MATERIA: Programación I
GUIA DE LABORATORIO #: 22, 23
Fecha:

## COMPETENCIAS:

Descripción de los arreglos unidimensionales y la resolución de problemas de programación que involucren arreglos.

## TEORÍA ASOCIADA:

### 1. INTRODUCCIÓN

Un arreglo bidimensional es una estructura de datos básica que consta de un conjunto de celdas dispuestos de manera rectangular, con n filas y m columnas. Cada celda se identifica con dos índices: el número de fila y el número de columna.

En el lenguaje C++, cada celda del arreglo contiene un único valor del tipo preestablecido. Los contenidos de las celdas pueden ser gestionados (para lectura o escritura) como si se tratase de variables independientes.

Una característica de los arreglos bidimensionales, es que, al igual que los arreglos unidimensionales, sus elementos ocupan áreas de memoria RAM consecutivas, por ese motivo estos arreglos son inmutables, es decir, no pueden cambiar su tamaño con el que son declarados inicialmente.

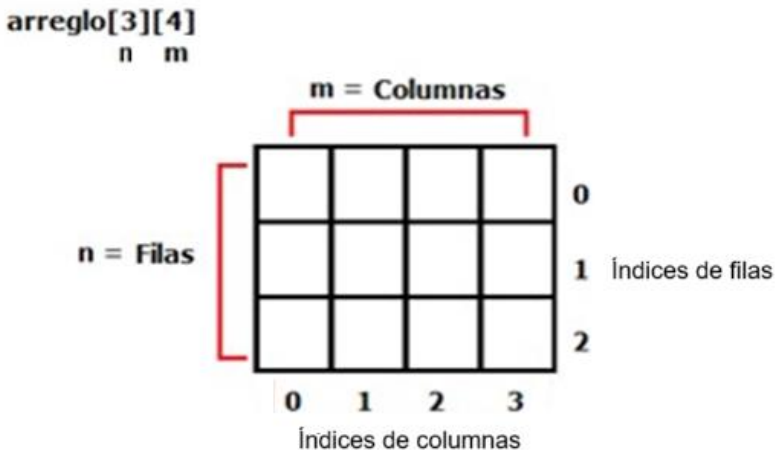
En C++, la declaración de un arreglo usa la siguiente sintaxis:

```
tipoDato identificador[n][m];
```

donde n y m corresponden al número de filas y al número de columnas que tendrá el arreglo.

Por ejemplo, la declaración de un arreglo bidimensional: matriz de 3 fila y 4 columnas que almacenen valores de tipo: int, es:

```
Int arreglo[3][4];
```



Los arreglos bidimensionales, de forma similar a los arreglos unidimensionales, pueden ser inicializados al momento de su declaración, de la forma:

```
int arreglo[3][4]={}   //inicializa todos los elementos a 0.

int arreglo[3][4]={{1, 2}, {}, {1,2,3,4}}

//inicializa parcialmente con los valores indicados, el resto de los elementos no indicados adoptan valores iguales a 0.
```

Cada elemento del arreglo tiene todas las características y atribuciones de una variable simple y como tal, puede ser gestionada para lectura y/o escritura. Por ejemplo para asignar e imprimir el elemento de índice 2 y columna 1, la sintaxis es:

```
cin >> arreglo[2][1];

cout >> arreglo[2][1];
```

Se debe cuidar de no desbordar (sobrepasar) el rango válido de los índices de un arreglo, pues C++ no comprueba los límites e invade las zonas de memoria RAM aledañas al arreglo, obteniéndose resultados incorrectos.

Una de las operaciones muy usadas es el recorrido, que consiste en acceder a cada elemento del arreglo, y realizar alguna operación con ellos.

El recorrido más empleado, denominado recorrido natural, es el que se realiza por filas, desde la primera fila hasta la última, en cada fila se accede a los elementos de izquierda a derecha. Para realizar esta operación, se utiliza dos estructuras repetitivas anidadas.

## EJERCICIOS RESUELTOS:

Problema1:
Escribir un programa que lea un número natural: n (0 < n < 10), declare un arreglo bidimensional de tamaño n x n (de grado n), cargue el mismo con los números enteros consecutivos desde el 1 hasta n x n, y los imprima de forma matricial.
EJEMPLO DE ENTRADA
3
EJEMPLO DE SALIDA
1 2 3
4 5 6
7 8 9

```
#include <iostream>

using namespace std;
int main() {
    int n;
    cin >> n;
```

```
    int matriz[n][n];
    int dato=1;
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < n; ++j) {
            matriz[i][j] = dato;
            dato++;
        }
    }
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < n; ++j) {
            cout<< matriz[i][j]<<" ";
        }
        cout<<endl;
    }
}
```

Problema2:
Escribir un programa que lea un número natural: n (0 < n < 10), cargue una matriz de orden n con los valores correspondientes a la matriz identidad, luengo la imprima de forma matricial.

