

# "ARREGLOS"

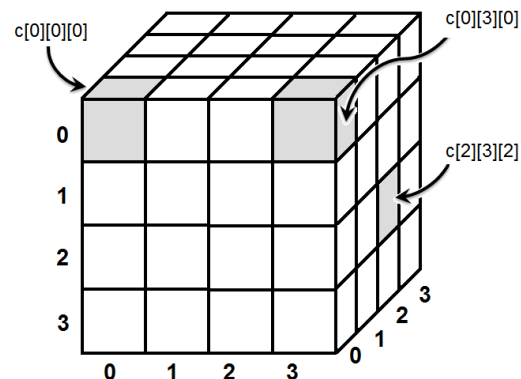
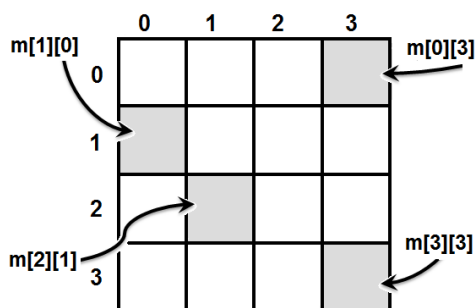
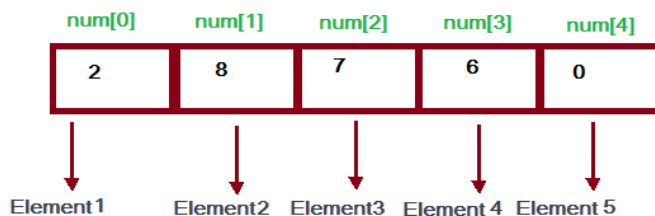
Las variables que hemos utilizado hasta ahora nos permiten el almacenamiento de un solo valor a la vez.

Para resolver cierto tipo de problemas con datos múltiples se requiere almacenamiento en conjunto. A esta organización de elementos se le conoce con el nombre de *arreglo*.

Otra definición de *arreglo* más completa (Luis Joyanes A.), es un conjunto finito y ordenado de elementos homogéneos. La propiedad "ordenado" significa que el elemento primero, segundo, tercero...n-ésimo de un arreglo puede ser identificado. Los elementos de un arreglo deberán ser homogéneos, es decir, del mismo tipo de datos. Por ejemplo, en un arreglo puede todos sus elementos ser de tipo caracter, otro arreglo puede tener sus elementos de tipo entero, etc.

Al tratar el tema de arreglos es necesario conocer el término de *dimensión*.

Dimensión	Descripción
0	Un solo punto.
1	(vector o lista) Una recta. Contiene largo.
2	(matriz o tabla) Contiene largo y ancho.
3	(cubo) Tiene largo, ancho y fondo.



Las operaciones que se pueden realizar con arreglos durante el proceso de resolución de un problema son :

- Asignación de valores
- Lectura/Escritura(capturar/imprimir valores)
- Recorrido (acceso secuencial)
- Actualizar valores
- Ordenar valores
- Búsqueda de valores

### Arreglo de una dimensión (Vectores).

Son aquéllos de una sola dimensión, por lo que también son llamados arreglos Unidimensionales. Los vectores se almacenan en memoria central de la computadora en un orden adyacente. Cada elemento de un vector se puede procesar como si fuese una variable simple al ocupar una posición de memoria.

### Declaración en lenguaje C

**tipo**    nom \_arreglo [tamaño];

donde :

nom\_arreglo :    nombre válido del arreglo.

tamaño :    Cantidad de elementos que puede almacenar

tipo: Tipo de datos de los elementos del arreglo: int, float, char, etc.

