	<b>Carátula para entrega de prácticas</b>	
Facultad de Ingeniería	Laboratorio de docencia	

## Laboratorios de computación Salas A y B

---

<i>Profesor:</i>	Alejandro Pimentel
<i>Asignatura:</i>	Laboratorio de programación
<i>Grupo:</i>	135
<i>No de Práctica(s):</i>	Práctica 11
<i>Integrante(s):</i>	González Segura Areli Rodríguez Guzmán Paola Mariel Villanueva Bustamante Victoria
<i>No. de Equipo de cómputo empleado:</i>	23
<i>No. de Lista o Brigada:</i>	5319, 4926, 8043
<i>Semestre:</i>	Primer semestre
<i>Fecha de entrega:</i>	28/octubre/2019
<i>Observaciones:</i>	Su práctica esta incompleta. En la segunda actividad el objetivo era hacer una suma de matrices, no solamente recibir una matriz y mostrarla
<b>CALIFICACIÓN:</b>	_____

## Objetivo:

Reconocer la importancia y utilidad de los arreglos, en la elaboración de programas que resuelvan problemas que requieran agrupar datos del mismo tipo, así como trabajar con arreglos tanto unidimensionales como multidimensionales.

## Introducción:

**Arreglos:** Espacios ordenados Un arreglo se puede ver como un conjunto de espacios finitos donde se almacenan elementos que son todos del mismo tipo. Un arreglo también puede verse como cajas ordenadas en fila y numeradas, donde en cada caja se almacena un solo elemento u objeto.

## Arreglos unidimensionales

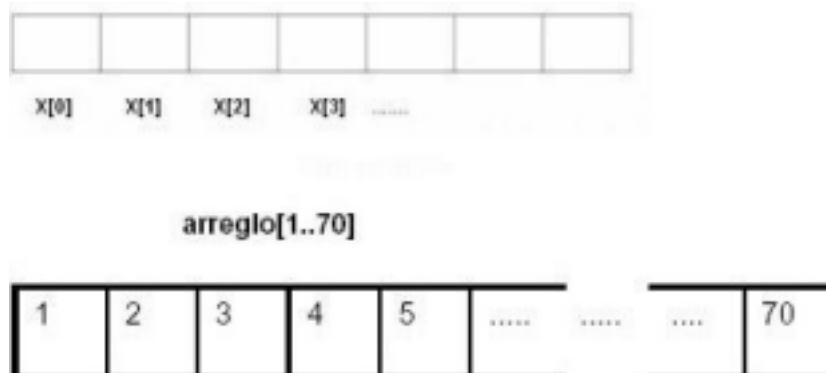
Es un tipo de datos estructurado que está formado por una colección finita y ordenada de datos del mismo tipo. Es la estructura natural para modelar listas de elementos iguales. Los datos que se guarden en los arreglos todos deben ser del mismo tipo.

El tipo de acceso a los arreglos unidimensionales es el acceso directo, es decir, podemos acceder a cualquier elemento del arreglo sin tener que consultar a elementos anteriores o posteriores, esto mediante el uso de un índice para cada elemento del arreglo que nos da su posición relativa.

Para implementar arreglos unidimensionales se debe reservar espacio en memoria.

Los arreglos nos permiten hacer un conjunto de operaciones para manipular los datos guardados en ellos, estas operaciones son: ordenar, buscar, insertar, eliminar, modificar entre otras.

**REPRESENTACION EN MEMORIA:** Los arreglos guardan en memoria la cantidad de espacios que se le indican en la declaración.

