

Arreglos en C++

Matrices

Tomás Peiretti

Arreglos multidimensionales

Los arreglos multidimensionales se pueden definir como **arreglos de arreglos**. En AEDD usaremos arreglos bidimensionales, a los que llamaremos **matrices** para resolver problemas.

```
1 int main() {  
2     char arreglo[100]; //el arreglo que conociamos hasta el momento  
3     int matriz[3][10]; //un arreglo bidimensional (matriz, arreglo de arreglos)  
4     int miArr[10][10][5]; //un arreglo tridimensional (un arreglo de arreglos  
5     //que contienen arreglos)  
6     bool arr[5][3][5][4][205]; //un arreglo de 5 dimensiones  
}
```

	0	1	2	3	4	5	6	7	8	9
0	-3	-1	33	34	35	1231	1253	1566	10001	13000
1	1321	3323	-33	-991	5572	1919	-90	6678	77777	-1
2	888	-1	7462	1337	969	4434	0	-745271	625	42

Arreglos multidimensionales

Para **acceder a un elemento** de un arreglo multidimensional, al igual que con los arreglos normales, se debe indicar el índice correspondiente. Solo que esta vez, tendremos que **indicar un índice para cada dimensión**:

Por ejemplo, si deseamos acceder al elemento de una matriz que se encuentra en la fila 1 y la columna 6 debemos hacer:

`matriz[0][6]`

	0	1	2	3	4	5	6	7	8	9
0	-3	-1	33	34	35	1231	1253	1566	10001	13000
1	1321	3323	-33	-991	5572	1919	-90	6678	77777	-1
2	888	-1	7462	1337	969	4434	0	-745271	625	42

Arreglos multidimensionales: funciones

Para pasar un arreglo multidimensional a una función, se debe indicar el tamaño de todas las dimensiones (excluyendo opcionalmente la primera). Sino, tendremos un error de compilación.

```
1 // ejemplo: miFuncion recibe una matriz como primer parametro ,
2 // y ademas un arreglo de 4 dimensiones como segundo parametro
3 void miFuncion(char matriz[][10], int arr[][5][3][5]) {
4     // ....
5 }
6
7 int main() {
8     char mat[30][10];
9     int arregloMulti[10][5][3][5];
10
11     miFuncion(mat, arregloMulti);
12     return 0;
13 }
```

- No olvidemos que los arreglos **por defecto pasan por referencia**

Arreglos multidimensionales: ejemplos

```
1 void imprimirMatriz(char mat[][20], int filas) {
2     for (int i=0; i<filas; i++) {
3         for (int j=0; j<20; j++) {
4             cout << mat[i][j] << " ";
5         }
6         cout << endl;
7     }
8 }
9
10 int sumaDiagonalPrincipal(int mat[][100], int tam) {
11     int sum = 0;
12     for (int i=0; i<tam; i++) {
13         sum += mat[i][i];
14     }
15     return sum;
16 }
17
18 bool esTriangularSuperior(int mat[][100], int filas, int columnas) {
19     bool esTriangularSup = true;
20     if (filas != columnas) {
21         esTriangularSup = false;
22     } else {
23         for (int i=1; i<filas; i++) {
24             for (int j=0; j<i; j++) {
25                 if (mat[i][j] != 0)
26                     esTriangularSup = false;
27             }
28         }
29     }
30     return esTriangularSup;
31 }
```

Ejercicios

- Beecrowd 1181
- Beecrowd 1182
- Beecrowd 1183
- Beecrowd 1184
- Beecrowd 1185
- Beecrowd 1186
- Beecrowd 1187
- Beecrowd 1188
- Beecrowd 1189
- Beecrowd 1435
- Beecrowd 1478
- Beecrowd 1534
- Beecrowd 1557
- Beecrowd 1827