Ejemplo de cómo declarar de un vector de 7 elementos, nombre del vector calificacion.

```
int calificacion[7];
```

0	1	2	3	4	5	6

Declaración de un Arreglo de una dimensión también llamado vector.

```
int x[4] ;
```

	5		
0	1	2	3


Asignación de valores

```
x[1]= 5;  
x[0]= 6;  
x[2]= x[1] + x[0] - (2 +1)  
x[3]= x[1] * x[1];
```

## Ejemplos vectores:

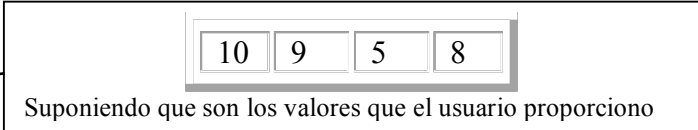
El siguiente programa almacena 4 valores en el arreglo de una dimensión, calcula la suma de los 4 valores almacenados, luego calcula el promedio, finalmente imprime el contenido de la posición 2 del arreglo y el promedio obtenido.

```
int main()  
{  
    int suma;  
    int y[4];  
    float promedio;  
  
    y[0]=8;  
    y[1]=6;  
    y[2]=7;  
    y[3]=2  
    suma=y[0] + y[1] + y[2] + y[3];  
    promedio= suma/4.0;  
    printf("El contenido del casillero #3 es %d", y[2] );  
    printf("El promedio es %f", promedio);  
    return 0;  
}
```



El siguiente programa pregunta por 4 valores, y al final imprime los valores capturados.

```
int main ()  
{  
    int y [4]  
    scanf("%d",&y[0]);  
    scanf("%d",&y[1]);  
    scanf("%d",&y[2]);  
    scanf("%d",&y[3]);  
    printf("%d ",y[0]);  
    printf("%d ",y[1]);  
    printf("%d ",y[2]);  
    printf("%d ",y[3]);  
}
```



Lo que imprime al final: 10 9 5 8

En un problema más complejo, suponiendo que se desea capturar y almacenar en un arreglo de una dimensión 100 valores y al finalizar la captura se requiere calcular el promedio de los 100 números. Este se puede resolver escribiendo 100 veces la sentencia `scanf`, y para realizar la suma se tendría que escribir de manera integro la fórmula:

$suma = y[1] + y[2] + \dots + y[100]$

ya que no se vale usar puntos suspensivos.

Cuando son muchos elementos se pueden usar cualquiera de los ciclos para resolver este tipo de problemas. Una solución se da en el siguiente programa

Usando ciclo **for**

```
int main()
{
    int suma, contador;
    int y[100];
    float promedio;
    for (contador=0; contador< 100 ; contador++)
    {
        printf("Dame el valor #%d",contador);
        scanf("%d", &y[contador]);
    }

    suma=0;
    for (contador=0; contador< 100; contador++ )
    {
        suma= suma + y[contador] ;
    }
    promedio= suma/100.0;
    printf("El promedio es %f", promedio);
}
```

Mismo ejemplo, usando un ciclo while y do-while

```
int main()
{
    int suma, contador;
    int y[100];
    float promedio;
    contador=0;
    while (contador< 100)
    {
        printf("Dame el valor #%d",contador);
        scanf("%d", &y[contador]);
        contador++;
    }

    suma=0;
    contador=0;
    do
    {
        suma= suma + y[contador] ;
        contador++;
    } while (contador <100);

    promedio= suma/100.0;
    printf("El promedio es %f", promedio);
}
```

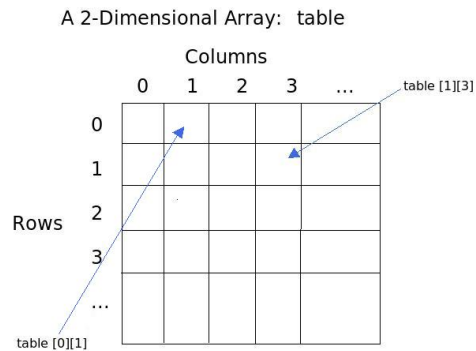
Ejercicios:

Escribir un programa en C, donde se declare un vector de 10 elementos de tipo entero, capturar los 10 elementos en la función main usando un ciclo for, después mandar a llamar una función sumarVector (definido por el usuario) el cual recibe en su argumento de entrada al vector capturado en el main y un valor n que es la dimensión del vector. La función sumarVector debe realizar la suma de todos los valores con un ciclo while, al finalizar regresar el resultado. El resultado se imprime en la función main.