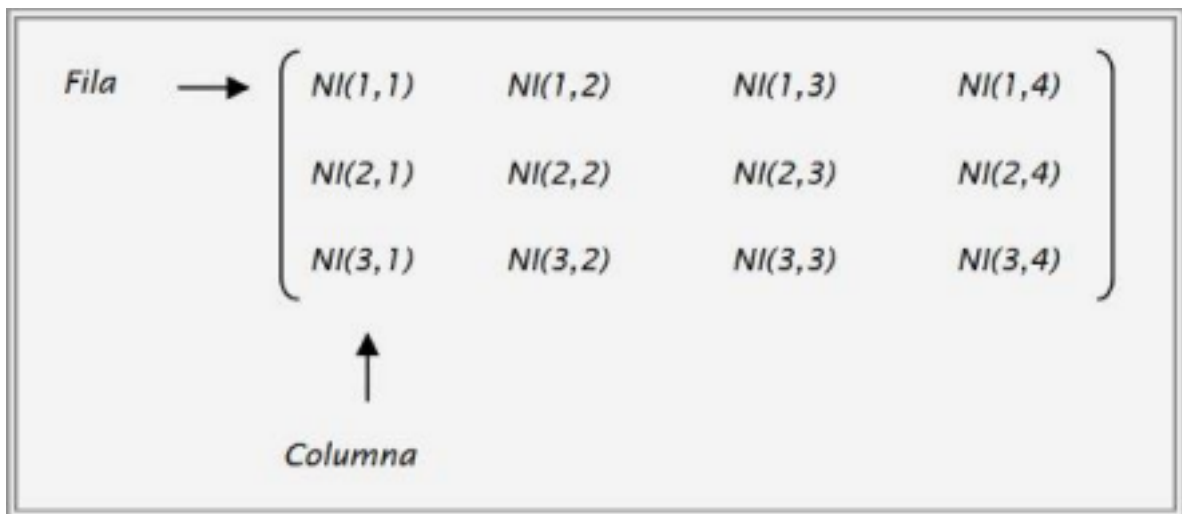


## Arreglos multidimensionales

Un arreglo multidimensional es simplemente una extensión de un arreglo unidimensional. Más que almacenar una sola lista de elementos, piense en un arreglo multidimensional como el almacenamiento de múltiples listas de elementos. Por ejemplo, un arreglo bidimensional almacena listas en un formato de tabla de dos dimensiones de filas y columnas, en donde cada fila es una lista.

Las filas proporcionan la dimensión vertical del arreglo, y las columnas dan la dimensión horizontal. Un arreglo de tres dimensiones almacena listas en un formato de tres dimensiones de filas, columnas y planos, en donde cada plano es un arreglo bidimensional.

Las filas proporcionan la dimensión vertical; las columnas, la dimensión horizontal; y los planos, la dimensión de profundidad del arreglo.



Desarrollo:

Actividad 1:

- Hacer un programa que:
- Pida al usuario un número.
- Genere un arreglo de esa longitud.
- Pida al usuario números suficientes para llenar el arreglo.
- Muestre al usuario el número menor y el mayor de dicho arreglo.

```
1 #include <stdio.h>
2 #include<stdlib.h>
3 int main(){
4     int arreglo[100],c;
5     int mayor, menor,i;
6     printf("ingrese cantidad de numeros ");
7     scanf("%i",&c);
8     for(i=0;i<c;i++)
9     {
10
11     printf("numero %d: ",i+1);
12     scanf("%d",&arreglo[i]);
13     if(mayor<arreglo[i]){
14         mayor=arreglo[i];
15     }
16     menor=mayor;
17     for(i=0;i<c;i++)
18     if(menor>arreglo[i])
19         menor=arreglo[i];
20
21     printf("el mayor es: %i\n", mayor);
22     printf("el menor es: %i\n", menor);
23
24     return 0;
25 }
26
27
28
```

Much cuidado, si usas variables sin valor, puedes tener errores muy impredecibles

