

Arreglos, cadenas y Matrices en C++

Victor Armando Jaramillo Moreno

1 de Marzo 2025

1 Arreglos

Hasta ahora hemos usado variables para almacenar valores en memoria. Ahora explicaremos un medio para almacenar múltiples valores juntos en una sola unidad: el **arreglo**.

Un arreglo es una secuencia de tamaño fijo de elementos del mismo tipo almacenados en un pedazo secuencial de memoria. Por ejemplo, un arreglo de enteros almacena valores de tipo `int`, mientras que un arreglo de caracteres almacena valores de tipo `char`. Nos referimos al tamaño del arreglo como su **dimensión**.

1.1 Declaración de un Arreglo

Para declarar un arreglo en C++, escribimos lo siguiente:

```
tipo_de_dato nombreDelArreglo [dimension];
```

Ejemplo:

```
int arr[4];
```

1.2 Inicialización de Arreglos

Podemos inicializar los elementos de un arreglo al declararlo:

```
int arr[4] = {6, 0, 9, 6};
```

También podemos omitir el tamaño y dejar que el compilador lo determine:

```
int arr[] = {6, 0, 9, 6, 2, 0, 1, 1};
```

1.3 Uso de Bucles con Arreglos

Un uso común de los arreglos es recorrerlos con bucles `for`:

```
int arr[4] = {1, 2, 3, 4};
for (int i = 0; i < 4; i++) {
    cout << arr[i] << " ";
}
cout << endl;
```

2 Arreglos Multidimensionales

C++ también tiene soporte para arreglos multidimensionales. Un arreglo bidimensional se declara así:

```
tipo_de_dato nombreDeArreglo[fila][columna];
```

Ejemplo:

```
int matriz[2][3] = {{1, 2, 3}, {4, 5, 6}};
```

Para recorrer la matriz usamos bucles anidados:

```
for (int i = 0; i < 2; i++) {
    for (int j = 0; j < 3; j++) {
        cout << matriz[i][j] << " ";
    }
    cout << endl;
}
```

3 Cadenas

Una cadena es una secuencia de caracteres. En C++, podemos usar la clase `string` de la biblioteca estándar.

3.1 Declaración e Inicialización

Ejemplo de declaración e inicialización:

```
string mensaje = "Hola Mundo";
```

3.2 Lectura de Cadenas

Usamos `cin` para leer una palabra y `getline` para leer una línea completa:

```
string nombre;
getline(cin, nombre);
cout << "Tu nombre es: " << nombre << endl;
```

3.3 Recorrer una Cadena con un Bucle

Podemos recorrer una cadena usando `for`: Para poder recorrerla, necesitamos primero obtener la longitud de la cadena, que se puede obtener con el valor de `texto.size()`

```
string texto = "Programar";
for (int i = 0; i < texto.size(); i++) {
    cout << texto[i] << endl;
}
```

3.4 Concatenación de Cadenas

Podemos unir cadenas usando el operador `+`:

```
string nombre = "Juan";
string apellido = "Pérez";
string nombreCompleto = nombre + " " + apellido;
cout << "Nombre completo: " << nombreCompleto << endl;
```

4 Conclusión

Los arreglos permiten almacenar múltiples valores del mismo tipo, mientras que las cadenas manejan texto de forma eficiente. Entender cómo usarlos y combinarlos con bucles mejora la capacidad de escribir programas dinámicos y eficientes en C++.