo

Matriz

Conclusiones

Bibliografía

- ▶ *Manual de preparación para concursantes ACM-ICPC de la Universidad de Matanzas.* Publicado en el Entorno Virtual de Aprendizaje de la Universidad de Matanzas (eva.umcc.cu). Sección *Bibliografía*.

Arreglo

En programación, un arreglo (llamados en inglés array) es una zona de almacenamiento continuo, que contiene una serie de elementos del mismo tipo.

Esta estructura de dato son adecuadas para situaciones en las que el acceso a los datos se realice de forma aleatoria e impredecible.

Arreglo

Todo arreglo se compone de un determinado número de elementos. Cada elemento es referenciado por la posición que ocupa dentro del arreglo. Dichas posiciones son llamadas índice y siempre son correlativos. Existen tres formas de indexar los elementos de un arreglo:

- ▶ Indexación base-cero (0)

Arreglo

Todo arreglo se compone de un determinado número de elementos. Cada elemento es referenciado por la posición que ocupa dentro del arreglo. Dichas posiciones son llamadas índice y siempre son correlativos. Existen tres formas de indexar los elementos de un arreglo:

- ▶ Indexación base-cero (0)
- ▶ Indexación base-uno (1)

Arreglo

Todo arreglo se compone de un determinado número de elementos. Cada elemento es referenciado por la posición que ocupa dentro del arreglo. Dichas posiciones son llamadas índice y siempre son correlativos. Existen tres formas de indexar los elementos de un arreglo:

- ▶ Indexación base-cero (0)
- ▶ Indexación base-uno (1)
- ▶ Indexación base-n (n)

Arreglo

La forma de acceder a los elementos de un arreglo es directa; esto significa que el elemento deseado es obtenido a partir de su índice y no hay que ir buscándolo elemento por elemento.

Arreglo: C++

En lenguaje de programación C++ el arreglo se puede declarar según la situación. Ahora veremos cada una de ellas:

```
/* Sabemos la cantidad de elementos a priori,  
<tipo_de _dato> <nombre_arreglo> [<cantidad>];*/  
bool isPimes[100];
```

Arreglo: C++

/*La cantidad de elementos puede variar y depende del valor de una variable.Es la mas usada

```
<tipo_de _dato> * <nombre_arreglo>= new <tipo_de  
_dato> [<variable>];
```

```
<tipo_de _dato> * <nombre_arreglo>= new <tipo_de  
_dato> [<cantidad>];
```

```
*/
```

```
int cantidadMaxima=100;
```

```
int * notas;
```

```
notas=new int [cantidadMaxima];
```

```
double * promedio=new double [100];
```

Arreglo: C++

```
/*Cuando conocemos los valores que integran el  
arreglo*/
```

```
string nombres []={ "Luis", "Ernesto", "Susana" };
```

```
/*Para acceder o modificar algun elemento del  
arreglo lo hacemos de la siguiente manera*/
```

```
int primeraNota=notas[0];
```

```
notas[23]=cantidadMaxima;
```

```
notas[cantidadMaxima-1]=cantidadMaxima;
```

```
int ultimaNota=notas[cantidadMaxima-1];
```

Arreglo: Java

Los arreglos de Java se tratan como objetos de una clase predefinida. Los arrays son objetos, pero con algunas características propias. Los arreglos pueden ser asignados a objetos de la clase Object y los métodos de Object pueden ser utilizados con arreglos.

```
/* Declaracion de un arreglo. Se inicializa a null
   */
int [] vector;

/* arreglo de 10 enteros, inicializados a 0 */
vector = new int[10];
```

Arreglo: Java

```
/* Declaracion e inicializacion de un arreglo de 3
   elementos
   con los valores entre llaves */
double [] v = {1.0, 2.65, 3.1};

/* Se crea un arreglo de 5 referencias a objetos
   Las 5 referencias son inicializadas a null */
MyClass [] lista=new MyClass[5];
```

