	Carátula para entrega de prácticas	
Facultad de Ingeniería		Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor: Alejandro Pimentel Alarcón

Asignatura: Fundamentos de programació.

Grupo: Bloque 135

No de Práctica(s): Práctica 11

Integrante(s): Partida Arias Emily Rachel

No. de Lista o Brigada: 41

Semestre: 2020-1

Fecha de entrega: Lunes 28 de Octubre

Observaciones:

CALIFICACIÓN: _____

Práctica 11. Arreglos unidimensionales y multidimensionales.

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:

Un arreglo unidimensional es una lista de valores guardados bajo el mismo nombre y del mismo tipo. Cada valor dentro del arreglo se le conoce como elemento del arreglo.

Un arreglo multidimensional es aquel con **n** dimensiones. un arreglo bidimensional se puede representar como una matriz, donde se puede acceder a sus elementos como si fuesen las coordenadas x, y del plano cartesiano, los arreglos con tres dimensiones se manejan con un tercer eje, el eje z.

Desarrollo:

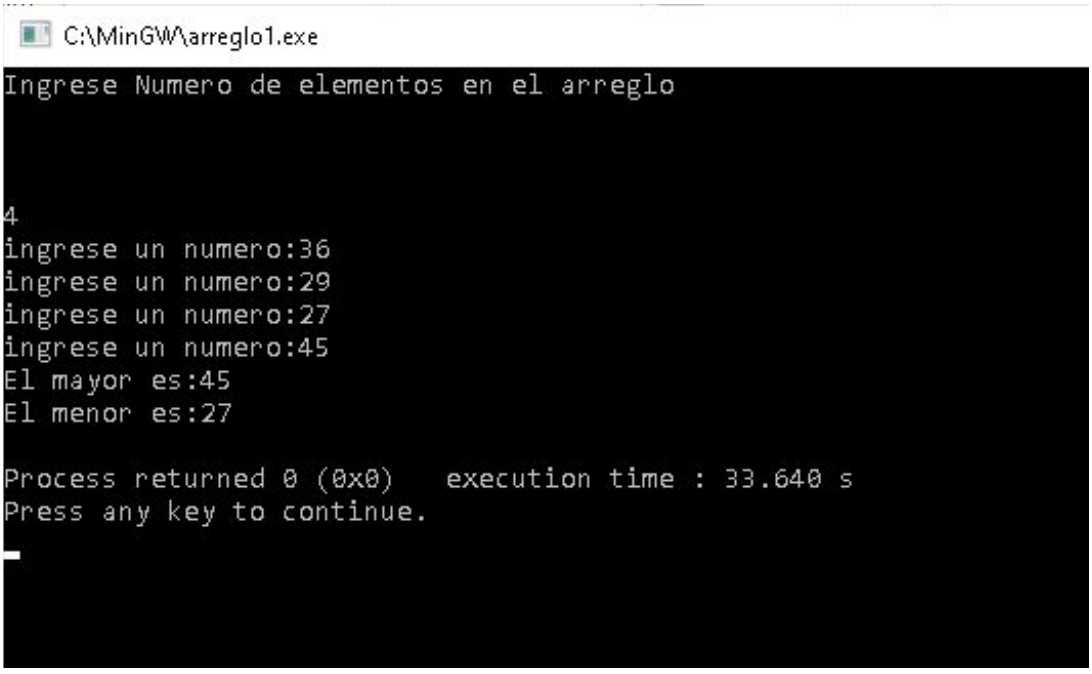
Actividad 1. Hacer un programa que:

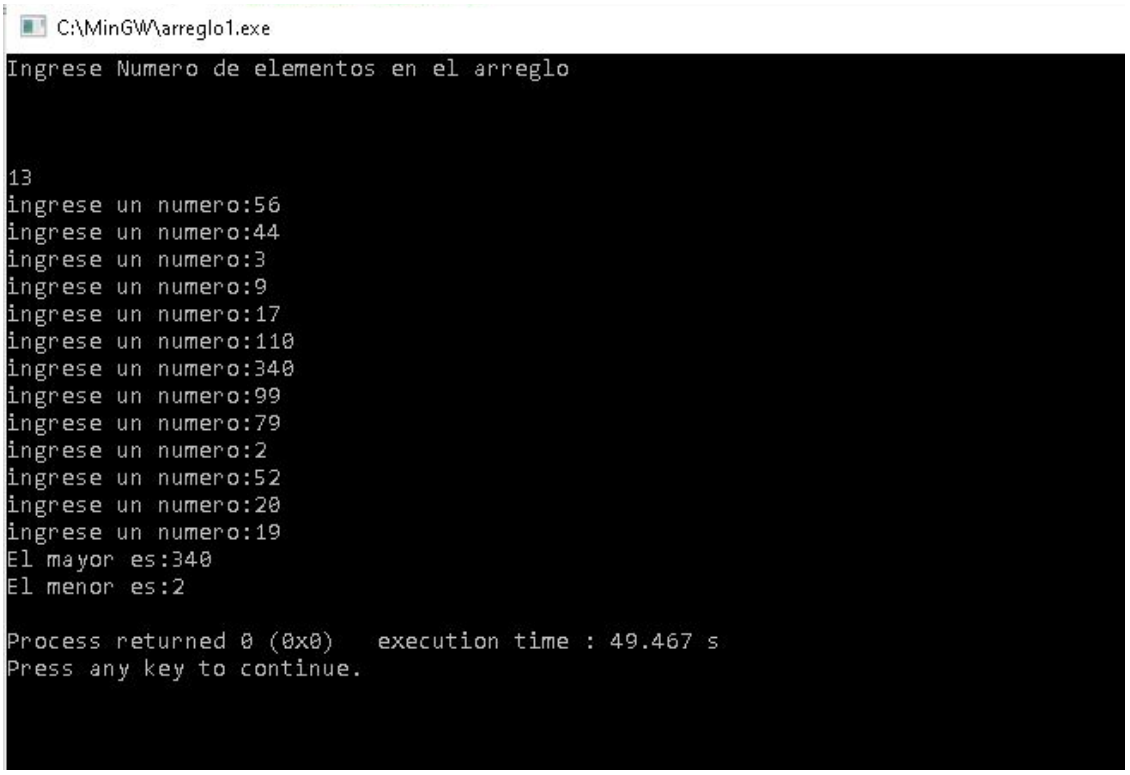
- **Pida al usuario un número.**
- **Genere un arreglo de esa longitud.**
- **Pida al usuario números suficientes para llenar el arreglo.**
- **Muestre al usuario el número menor y el mayor de dicho**
- **arreglo.**

```c
#include <stdio.h>
int main()

{
int a,num;
printf("Ingrese Numero de elementos en el arreglo\n\n");
scanf("%i",&num);
int lista [num];
for(int a=0; a<num; a++)
    {
        printf("ingrese un numero:");
        scanf("%i", &lista[a]);
    }
int valor1=lista[0];
int valor2=lista[0];
for(int a=0; a<num; a++)
       {
         if(lista[a]>valor1)
         {
         valor1=lista[a];
         }
         if(lista[a]<valor2)
         {
         valor2=lista[a];
         }
       }
printf("El mayor es:%i\n", valor1);
printf("El menor es:%i\n", valor2);
return 0;
}
```

Al principio del programa introducimos las variables que utilizaremos, usamos un arreglo en el que el valor de entrada pasa a ser el número de elementos en la lista, mediante scanf vamos recibiendo los datos que se van guardando dentro del arreglo y en la parte final se compara el valor de las listas para que el programa verifique tanto el mayor como el menor elementos del arreglo.