Escribir un programa en C, que contenga una función llamado buscarElemento el cual recibe en su argumento 3 variables, un vector de tipo entero de 20 elementos, un numero entero n. La función debe buscar si el valor n se encuentre entre los valores almacenados en el vector, regresa un 1 si esta, un 0 si no está. En el método main deben capturarse los 20 elementos del vector, preguntar al usuario que valor desea buscar e invocar a la función para saber si lo encuentra.

# Matrices

Son aquéllos de dos dimensiones por lo que también son llamados **arreglos Bidimensionales**. Cada elemento de una matriz se puede procesar como si fuese una variable simple al ocupar una posición de memoria. Las matrices tienen filas y columnas.



Ejemplo, declaración de un arreglo de 4 filas y 5 columnas:

```
int x[4][5];
```

0 1 2 3 4

		8		

**Asignación de valores**

```
x[1][3]= 8
```

Declaración

```
tipo nom_arreglo [tamaño_renglon] [tamaño_columna];
```

donde :

nom\_arreglo : nombre válido del arreglo.

tamaño\_renglon: Cantidad de renglones para la matriz

tamaño\_columna: Cantidad de columnas para la matriz

tipo de dato : tipo de datos de los elementos del arreglo: int, float, char, etc.

La cantidad de elementos que almacena una matriz se obtiene multiplicando el número de renglones por el número de columnas.

## Ejemplos :

El siguiente programa almacena 12 valores en el arreglo de dimensión 3x4, calcula la suma de los 12 valores almacenados, luego calcula el promedio, finalmente imprime el contenido de la posición 2,3 del arreglo y el promedio obtenido.

```
int main()
```

```
{
```

```
    int suma;
```

```
    int y[3][4];
```

```
    float promedio;
```

```
    y[0][0]=8;
```

```
    y[0][1]=6;
```

```
    y[0][2]=7;
```

```
    y[0][3]=8
```

```
    y[1][0]= 2;
```

```
    y[1][1]= 2;
```

```
    y[1][2]= 2;
```

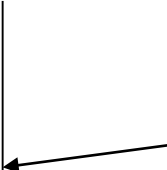
```
    y[1][3]= 2;
```

```
    y[2][0]= 4;
```

```
    y[2][1]= 3;
```

```
    y[2][2]= 2;
```

```
    y[2][3]= 1;
```



8	6	7	8
2	2	2	2
4	3	2	1

```
    suma = y[0][0] + y[0][1] + y[0][2] + y[0][3] + y[1][0] + y[1][1] + y[1][2] +  
y[1][3] + y[2][0] + y[2][1] + y[2][2] + y[2][3];  
    promedio= suma/12.0;  
    printf("El valor almacenado en un casillero en especial es %d", y[2][3]);  
    printf("El promedio es %f",promedio);  
}
```