Arreglo: Java

```
// Se asigna a lista[0] el mismo valor que unaRef  
lista[0] = unaRef;  
  
/*Se asigna a lista[1] la referencia al nuevo  
objeto  
El resto (lista[2]...lista[4] siguen con valor null  
*/  
lista[1] = new MyClass();
```

Arreglo

Tanto en C++ como en Java para moverse o iterar sobre un arreglo se hace utilizando la estructura de repetición *for* bien desde el principio al final

```
for(int i=0;i<cantidadElementos;i++){  
    ...  
}
```

O desde el final al principio

```
for(int i=cantidadElementos-1;i>=0;i--){  
    ...  
}
```


Arreglo

De igual forma es aconsejable llevar por cada estructura de datos de tipo arreglo una variable entera que nos indique la cantidad real de elementos almacenados en el arreglo.

Matriz

Una matriz es un arreglo de arreglos fila, o más en concreto un arreglo de referencias a los arreglos fila. Con este esquema, cada fila podría tener un número de elementos diferente. Es una estructura con dimensiones ancho y altura o columna y filas.

Matriz:C++

Los arreglos bidimensionales o matrices en C++ se puede declarar similar a como se hace un arreglo unidimensional.

```
/*Se conoce de antemano las dimensiones esta manera  
es estatica se recomienda que sea dinamica*/  
int mat [3][4];
```

```
/*De forma dinamica*/  
int ** mat;  
mat = new int * [ncolumns];  
for(int i=0;i<ncolumns;i++)  
    mat[i]=new int [nfilas];
```

Matriz:C++

```
/*Con los valores conocidos*/  
double carrots[3][4]  
{ {2.5, 3.2, 3.7, 4.1}, // primera fila  
  {4.1, 3.9, 1.6, 3.5}, // segunda fila  
  {2.8, 2.3, 0.9, 1.1} // tercera fila  
};
```

Matriz:Java

Los arrays bidimensionales de Java se crean de un modo muy similar al de C++. En Java una matriz se puede crear directamente en la forma,

```
int [][] mat = new int[3][4];
```

Matriz:Java

O bien se puede crear de modo dinámico dando los siguientes pasos:

1. Crear la referencia indicando con un doble corchete que es una referencia a matriz,

```
int [] [] mat;
```

2. Crear el vector de referencias a las filas,

```
mat = new int[nfilas] [];
```

3. Reservar memoria para los vectores correspondientes a las filas,

```
for (int i=0; i<nfilas; i++);  
    mat[i] = new int[ncols];
```

Matriz:Java

A continuación se presentan algunos ejemplos de creación de arrays bidimensionales:

```
// crear una matriz 3x3
// se inicializan a cero
double mat[][] = new double[3][3];
int [][] b = {{1, 2, 3},{4, 5, 6}}, // esta coma es
    permitida
    };
int c = new[3][]; // se crea el array de
    referencias a arrays
c[0] = new int[5];
c[1] = new int[4];
c[2] = new int[8];
```

Matriz

Tanto en C++ como en Java para moverse o iterar sobre una matriz se hace utilizando la estructura de repetición *for* anidada con otra estructura del mismo tipo:

```
for(int i=0;i<filas;i++){  
    for(int j=0;j<columnas;j++){  
        ...  
    }  
}
```

Note que existe hasta 8 formas diferentes de moverse o iterar sobre una matriz. La utilización de cada una de ellas depende del problema o la situación descripta.

Conclusiones

- ▶ Los arreglos son muy útiles en problema donde la solución depende del trabajo que se realice sobre un grupos de datos.
- ▶ Existen un grupo problemas que los mapas o tableros de juego son representados mediante matrices.
- ▶ Ambas estructuras son utilizadas en algoritmos conocidos para la solución de problemas de diferentes áreas del conocimiento.

UNIVERSIDAD DE MATANZAS

cosechando el saber

FIN