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

**Laboratorio de Computación**

**Salas A y B**

<b>Profesor(a):</b>	<b>César Fabián Domínguez Velasco</b>
<b>Asignatura:</b>	<b>Fundamentos de Programación</b>
<b>Grupo:</b>	<b>15</b>
<b>No de Práctica(s):</b>	<b>10</b>
<b>Integrante(s):</b>	<b>Camacho Duarte Héctor Enrique</b>
	<b>Gutiérrez Esquivel Giovani Emiliano</b>
	<b>Flores Jiménez Diego</b>
	<b>Moreno Chapan Amilet</b>
	<b>Reyes García Raúl de Jesús</b>
<b>No. de lista</b>	<b>07,11,15,16,25,35.</b>
<b>o brigada:</b>	
<b>Semestre:</b>	<b>2024-02</b>
<b>Fecha de entrega:</b>	<b>10/04/24</b>
<b>Observaciones:</b>	
<b>CALIFICACIÓN:</b>	

**Objetivos:**

El alumno utilizará arreglos de dos dimensiones en la elaboración de programas que resuelvan problemas que requieran agrupar datos del mismo tipo, en estructuras que utilicen dos índices.

**Introducción:**

Los arreglos bidimensionales son una estructura fundamental que permite almacenar y manipular conjuntos de datos de manera eficiente y organizada. Un arreglo de dos dimensiones es una colección de datos contiguos del mismo tipo, con un tamaño fijo que se define en el momento de su creación. Este tipo de arreglo se estructura en filas y columnas, lo que facilita el acceso y la manipulación de los datos utilizando dos índices: uno para la fila y otro para la columna.

El uso de arreglos bidimensionales en la programación es crucial para mejorar la eficiencia del código. Agrupar datos del mismo tipo en un arreglo no solo permite una manipulación más clara y ordenada, sino que también optimiza el procesamiento y almacenamiento de la información. Los arreglos bidimensionales son especialmente útiles en aplicaciones que requieren el manejo de grandes cantidades de datos estructurados, como matrices.

En esta práctica de laboratorio se enfocará en el estudio y aplicación de arreglos bidimensionales, aprenderemos a declarar, inicializar y manipular este tipo de estructura en diversos contextos. A través de ejercicios prácticos, se explorarán diferentes métodos para acceder y modificar elementos dentro de un arreglo bidimensional, comprendiendo así su importancia y utilidad en la programación eficiente.

## Programas

### Programa1.c "Impresión de una matriz 3x3"

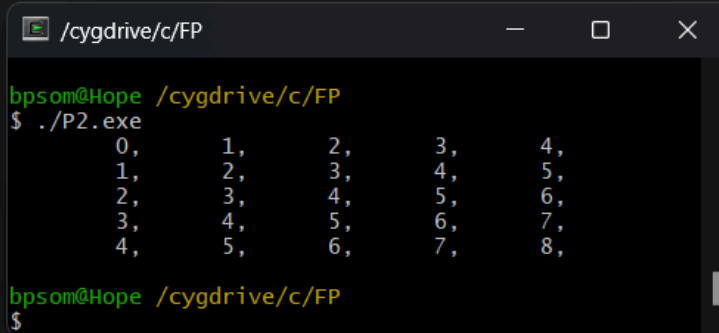
```
C P1.c > main()
1  #include<stdio.h>
2  int main()
3  {
4      int matriz[3][3] = {{1,2,3},{4,5,6},{7,8,9}};
5      int i, j;
6      printf("Imprimir Matriz\n");
7      for (i=0 ; i<3 ; i++) //Representa al renglón del arreglo
8      {
9          for (j=0 ; j<3 ; j++)//Representa a la columna del arreglo
10         {
11             printf("%d, ",matriz[i][j]);
12         }
13         printf("\n");
14     }
15     return 0;
16 }
17
```

```
/cygdrive/c/FP
bpsom@Hope /cygdrive/c/FP
$ gcc P1.c -o P1.exe
bpsom@Hope /cygdrive/c/FP
$ ./P1.exe
Imprimir Matriz
1, 2, 3,
4, 5, 6,
7, 8, 9,
bpsom@Hope /cygdrive/c/FP
$
```

Este programa imprime una matriz utilizando dos ciclos For de forma anidada. En el programa 1B.c imprime exactamente lo mismo a través de dos ciclos While, uno anidado dentro de otro, el contenido de cada elemento de este arreglo es la suma de sus índices. En el programa 1C.c hace lo mismo solo que en este programa utilizamos dos Do-While.

