Universidad Nacional de Ingeniería
CTIC - UNI

Programación en C++

Sesión 6: Arreglos - I

¿Qué es un arreglo?

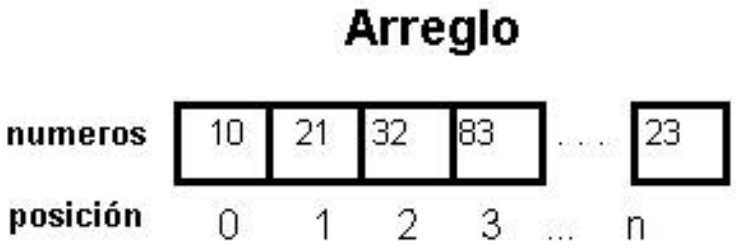
- Los arreglos son estructuras de datos complejas que agrupan datos de un mismo tipo en particular, llamado el tipo base del arreglo.

- El tipo base de un arreglo puede ser cualquiera de los tipos básicos de C++, o incluso algunos tipos complejos como las estructuras que veremos más adelante.

- Un arreglo puede ser visto como una colección ordenada de elementos de un mismo tipo.

Arreglos

- Ejemplo: un arreglo unidimensional que almacena n+1 números enteros puede ser visualizado de la siguiente manera,



- Nótese:
  - Todos los elementos son del mismo tipo
  - Los elementos tienen un ordenamiento espacial
  - Las posiciones de los n elementos van desde la 0 a la n-1

Declaración de Arreglos Unidimensionales (Vectores)

- Declaración
  - `<tipo-base> <identificador>[<NumElementos>];`
  - **Ej.:** `int vec[20];`

- Declaración con inicialización
  - `<tipo-base> <identificador>[<NumElementos>] = {valor1, valor2, ...};`
  - **Ej.:** `int edades[5] = {17,19,21,20,18};`

Declaración de Arreglos Unidimensionales (Vectores)

- Declaración con inicialización
  - `<tipo-base> <identificador>[<NumElementos>] = {valor1, valor2, ...};`
  - Ej.: `int edades[5] = {17,19,21,20,18}`
- Observaciones
  - Con los valores indicados entre llaves **`{}`** se inicializarán los elementos del arreglo.
  - Los valores deben ser del **`<tipo-base>`** del arreglo.
  - Si se incluyen menos valores que elementos en el arreglo, los últimos serán inicializados en cero.
  - Si se hace inicialización, es posible omitir el tamaño del arreglo y dejar que el compilador lo deduzca a partir de los valores asignados.

Ejemplos de uso

1. Declaración de un arreglo de 50 enteros:
```
int A[50];
```

2. Declaración de un arreglo de 100 caracteres:
```
char A[100];
```

3. Declaración e inicialización de un arreglo de 10 enteros:
```
int A[10] = { 2, 5, 8, 100, 1, 2, 100, 5, 5, 5 }
```

4. Inicialización parcial: El resto se inicializa en cero:
```
int A[100] = { 25, 5, 100, 25, 5 }
```

5. Declaración e inicialización de un arreglo de 10 caracteres:
```
char A[8] = { 'a','E', '7', '\t', '@', 'U', '*', '\n' }
```

6. Determinación en forma implícita del tamaño de un arreglo:
```
int A[] = { 5, 10, 2, 15, 20 }
```

Más ejemplos de uso

7. Asignando un valor a la sexta posición de un arreglo de enteros:

```
A[5] = 200;
```

8. Imprimiendo un arreglo de 100 enteros mediante un ciclo for:

```
int i;
for (i=0; i<100; i++)
   cout<<A[i]<<"\n";
```

9. Leyendo del usuario el contenido de un arreglo de 20 enteros, mediante un ciclo for:

```
int i;
for (i=0; i<20; i++)
   cin>>A[i];
```

10. Una función que recibe un arreglo de enteros como argumento y calcula el promedio:

```
int promedio(int A[], int num_elementos) {
   int prom = 0;
   int i;
   for (i=0; i<num_elementos; i++)
      prom = prom + A[i];
   return(prom/num_elementos);
}
```

Arreglos observaciones importantes

- Puesto que los arreglos son estructuras complejas **no es posible** asignar un arreglo a otro mediante una simple asignación (=). Para hacer esto es necesario escribir un ciclo y asignar elemento a elemento.

- También es común cometer estos errores olvidando que las posiciones de los arreglos están numeradas a partir del índice cero. Es decir, en un arreglo de tamaño N las posiciones están numeradas de 0 a N-1.

Arreglos observaciones importantes

C **no controla** la validez de los índices que se emplean para referenciar un arreglo. **Ejemplo:** la siguiente porción de código compila sin problemas pero probablemente produzca un error en tiempo de ejecución al referenciar posiciones inexistentes del arreglo.

```
/* Ejemplo de error por acceso fuera de rango a un
arreglo */

/* Posiciones con índices del 20 al 29 son
inválidas. */

      int i,  arreglo[20];

      for (i=0; i<30; i++)

        arreglo[i] = 0;
```

Y más ejemplos...