El siguiente programa pregunta por 12 valores, y al final imprime los valores capturados.

```

int main()
{
    int y[3][4];

    scanf("%d",&y[0][0]);
    scanf("%d",&y[0][1]);
    scanf("%d",&y[0][2]);
    scanf("%d",&y[0][3]);

    scanf("%d",&y[1][0]);
    scanf("%d",&y[1][1]);
    scanf("%d",&y[1][2]);
    scanf("%d",&y[1][3]);

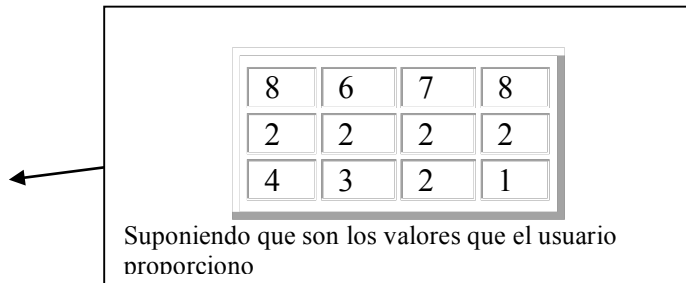
    scanf("%d",&y[2][0]);
    scanf("%d",&y[2][1]);
    scanf("%d",&y[2][2]);
    scanf("%d",&y[2][3]);

    printf("%d ",y[0][0]);
    printf("%d ",y[0][1]);
    printf("%d ",y[0][2]);
    printf("%d ",y[0][3]);

    printf("%d ",y[1][0]);
    printf("%d ",y[1][1]);
    printf("%d ",y[1][2]);
    printf("%d ",y[1][3]);

    printf("%d ",y[2][0]);
    printf("%d ",y[2][1]);
    printf("%d ",y[2][2]);
    printf("%d ",y[2][3]);
}

```



**Lo que imprime al final:**

```

8      6      7      8
2      2      2      2
4      3      2      1

```

En un problema más complejo, suponiendo que se desea capturar y almacenar valores en un arreglo de dimensión 100x200 elementos, y al finalizar la captura se requiere calcular el promedio de todos los números. Este se puede resolver escribiendo 20000 veces la



sentencia leer, y para realizar la suma se tendría que escribir de manera integro la fórmula:

$suma = y[1][1] + y[1][2] + \dots + y[100][200]$

ya que no se vale usar puntos suspensivos.

Se puede usar cualquiera de los ciclos para resolver este tipo de problemas. En arreglos de dos dimensiones o matrices se necesitan 2 ciclos para poder llenar o recorrer la matriz.

Una solución se da en el siguiente algoritmo.

```
int main()
{
    int suma, renglon, columna;
    int numeros[100][200];
    float promedio;

    for (renglon=0; renglon< 100; renglon++)
    {
        for (columna=0; columna<200; columna++)
        {
            scanf("%d", &numeros[renglon][columna]);
        }
    }

    renglon=0;
    suma=0;
    do
    {
        columna=0;
        do
        {
            suma= suma + numeros[renglon][columna] ;
            columna=columna + 1;
        } while (columna<200);
        renglon=renglon+1;
    } while (renglon<100);

    promedio= suma/20000.0;
    printf("El promedio %f", promedio);
}
```

Ejercicio. Escribir un programa que permita capturar los datos de una matrix de 3x5,

ejemplo

4 7 1 3 5

2 0 6 9 7

3 1 2 6 4

Y lo imprima de la siguiente forma:

```
4 2 3
7 0 1
1 6 2
3 9 6
5 7 4
```

Debe funcionar para sin importar que números se capturen.

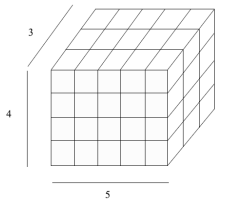
**Actividad de Proyecto.** En un arreglo bidimensional se almacena la cantidad de computadoras vendidos por tres vendedores en cuatro zonas diferentes. Se pide:

- Capturar la cantidad de computadoras que vendió cada vendedor en cada zona
- Imprimir la matriz
- Imprimir en que zona se vendió más computadoras.
- El vendedor que menos computadores vendió.
- La cantidad de computadores vendidos por todos los vendedores en todas las zonas.
- Dado una cantidad de computadoras, ejemplo 4, nos muestre que vendedor y en que zona se vendieron esas 4 computadoras.

## Arreglos de 3 Dimensiones

Declaración

```
entero x[4][5][3];
```



Un array de tres dimensiones (4 x 5 x 3)

```
x[0][3][1]=3
```