## Programa2.C “Impresión de una matriz 5x5”

```
C P2.c > main()
1  #include<stdio.h>
2  int main()
3  {
4      int i,j,a[5][5];
5      for (i=0 ; i<5 ; i++)//Representa al renglón del arreglo
6      {
7          for (j=0 ; j<5 ; j++)//Representa a la columna del arreglo
8          {
9              a[i][j]=i+j;
10             printf("\t%d, ",a[i][j]);
11         }
12         printf("\n");
13     }
14     return 0;
15 }
```



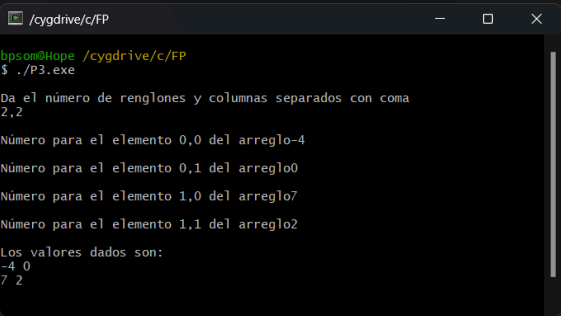
```
/cygdrive/c/FP
bpsom@Hope /cygdrive/c/FP
$ ./P2.exe
0,      1,      2,      3,      4,
1,      2,      3,      4,      5,
2,      3,      4,      5,      6,
3,      4,      5,      6,      7,
4,      5,      6,      7,      8,

bpsom@Hope /cygdrive/c/FP
$
```

En este programa imprimimos una matriz 5x5, utilizando dos ciclos For anidados el primer ciclo For lo utilizamos para representar los renglones, y el segundo para representar las columnas del arreglo. El programa 2b.c y 2c.c realizan lo mismo pero con dos ciclos while y dos ciclos Do-While respectivamente.

### Programa 3.c “Matriz de máximo 10x10”

```
#include <stdio.h>
int main ()
{
    int lista[10][10]; // Se declara el arreglo multidimensional
    int i,j;
    int renglon,columna;
    printf("\nDa el número de renglones y columnas separados con coma\n");
    scanf("%d,%d",&renglon,&columna);
    if(((renglon>=1) && (renglon<=10))&&((columna>=1) && (columna<=10)))
    {
        // Acceso a cada elemento del arreglo multidimensional usando for
        for (i= 0 ; i <= renglon-1 ; i++)
        {
            for(j= 0 ; j <= columna-1 ; j++)
            {
                printf("\nNúmero para el elemento %d,%d del arreglo", i,j );
                scanf("%d",&lista[i][j]);
            }
        }
        printf("\nLos valores dados son: \n");
        // Acceso a cada elemento del arreglo multidimensional usando for
        for (i= 0 ; i <= renglon-1 ; i++)
        {
            for(j= 0 ; j <= columna-1 ; j++)
            {
                printf("%d ", lista[i][j]);
            }
            printf("\n");
        }
        else printf("Los valores dados no es válido");
        printf("\n");
    }
    return 0;
}
```