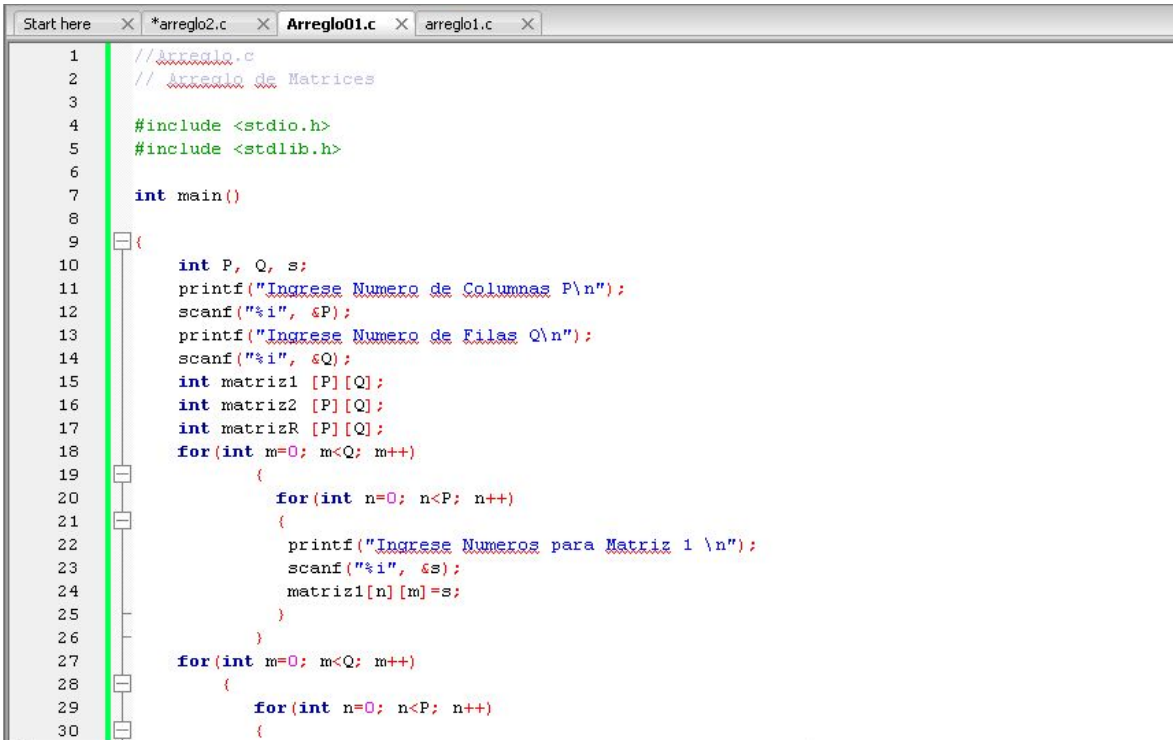
```
C:\MinGW\arreglo1.exe
Ingrese Numero de elementos en el arreglo



10
ingrese un numero:5
ingrese un numero:4
ingrese un numero:6
ingrese un numero:7
ingrese un numero:3
ingrese un numero:8
ingrese un numero:79
ingrese un numero:34
ingrese un numero:100
ingrese un numero:33
El mayor es:100
El menor es:3

Process returned 0 (0x0)   execution time : 41.402 s
Press any key to continue.
```


C:\MinGW\arreglo1.exe
Ingrese Numero de elementos en el arreglo
4
ingrese un numero:36
ingrese un numero:29
ingrese un numero:27
ingrese un numero:45
El mayor es:45
El menor es:27
Process returned 0 (0x0) execution time : 33.640 s
Press any key to continue.