Compile Log | Debug | Find Results

Sel: 0 Lines: 28 Length: 477 Insert Done parsing

Icons: Chrome, Internet Explorer, Word, Windows, Dev-C++, File Explorer

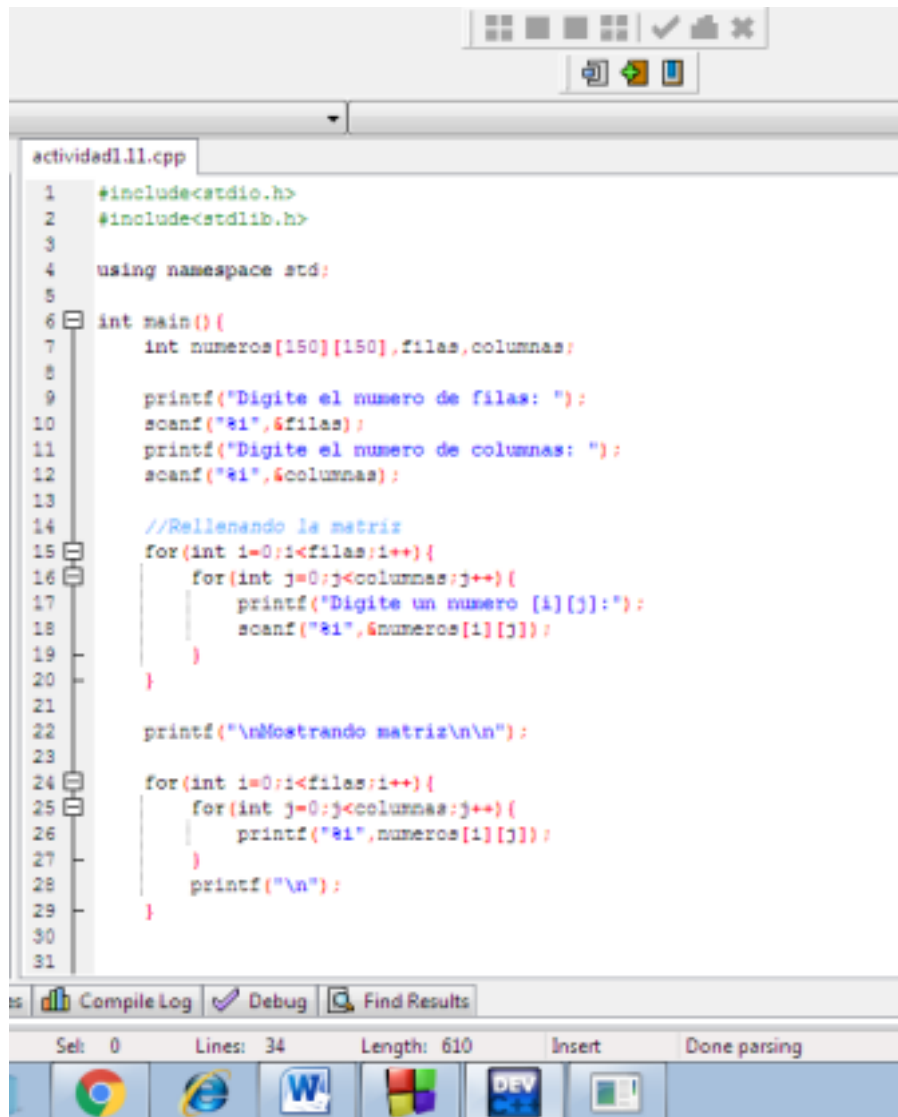
Compilado

```
C:\Program Files\Dev-Cpp\ConsolePauser.exe
ingrese cantidad de numeros
3
numero 1: 10
numero 2: 15
numero 3: 16
el mayor es: 16
el menor es: 10

Process exited with return value 0
Press any key to continue . . .
```

## Actividad 2

- Hacer un programa que:
- Pida al usuario uno dos números N y M.
- Genere dos matrices de  $N \times M$ .
- Pida al usuario números suficientes para llenar ambas matrices.
- Muestre al usuario la matriz resultado de sumar las dos de entrada.