EJEMPLO DE ENTRADA
5
EJEMPLO DE SALIDA
1 0 0 0 0
0 1 0 0 0
0 0 1 0 0
0 0 0 1 0
0 0 0 0 1

```
#include <iostream>

using namespace std;
int main() {
    int n;
    cin >> n;
    int matriz[n][n]={};
    for (int i = 0; i < n; ++i)
            matriz[i][i] = 1;
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < n; ++j)
            cout<< matriz[i][j]<<" ";
        cout<<endl;
    }
}
```

Problema3:
Escribir un programa que lea un número natual: n (0 < n < 100), cargue una matriz de tamaño nxn con números aleatorios de cuatro dígitos, luego busque el elemento de mayor valor, luego lo imprima en la primera línea y a continuación imprima los índices de fila y columna donde se ubica el mismo, en la segunda línea.
Por ejemplo, suponiendo que n=3 y los valores de matriz generada es:

7233 6324 8432
2943 5933 2934
9342 8423 4931

EJEMPLO DE ENTRADA
3
EJEMPLO DE SALIDA
9342
2, 0

```
#include <iostream>
using namespace std;
int main()
{
    int n;
    cin >> n;
    srand(time(NULL));
    int matriz[n][n];
    for (int i = 0; i < n; ++i)
        for (int j = 0; j < n; ++j)
            matriz[i][j]=1000+rand()%9000;
    int fila=0, columna=0;
    for (int i = 0; i < n; ++i)
        for (int j = 0; j < n; ++j)
            if(matriz[i][j]>matriz[fila][columna])
            {
                fila=i;
                columna=j;
            }
    cout<< matriz[fila][columna]<<endl;
    cout<<fila<<" "<<columna<<endl;

}
```

## EJERCICIOS PARA RESOLVER EN CLASES

Problema4:
Escribir un programa que lea un número natual: n (0 < n < 100), cargue una matriz de tamaño nxn con números aleatorios de dos dígitos, luego imprima los elementos de la diagonal principal, en la primera línea, y los elementos de la diagonal secundaria en la segunda línea.

Suponiendo que n=4 y los valores generados aleatoriamente son:
48 22 59 72
94 92 71 23
45 63 95 12
83 52 43 41

EJEMPLO DE ENTRADA
4
EJEMPLO DE SALIDA
Diagonal principal: 48 92 95 41
Diagonal secundaria: 72 71 63 83

Problema 5:
Escribir un programa que lea un número natural: n (0 < n < 100), cargue el triángulo inferior de una matriz de tamaño nxn con números aleatorios de tres dígitos, y a continuación imprima la matriz.

EJEMPLO DE ENTRADA
3
EJEMPLO DE SALIDA
548 0 0
694 192 0
345 563 295

Problema6:
Escribir un programa que lea un número natural: n (0 < n < 100), cargue una matriz de tamaño nxn con números aleatorios de 2 dígitos, luego crea otra matriz de orden nxn, copie los datos de la primera matriz en la segunda de forma que aparezcan rotados 90º en el sentido de las agujas del reloj y por último imprima ambas matrices separadas por una línea en blanco entre ellas.
Si n=4 y la matriz con números aleatorios fuese:
84 68 23 41
93 63 43 55
34 56 30 73
28 15 32 54
EJEMPLO DE ENTRADA
4
EJEMPLO DE SALIDA
84 68 23 41
93 63 43 55
34 56 30 73
28 15 32 54

28 34 93 84
15 56 63 68
32 30 43 23
54 73 55 41

## EJERCICIOS PARA RESOLVER EXTRACLASE:

Problema7:
Reescribir el programa del problema 6, empleando solo una matriz.
Si n=4 y la matriz con números aleatorios fuese:
84 68 23 41
93 63 43 55
34 56 30 73
28 15 32 54
EJEMPLO DE ENTRADA
4
EJEMPLO DE SALIDA
84 68 23 41
93 63 43 55
34 56 30 73
28 15 32 54

28 34 93 84
15 56 63 68
32 30 43 23
54 73 55 41

Problema8:
Escribir un programa que lea un número n (0 < n < 100), y cargue una matriz de orden nxn con valores iguales a 1, excepto los cuatro bordes que deberán tener valores iguales a 0; por último, imprimir esa matriz.
EJEMPLO DE ENTRADA
4
EJEMPLO DE SALIDA
0 0 0 0 0
0 1 1 1 0
0 1 1 1 0
0 1 1 1 0
0 0 0 0 0

Problema9:
Escribir un programa que lea un número entero: n (0 < n < 100), los cargue con valores aleatorios correspondiente a letras minúsculas y mayúsculas del idioma inglés y los imprima.

EJEMPLO DE ENTRADA
4
EJEMPLO DE SALIDA
R t s K
e Z t O
y p L a
Q w b n

Problema10:
Escribir un programa que lea un número entero: n (0 < n < 15), lea las notas de n estudiantes y las cargue en una matriz de n filas y 2 columnas, luego imprima los datos de la matriz y por último calcule el promedio de todas las notas y lo imprima, de acuerdo al formato del ejemplo de salida.

EJEMPLO DE ENTRADA
5
Antonio Gallardo
67
Beatriz Antezana
83
Jose Pairumani
43
Antonia Salas
72
Pablo Tintaya
82
EJEMPLO DE SALIDA
Antonio Gallardo: 67
Beatriz Antezana: 83
Jose Pairumani: 43
Antonia Salas: 72
Pablo Tintaya: 82
PROMEDIO: 69.4