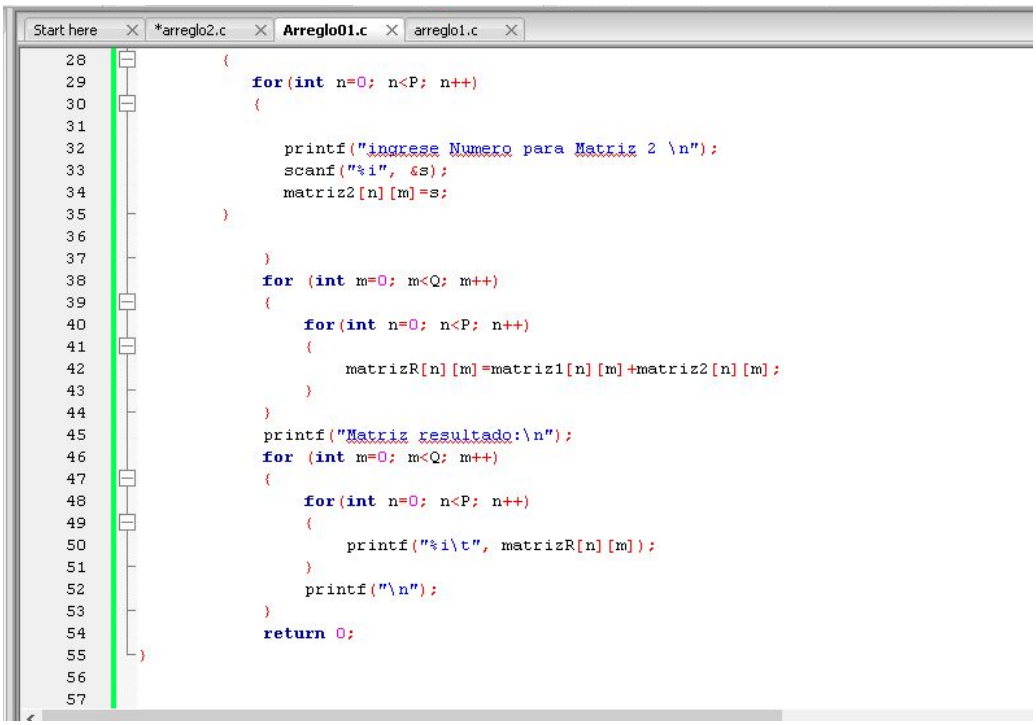

C:\MinGW\arreglo1.exe
Ingrese Numero de elementos en el arreglo
13
ingrese un numero:56
ingrese un numero:44
ingrese un numero:3
ingrese un numero:9
ingrese un numero:17
ingrese un numero:110
ingrese un numero:340
ingrese un numero:99
ingrese un numero:79
ingrese un numero:2
ingrese un numero:52
ingrese un numero:20
ingrese un numero:19
El mayor es:340
El menor es:2
Process returned 0 (0x0) execution time : 49.467 s
Press any key to continue.

Actividad 2. Hacer un programa que:

- Pida al usuario dos números N y M.
- Genere dos matrices de N × M.
- Pida al usuario números suficientes para llenar ambas
- matrices.
- Muestre al usuario la matriz resultado de sumar las dos de Entrada.

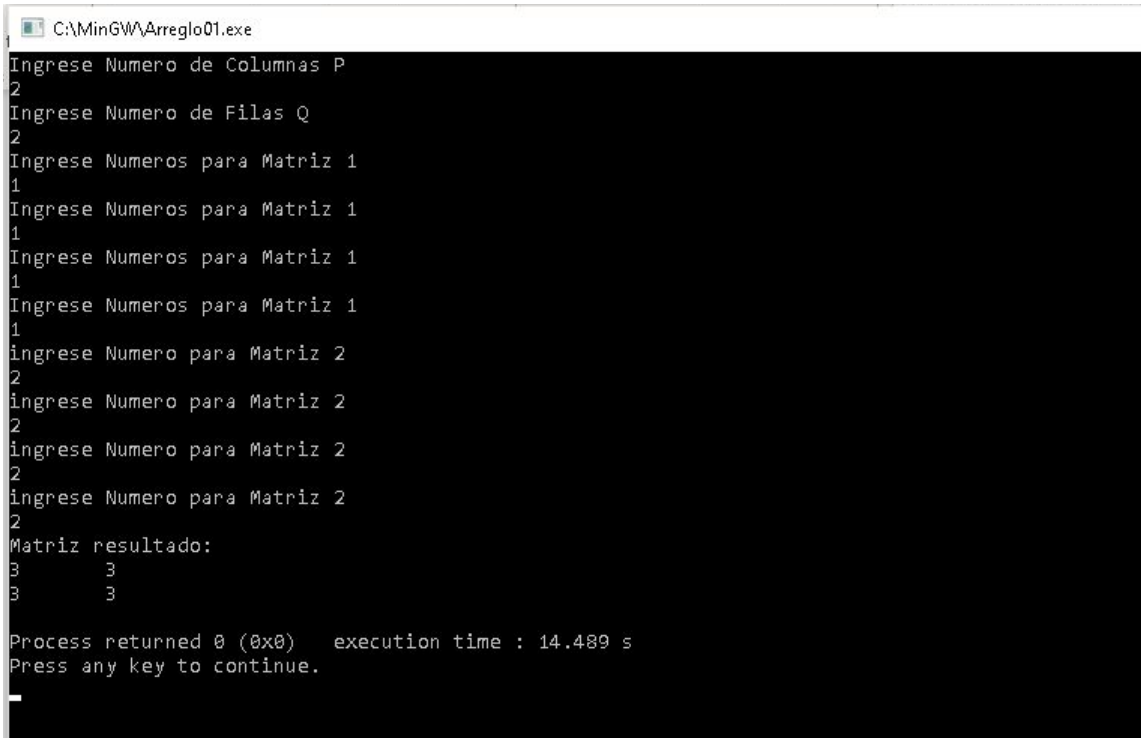


```
1  //Arreglo.c
2  // Arreglo de Matrices
3
4  #include <stdio.h>
5  #include <stdlib.h>
6
7  int main()
8
9  {
10     int P, Q, s;
11     printf("Ingrese Numero de Columnas P\n");
12     scanf("%i", &P);
13     printf("Ingrese Numero de Filas Q\n");
14     scanf("%i", &Q);
15     int matriz1 [P][Q];
16     int matriz2 [P][Q];
17     int matrizR [P][Q];
18     for(int m=0; m<Q; m++)
19           {
20             for(int n=0; n<P; n++)
21             {
22              printf("Ingrese Numeros para Matriz 1 \n");
23              scanf("%i", &s);
24              matriz1[n][m]=s;
25             }
26           }
27     for(int m=0; m<Q; m++)
28         {
29            for(int n=0; n<P; n++)
30            {
```



```
28         {
29            for(int n=0; n<P; n++)
30            {
31
32               printf("ingrese Numero para Matriz 2 \n");
33               scanf("%i", &s);
34               matriz2[n][m]=s;
35         }
36
37           }
38           for (int m=0; m<Q; m++)
39           {
40               for(int n=0; n<P; n++)
41               {
42                   matrizR[n][m]=matriz1[n][m]+matriz2[n][m];
43               }
44           }
45           printf("Matriz resultado:\n");
46           for (int m=0; m<Q; m++)
47           {
48               for(int n=0; n<P; n++)
49               {
50                   printf("%i\t", matrizR[n][m]);
51               }
52               printf("\n");
53           }
54           return 0;
55 }
56
57
```