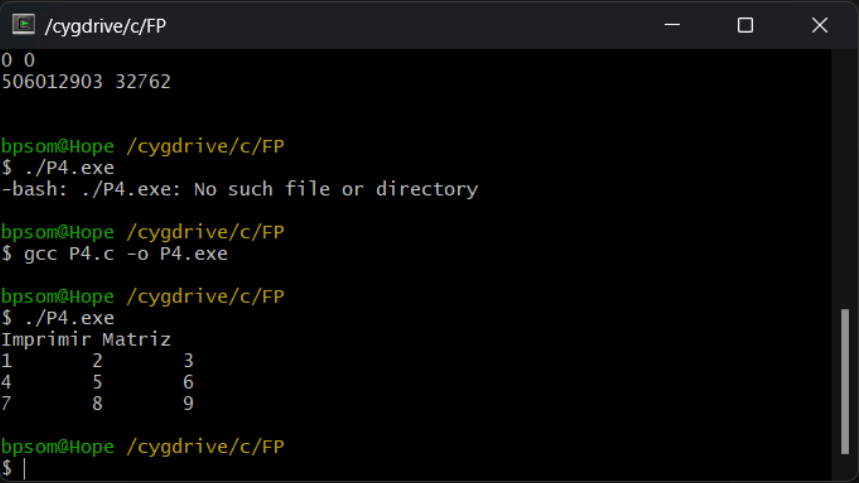


```
bpson@Hope /cygdrive/c/FP
$ ./P3.exe
Da el número de renglones y columnas separados con coma
2,2
Número para el elemento 0,0 del arreglo-4
Número para el elemento 0,1 del arreglo0
Número para el elemento 1,0 del arreglo7
Número para el elemento 1,1 del arreglo2
Los valores dados son:
-4 0
7 2
```

Este programa genera un arreglo multidimensional de máximo 10 renglones y 10 columnas, para poder almacenar datos en cada elemento y posteriormente mostrar el contenido de esos elementos se hace uso de ciclos for anidados respectivamente. Este código se podría replicar con ciclos While y Do-While.

### Programa4.c “Impresión de una matriz de 3x3”

```
C P4.c > main()
1  #include<stdio.h>
2  int main()
3  {
4      int matriz[3][3] = {{1,2,3},{4,5,6},{7,8,9}};
5      int i, cont=0, *ap;
6      ap = *matriz; //Esta sentencia es análoga a: ap = &matriz[0][0];
7      printf("Imprimir Matriz\n");
8      for (i=0 ; i<9 ; i++)
9      {
10         if (cont == 3) //Se imprimió un renglón y se hace un salto de línea
11         {
12             printf("\n");
13             cont = 0; //Inicia conteo de elementos del siguiente renglón
14         }
15         printf("%d\t",*(ap+i)); //Se imprime el siguiente elemento de la matriz
16         cont++;
17     }
18     printf("\n");
19     return 0;
20 }
```



```
/cygdrive/c/FP
0 0
506012903 32762

bpsom@Hope /cygdrive/c/FP
$ ./P4.exe
-bash: ./P4.exe: No such file or directory

bpsom@Hope /cygdrive/c/FP
$ gcc P4.c -o P4.exe

bpsom@Hope /cygdrive/c/FP
$ ./P4.exe
Imprimir Matriz
1 2 3
4 5 6
7 8 9

bpsom@Hope /cygdrive/c/FP
$ |
```

En este último programa imprime una matriz 3x3 haciendo uso de arreglos bidimensionales y de apuntadores, más en concreto accedemos a sus elementos a través de un apuntador utilizando un ciclo For. Los programas 4B.c y 4C.c realizan lo mismo pero al igual que en los anteriores utilizando los ciclos While y Do-While.

## Conclusión

En conclusión, los arreglos bidimensionales representan una herramienta poderosa en la programación, ofreciendo una manera eficiente y organizada de manejar conjuntos de datos del mismo tipo. Al permitir el acceso mediante dos índices, estos arreglos facilitan el procesamiento de información estructurada en filas y columnas, optimizando tanto la claridad como la eficacia del código.

A lo largo de esta práctica de laboratorio, hemos profundizado en la declaración, inicialización y manipulación de arreglos bidimensionales, consolidando nuestra comprensión de su funcionamiento y su aplicación en diversos escenarios.

Consideramos cumplimos el objetivo propuesto en la práctica de acuerdo a lo anterior.

## Bibliografía

Laboratorio Salas A y B. (n.d.). <http://lcp02.fi-b.unam.mx/>

*El lenguaje de programación C. Brian W. Kernighan, Dennis M. Ritchie,*  
*segunda edición, USA, Pearson Educación 1991.*