



## Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

# Laboratorios de computación salas A y B

*Profesor:* Ing. Claudia Rodriguez Espino

*Asignatura:* Fundamentos de Programación

*Grupo:* 1102

*No de Práctica(s):* Práctica número 11

*Integrante(s):* Alejandro Nava Cruz

*Semestre:* 2018-1

*Fecha de entrega:* 12 de Noviembre de 2017

*Observaciones:*

**CALIFICACIÓN:** \_\_\_\_\_

# Practica 11: Arreglos unidimensionales y multidimensionales

## Objetivo:

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

## Desarrollo:

Un arreglo, como su nombre lo dice, es un conjunto de datos de un mismo tipo con un tamaño fijo (ya sea puesto por el usuario o predefinido al momento de crearse).

A cada dato conocido o dato puesto en el arreglo, se le asocia una posición propia o particular, en el cual se requiere indicar para acceder a un elemento en específico. Para facilitar esto, se tiene que lograr a partir de índices.

Como ya se especificó, los arreglos pueden ser unidimensionales o multidimensionales, y estos nos ayudan a facilitar varios procesos y ahorrar líneas de código en el programa.

Cuando se inicializa un arreglo se hace de la manera siguiente:

```
Tipo de dato nom_arreglo[Tamaño del arreglo];
```

# Actividades hechas en casa

## 1. Suma de matrices

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<conio.h>
int MAT1[2][2],MAT2[2][2];
int filas,columnas,x,y;
int main(){
    printf("\t\t\tPrograma para sumar 2 matrices\n\n");
    printf("Hola, ingresa los valores ENTEROS de la matriz 1 y 2 por favor:\n\n");
    printf("MATRIZ 1 (x,y):\n\n");
    for(filas=0;filas<2;filas++) {
        for(columnas=0;columnas<2;columnas++){
            printf("\tMATRIZ 1 (%i,%i)= ",filas+1,columnas+1);
            scanf("%i",&MAT1[filas][columnas]);
        }
    }
    printf("\n");
    printf("MATRIZ 2 (x,y):\n\n");
    for(filas=0;filas<2;filas++) {
        for(columnas=0;columnas<2;columnas++){
            printf("\tMATRIZ 2 (%i,%i)= ",filas+1,columnas+1);
            scanf("%i",&MAT2[filas][columnas]);
        }
    }
    printf("\n\n");
    for(filas=0;filas<2;filas++) {
        for(columnas=0;columnas<2;columnas++){
            printf(" %i ",MAT1[filas][columnas]);
            if(columnas==1) {
                printf("\t+ ");
                for(y=0;y<2;y++) {
                    printf(" %i ",MAT2[filas][y]);
                }
            }
        }
        printf("\n");
    }
    printf("\n\n");
    int MAT3[2][2];
    for(filas=0;filas<2;filas++) {
        for(columnas=0;columnas<2;columnas++) {
            MAT3[filas][columnas]=MAT1[filas][columnas]+MAT2[filas][columnas];
        }
    }
    printf("MATRIZ 3 (x,y):\n\n");
    for(filas=0;filas<2;filas++) {
        for(columnas=0;columnas<2;columnas++){
            printf("\tMATRIZ 3 (%i,%i)= %i\n",filas+1,columnas+1,MAT3[filas][columnas]);
        }
    }
}
```

```
C:\Program Files (x86)\Dev-Cpp\ConsolePauser.exe
Favor de ingresar los valores ENTEROS de la matriz 1 y 2 por favor:

MATRIZ 1 (x,y):
    MATRIZ 1 (1,1)= 2
    MATRIZ 1 (1,2)= 2
    MATRIZ 1 (2,1)= 2
    MATRIZ 1 (2,2)= 2

MATRIZ 2 (x,y):
    MATRIZ 2 (1,1)= 3
    MATRIZ 2 (1,2)= 5
    MATRIZ 2 (2,1)= 5
    MATRIZ 2 (2,2)= 3

2  2      +  3  5
2  2      +  5  3

MATRIZ 3 (x,y):
    MATRIZ 3 (1,1)= 5
    MATRIZ 3 (1,2)= 7
    MATRIZ 3 (2,1)= 7
    MATRIZ 3 (2,2)= 5

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

## 2. Matriz por escalar

```
#include<stdio.h>
#include<conio.h>
int i=0;
int j=0;
int col=0;
int fil=0;
int esc;
main()
{
printf("\t\t\tPrograma para multiplicar una matriz por una escalar\n\n");
printf("Dame el valor de la escalar: ");
scanf("%d",&esc);
printf("Cuantas columnas quieres?: ");
scanf("%d",&col);
printf("Cuantas filas quieres?: ");
scanf("%d",&fil);
int matriz[col][fil];
for(i=0;i<col;i++)
{
    for(j=0;j<fil;j++)
    {
        printf("Dame el valor para cada casilla [%d][%d]",i,j);
        scanf("%d",&matriz[i][j]);
    }
}

printf("\nOriginal");
for(i=0;i<col;i++)
{
    printf("\t\n");
    for(j=0;j<fil;j++)
    {
        printf("%d\t",matriz[i][j]);
    }
}

printf("\t\n");
int resultante[i][j];

printf("\nMultiplicacion de la matriz por la escalar:");
for(i=0;i<col;i++)
{
    printf("\t\n");
    for(j=0;j<fil;j++)
    {
        resultante[col][fil] = matriz[i][j]*esc;
        printf("%d\t",resultante[col][fil]);
    }
}

printf("\n\n\t");
getch();
}
```

```
C:\Program Files (x86)\Dev-Cpp\ConsolePauser.exe
Programa para multiplicar una matriz por una escalar

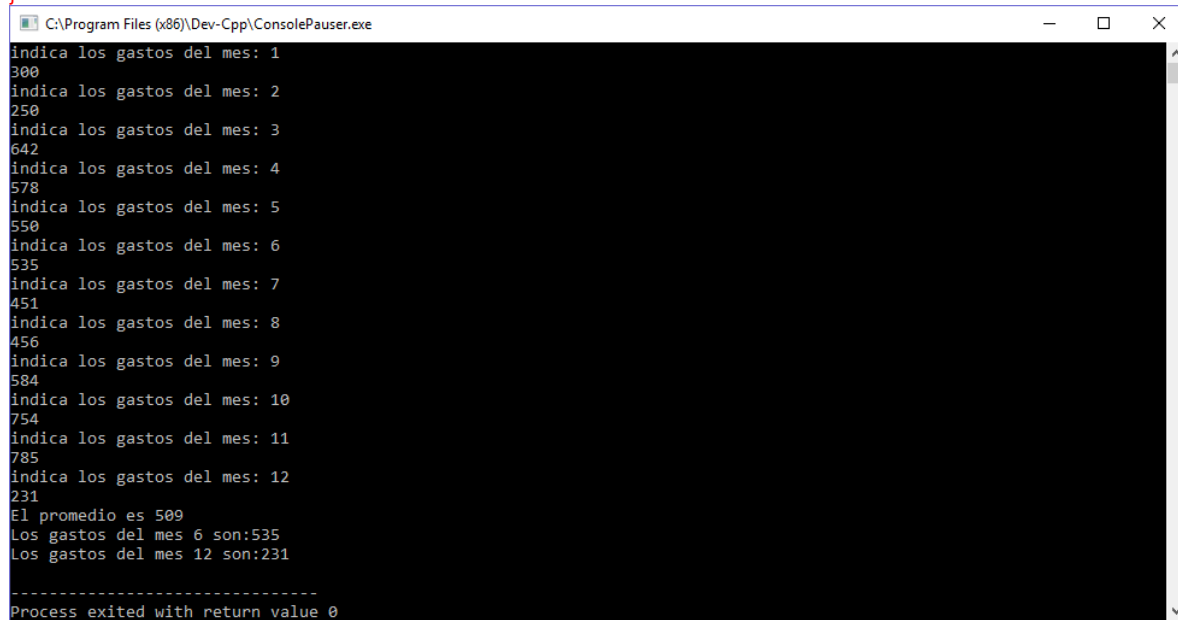
Dame el valor de la escalar: 5
Cuántas columnas quieres?: 2
Cuántas filas quieres?: 2
Dame el valor para cada casilla [0][0]3
Dame el valor para cada casilla [0][1]2
Dame el valor para cada casilla [1][0]1
Dame el valor para cada casilla [1][1]3

Original
3      2
1      3

Multiplicación de la matriz por la escalar:
15     10
5      15
```

### 3. Promedio de gastos mensuales

```
#include<stdio.h>
int mes[15],a,promedio;
main()
{
for (a=1;a<13;a++)
{
printf("indica los gastos del mes: %d\n",a);
scanf ("%d",&mes[a]);
}
for(a=1;a<13;a++)
{
promedio=promedio+mes[a];
}
promedio=promedio/12;
printf("El promedio es %d\n",promedio);
printf("Los gastos del mes 6 son:%d\n",mes[6]);
printf("Los gastos del mes 12 son:%d\n",mes[12]);
}
```



```
C:\Program Files (x86)\Dev-Cpp\ConsolePauser.exe
indica los gastos del mes: 1
300
indica los gastos del mes: 2
250
indica los gastos del mes: 3
642
indica los gastos del mes: 4
578
indica los gastos del mes: 5
550
indica los gastos del mes: 6
535
indica los gastos del mes: 7
451
indica los gastos del mes: 8
456
indica los gastos del mes: 9
584
indica los gastos del mes: 10
754
indica los gastos del mes: 11
785
indica los gastos del mes: 12
231
El promedio es 509
Los gastos del mes 6 son:535
Los gastos del mes 12 son:231
-----
Process exited with return value 0
```

## Conclusiones:

Los arreglos son muy importantes para agrupar elementos de manera sencilla, sirven para distintos motivos y funciones, entre uno de ellos está el ahorrar tiempo y líneas de código.

## Bibliografía:

Práctica número 12, Facultad de Ingeniería, en Laboratorios A y B (2017), Sitio Web:  
<http://lcp02.fib.unam.mx/>