Introducimos al programa tres variables P Q y s, donde P Y Q serán los arreglos de las filas y columnas de las matrices y s es una variable que realiza la suma de ambas matrices, una vez que se recibe el numero de filas y columnas se hace que el programa vaya corriendo los índices de cada arreglo, para ir recibiendo los datos y que los vaya guardando dentro de él, así para las dos matrices una vez hecho esto; el programa realiza la suma de las matrices imprimiendo el resultado en la pantalla.



C:\MinGW\Arreglo01.exe

```
Ingrese Numero de Columnas P
3
Ingrese Numero de Filas Q
2
Ingrese Numeros para Matriz 1
5
Ingrese Numeros para Matriz 1
3
Ingrese Numeros para Matriz 1
2
Ingrese Numeros para Matriz 1
7
Ingrese Numeros para Matriz 1
10
Ingrese Numeros para Matriz 1
11
ingrese Numero para Matriz 2
17
ingrese Numero para Matriz 2
4
ingrese Numero para Matriz 2
9
ingrese Numero para Matriz 2
5
ingrese Numero para Matriz 2
16
ingrese Numero para Matriz 2
13
Matriz resultado:
22      7       11
12      26      24

Process returned 0 (0x0)   execution time : 46.138 s
Press any key to continue.
```

C:\MinGW\Arreglo01.exe

```
Ingrese Numero de Columnas P
2
Ingrese Numero de Filas Q
4
Ingrese Numeros para Matriz 1
4
Ingrese Numeros para Matriz 1
6
Ingrese Numeros para Matriz 1
7
Ingrese Numeros para Matriz 1
9
Ingrese Numeros para Matriz 1
10
Ingrese Numeros para Matriz 1
22
Ingrese Numeros para Matriz 1
8
Ingrese Numeros para Matriz 1
12
ingrese Numero para Matriz 2
13
ingrese Numero para Matriz 2
25
ingrese Numero para Matriz 2
39
ingrese Numero para Matriz 2
11
ingrese Numero para Matriz 2
8
ingrese Numero para Matriz 2
40
ingrese Numero para Matriz 2
8
ingrese Numero para Matriz 2
12
Matriz resultado:
17      31
46      20
18      62
16      24

Process returned 0 (0x0)   execution time : 72.358 s
Press any key to continue.
```

Referencias:

https://blog.michelletorres.mx/arreglos-unidimensionales-en-c/#targetText=Arreglos%20unidimensionales%20en%20C%2B%2B,de%20un%20nombre%20en%20com%C3%BAn.&targetText=En%20el%20lenguaje%20C%2B%2B%20un,arreglos%20unidimensionales%20(una%20dimensi%C3%B3n).