11. Llamando una función que recibe un arreglo de enteros como parámetro:

```
        int prom, A[100];
        ...
        prom = promedio(A);

       ....
int promedio(int A[]){
       ...
}
```

12. Contando el número de minúsculas de un array de caracteres

```
int numero_minusculas (char s[],int n){
      int c=0,i=0;
      while (i < n){
        if (s[i]>64 &&s[i]<97)
           c++;
        i++;
      }
      return (c);
}
```

Ordenación por Burbuja

Este método es clásico y muy sencillo, aunque por desgracia poco eficiente.

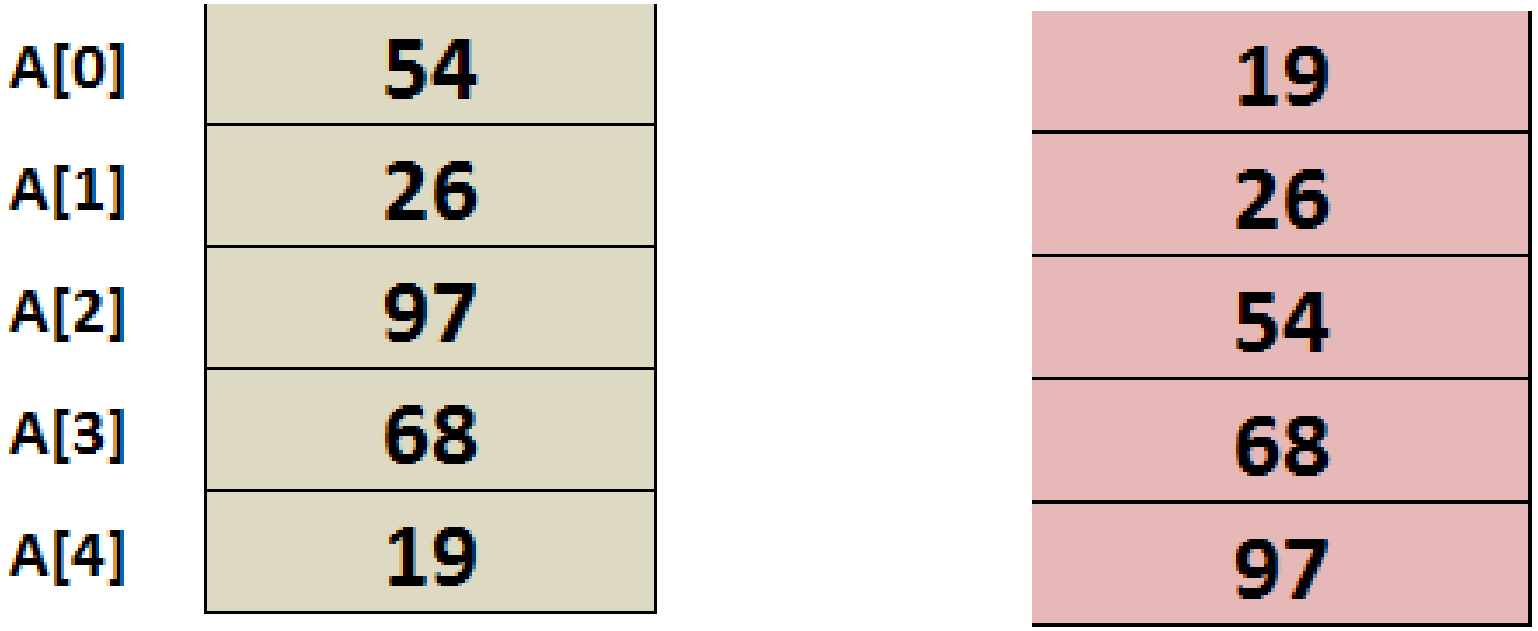
Técnica

Este método se basa en la comparación de elementos adyacentes de la lista (vector) e intercambiar sus valores si están desordenados.

De este modo, se dice que los valores más pequeños burbujean hacia la parte superior de la lista (hacia el primer elemento), mientras que los valores más grandes se hunden hacia el fondo de la lista.

Ordenación por Burbuja

Gráficamente, el proceso anterior con una lista de 5 elementos:



Lista sin ordenar

Lista Ordenada

Ordenación por Burbuja

Pasada 1: i=1

A[0]	54
A[1]	26
A[2]	97
A[3]	68
A[4]	19

Pasada 2: i=2

A[0]	26
A[1]	54
A[2]	68
A[3]	19
A[4]	97

Ordenación por Burbuja

Pasada 3: i=3

A[0]	26
A[1]	54
A[2]	19
A[3]	68
A[4]	97

Pasada 4: i=4

A[0]	26
A[1]	19
A[2]	54
A[3]	68
A[4]	97

Ejercicios:

- Hacer un programa que pregunta 5 números y luego los imprime en orden inverso.

- Hacer un programa que pregunta los coeficientes de un polinomio de grado 5
  ($a_0 + a_1x + a_2x^2 + a_3x^3 + a_4x^4 + a_5x^5$)
  y luego evalúa el polinomio desde x=-10 a x=10.