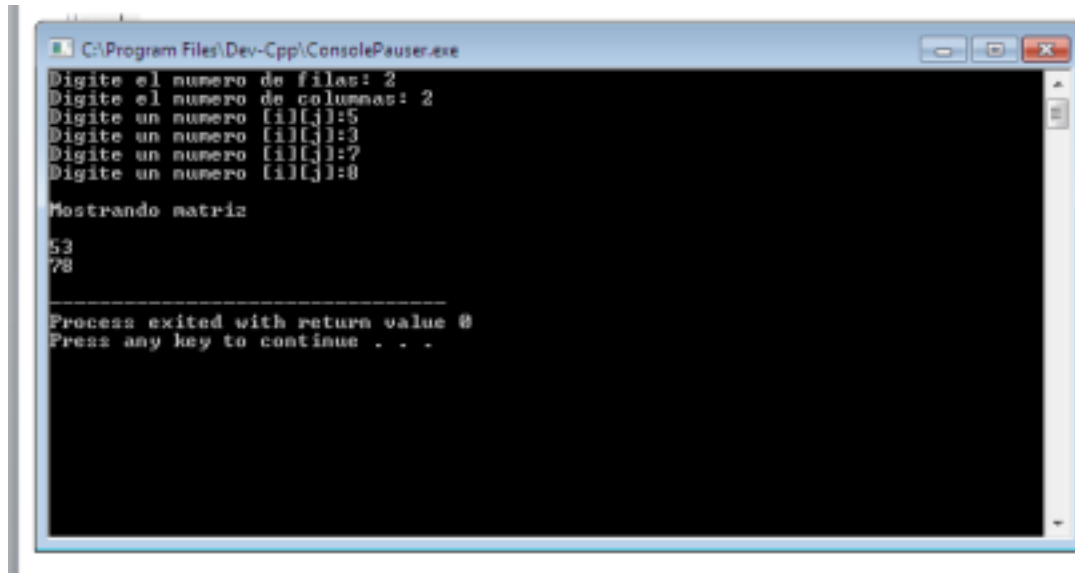


```
actividad1.11.cpp
1  #include<stdio.h>
2  #include<stdlib.h>
3
4  using namespace std;
5
6  int main(){
7      int numeros[150][150],filas,columnas;
8
9      printf("Digite el numero de filas: ");
10     scanf("%i",&filas);
11     printf("Digite el numero de columnas: ");
12     scanf("%i",&columnas);
13
14     //Rellenando la matriz
15     for(int i=0;i<filas;i++){
16         for(int j=0;j<columnas;j++){
17             printf("Digite un numero [%i][%i]:",i,j);
18             scanf("%i",&numeros[i][j]);
19         }
20     }
21
22     printf("\nMostrando matriz\n\n");
23
24     for(int i=0;i<filas;i++){
25         for(int j=0;j<columnas;j++){
26             printf("%i",numeros[i][j]);
27         }
28         printf("\n");
29     }
30
31 }
```

The screenshot shows a C++ IDE with a file named 'actividad1.11.cpp'. The code defines a 150x150 matrix 'numeros' and prompts the user for the number of rows and columns. It then uses nested loops to fill the matrix with user input. After displaying the first matrix, it prompts for the second matrix. The code is currently at line 31, which is the end of the first matrix display loop. The status bar at the bottom indicates 'Sel: 0', 'Lines: 34', 'Length: 610', and 'Done parsing'.

¿Y la segunda matriz?  
Falta la suma

## Compilado



```
C:\Program Files\Dev-Cpp\ConsolePauser.exe
Digite el numero de filas: 2
Digite el numero de columnas: 2
Digite un numero [i][j]:5
Digite un numero [i][j]:3
Digite un numero [i][j]:7
Digite un numero [i][j]:8

Mostrando matriz

53
78

-----
Process exited with return value 0
Press any key to continue . . .
```

## Conclusiones:

Los arreglos permiten manejar de forma sencilla y directa conjuntos de datos del mismo tipo, de los cuales conocemos su cantidad y con los cuales se realizarán operaciones similares.

Un arreglo se puede ver como un conjunto de espacios finitos donde se almacenan elementos. Un arreglo también puede verse como cajas ordenadas en fila y numeradas, donde en cada caja se almacena un solo elemento.