



## Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

# Laboratorios de computación salas A y B

*Profesor:* MC. Alejandro Esteban Pimentel Alarcón

*Asignatura:* Fundamentos de Programación

*Grupo:* 3

*No de Práctica(s):* Práctica 11

*Integrante(s):* Martínez Marcelino Dalila

*No. de Equipo de  
cómputo empleado:* No. de cuenta: 313080119

*No. de Lista o Brigada:*

*Semestre:* 2020-1

*Fecha de entrega:* Lunes 28 de octubre de 2019

*Observaciones:* Excelente

**CALIFICACIÓN:** 10

## Práctica No. 11

### Introducción:

Esta práctica consistirá en conocer como se elabora un arreglo y los tipos de arreglos que se pueden hacer. El estudiante elaborara arreglos de tipo unidimensional, y también arreglos de tipo matriz, con ayuda de procesos de iteración que recorrerá la lista para llenar los arreglos y que también se utilizara para imprimirlos. Utilizara arreglos para problemas que requieran ingresar varios datos y que se requiera agruparlos.

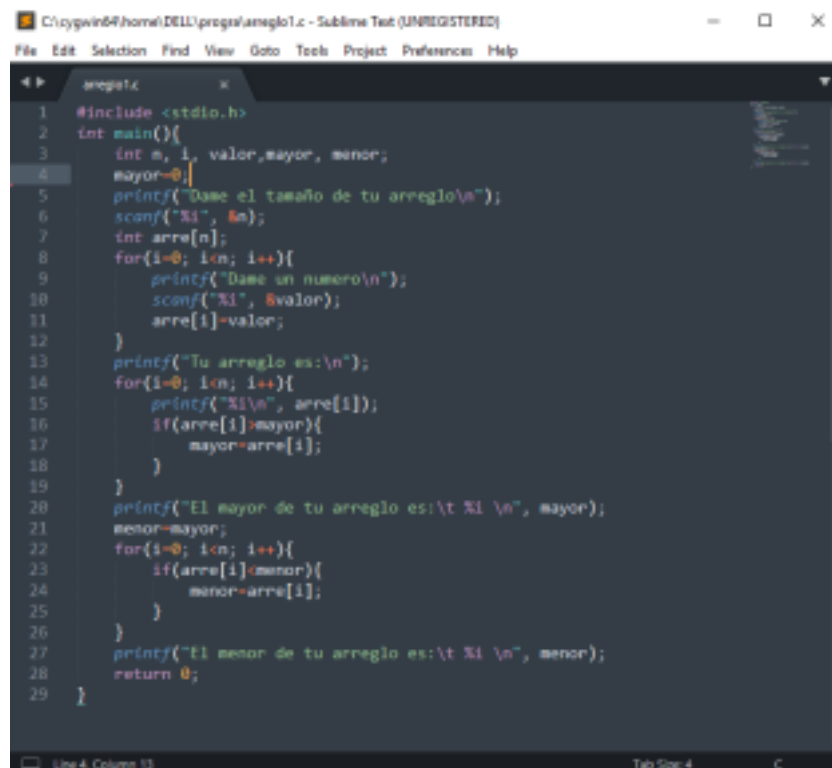
### Objetivo:

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.

### 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.
- 🟢 El programa es el siguiente:



```
C:\cygwin64\home\DELL\programas\arreglo1.c - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

arreglo1.c
1  #include <stdio.h>
2  int main(){
3      int n, i, valor, mayor, menor;
4      mayor=0;
5      printf("Dame el tamaño de tu arreglo\n");
6      scanf("%i", &n);
7      int arre[n];
8      for(i=0; i<n; i++){
9          printf("Dame un numero\n");
10         scanf("%i", &valor);
11         arre[i]=valor;
12     }
13     printf("Tu arreglo es:\n");
14     for(i=0; i<n; i++){
15         printf("%i\n", arre[i]);
16         if(arre[i]>mayor){
17             mayor=arre[i];
18         }
19     }
20     printf("El mayor de tu arreglo es:\t %i \n", mayor);
21     menor=mayor;
22     for(i=0; i<n; i++){
23         if(arre[i]<menor){
24             menor=arre[i];
25         }
26     }
27     printf("El menor de tu arreglo es:\t %i \n", menor);
28     return 0;
29 }
```

- Al compilar el programa y correrlo, se ve de la siguiente forma:

```
DELL@DESKTOP-NJVSQTU ~/progra
$ gcc arreglo1.c -o arrei
DELL@DESKTOP-NJVSQTU ~/progra
$ ./arrei
Dame el tamaño de tu arreglo
5
Dame un numero
8
Dame un numero
9
Dame un numero
3
Dame un numero
10
Dame un numero
4
Tu arreglo es:
8
9
3
10
4
El mayor de tu arreglo es:    10
El menor de tu arreglo es:    3
DELL@DESKTOP-NJVSQTU ~/progra
$
```

- Para poder obtener el mayor de los números en el arreglo, se hace una comparación de dos variables; en este caso la variable mayor y la variable `arre[i]`, inicialmente la variable mayor vale cero. Ambas variables entran en un ciclo for y dentro de el un if que condicione, al hacer la comparación, si la variable `arre[i]` es mayor que la variable mayor, entonces mayor ahora vale lo que Valia la variable `arre[i]`, y esto ocurre sucesivamente hasta que el ciclo for termina, y el ultimo valor guardado en la variable mayor es el que se imprime.
- Para obtener el menor, asignamos el valor de la variable mayor a la variable menor y, volvemos a entrar en un ciclo for y dentro de el un if, que tenga como condición si la variable `arre[i]` es menor que la variable menor, ahora la variable menor valdrá lo que Valia la variable `arre[i]`, esto se repite varias veces mientras se ejecute el ciclo for y se cumpla la condición en if. Cuando el ciclo termina, el ultimo valor guardado en menor es el que se imprime.

## Actividad 2.

Hacer un programa que:

- Pida al usuario dos números N y M.
- Genere dos matrices de N x M.
- Pida al usuario números suficientes para llenar ambas matrices.
- Muestre al usuario la matriz resultado de sumar la dos de entrada.
- El programa queda de la siguiente forma:

```
File Edit Selection Find View Goto Tools Project Preferences Help
arreglo2.c
1 #include <stdio.h>
2 int main(){
3     int i, j, n, m, n1, n2;
4     printf("Dame dos numeros\n");
5     scanf("%i", &n);
6     scanf("%i", &m);
7     int matA[n][m];
8     int matB[n][m];
9     int matR[n][m];
10    for(i=0; i<n; i++){
11        for(j=0; j<m; j++){
12            printf("Dame un numero para la matriz A\n");
13            scanf("%i", &n1);
14            matA[i][j]=n1;
15            printf("Dame un numero para la matriz B\n");
16            scanf("%i", &n2);
17            matB[i][j]=n2;
18            matR[i][j]=matA[i][j]+matB[i][j];
19        }
20    }
21    printf("La matriz A es:\n");
22    for(i=0; i<n; i++){
23        for(j=0; j<m; j++){
24            printf("%i\t", matA[i][j]);
25        }
26        printf("\n");
27    }
28    printf("La matriz B es:\n");
29    for(i=0; i<n; i++){
30        for(j=0; j<m; j++){
31            printf("%i\t", matB[i][j]);
32        }
33        printf("\n");
34    }
35    printf("La matriz resultante de la suma de tus matrices es: \n");
36    for(i=0; i<n; i++){
37        for(j=0; j<m; j++){
38            printf("%i\t", matR[i][j]);
39        }
40        printf("\n");
41    }
42    return 0;
43 }
```

Al compilar y correr el programa se ve de la siguiente forma:

```
~/progra
DELL@DESKTOP-NJV5QTU ~/progra
$ gcc arreglo2.c -o arre2

DELL@DESKTOP-NJV5QTU ~/progra
$ ./arre2
Dame dos numeros
3
4
Dame un numero para la matriz A
5
Dame un numero para la matriz B
6
Dame un numero para la matriz A
7
Dame un numero para la matriz B
3
Dame un numero para la matriz A
9
Dame un numero para la matriz B
2
Dame un numero para la matriz A
0
Dame un numero para la matriz B
3
Dame un numero para la matriz A
7
```

```

Dame un numero para la matriz B
3
Dame un numero para la matriz A
7
Dame un numero para la matriz B
4
Dame un numero para la matriz A
2
Dame un numero para la matriz B
1
Dame un numero para la matriz A
0
Dame un numero para la matriz B
3
Dame un numero para la matriz A
8
Dame un numero para la matriz B
2
Dame un numero para la matriz A
1
Dame un numero para la matriz B
12
Dame un numero para la matriz A
9
Dame un numero para la matriz B
10

```

```

Dame un numero para la matriz A
3
Dame un numero para la matriz B
8
La matriz A es:
5      7      9      0
7      7      2      0
8      1      9      3
La matriz B es:
6      3      2      3
3      4      1      3
2      12     10     8
La matriz resultante de la suma de tus matrices es:
11     10     11     3
10     11     3      3
10     13     19     11
DELL@DESKTOP-KJVSQTU ~/progra
$

```

- Hay dos formas de rellenar las matrices, una de ellas seria hacer dos ciclos for para rellenar las matrices por separado, pero esto también involucraría que tuviéramos que hacer un tercer ciclo para hacer la sumatoria de ambas matrices.
- La otra forma es como lo presento en este programa, utilizando un solo ciclo relleno ambas matrices, pidiendo un numero para la matA y un numero para la matB y este proceso se repite hasta que ambas matrices se encuentran llenas. De esta otra forma también puedo rellenar la matriz resultante de la suma haciendo las operaciones después de que han ingresado los números y guardarlos en matR.
- Aunque el programa solo pide imprimir la matriz resultante de la suma de las matrices de entrada, este programa también imprime las matrices que el usuario genero al ingresar los números, y para ellos utilizo un ciclo for para cada una de las matrices. De esta forma el usuario podrá ver como se ordenó los numero que ingreso inicialmente y si el resultado es correcto.

### Conclusión:

El hacer arreglos no es difícil, puesto que para hacer arreglos unidimensionales solo basta con indicarle de que longitud tiene que ser. Lo mismo pasa para hacer arreglos de tipo matriz, solo hay que entender cómo se van ubicando los números, es decir, como se ordenan sus filas y columnas. Y el rellenar cualquiera de los dos tipos de arreglos resulta ser fácil ya que solo bastara aplicar procesos de iteración hasta que este haya sido llenado por completo. Y finalmente, para imprimir cualquier arreglo haremos uso nuevamente de un proceso de iteración, proceso que hemos estado trabajo con anterioridad.