	<p style="text-align: center;">UNIVERSIDAD DON BOSCO FACULTAD DE INGENIERIA ESCUELA DE COMPUTACIÓN</p>
<p style="text-align: center;">Ciclo II</p>	<p style="text-align: center;">Programación de Algoritmos Guía de Laboratorio No. 9 Arreglos bidimensionales (matrices)</p>

I. RESULTADOS DE APRENDIZAJE

Que el estudiante:

- Elabore aplicaciones avanzadas en C++ que utilicen arreglos y matrices
- Domine el uso de arreglos para solución de problemas complejos
- Utilice los arreglos de cadenas de caracteres

II. INTRODUCCIÓN

Los arreglos en C++ pueden tener múltiples subíndices. Un uso común de los arreglos con múltiples subíndices es representar *tablas de valores*, las cuales contienen información organizada en filas y columnas.

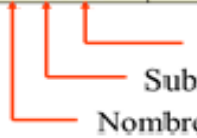
Para identificar un elemento en especial de una tabla, debemos especificar dos subíndices: el primero identifica la fila del elemento, y el segundo identifica la columna del elemento.

En general, a un arreglo con m filas y n columnas se llama *arreglo de m por n*, así: **m x n**

En la Imagen 9.1 se muestra un arreglo a con doble subíndice. Este arreglo contiene tres filas y cuatro columnas, de manera que se dice que es un arreglo de 3 por 4,

Imagen 9.1: Definición de los elementos de un arreglo bidimensional

	Columna 0	Columna 1	Columna 2	Columna 3
Fila 0	A[0][0]	A[0][1]	A[0][2]	A[0][3]
Fila 1	A[1][0]	A[1][1]	A[1][2]	A[1][3]
Fila 2	A[2][0]	A[2][1]	A[2][2]	A[2][3]



Cada elemento del arreglo *a* se identifica en la figura con el nombre del elemento de la forma *a[i][j]* tal como se hacía con los vectores *a[i]*.

- ✓ *a* es el nombre del arreglo.
- ✓ *i* , *j* son los subíndices que identifican de manera única a cada elemento de *a*.

Observe que los elementos en la primera fila tienen un primer subíndice 0; los nombres de los elementos en la cuarta columna tienen un segundo subíndice 3.

Declaración de matrices

El lenguaje C++, permite al programador declarar matrices de cualquier tipo y prácticamente de cualquier tamaño. En el pseudolenguaje, una matriz se declara usando el siguiente formato:

<TIPOdeDato><NOMBRE> [<N>][<M>]

En este formato aparecen en mayúsculas y entre los caracteres < y > los componentes que el programador puede determinar. Así por ejemplo, si se quiere declarar una matriz con nombre *matriz*, de dimensión 15x4 y que pueda almacenar datos de tipo carácter, se debe escribir la siguiente línea.

char matriz [15][4];

Ejemplo:

Declarar arreglo bidimensional llamado *R* que almacene datos tipo *double*, y que tenga 6 (primera dimensión) por 4 (segunda dimensión) elementos, es decir, un total de $(6 \times 4) = 24$ elementos. Para lograrlo basta esta instrucción:

double R[6][4];

A este arreglo *R* lo podría ver como una “tabla”, el cual tendrá 6 filas por 4 columnas, tal como se muestra a la derecha:

Para guardar valor 6.241 en la posición indicada por la X de la figura 2, se hará así:

R [1][3] = 6.241;

Figura 2: Elementos del arreglo bidimensional *R*

	0	1	2	3
0				
1				X
2				
3				
4				
5				

Cadenas de caracteres en C++

Una cadena de caracteres en C++ es un arreglo unidimensional de caracteres.

Cada posición del arreglo almacena un carácter de la cadena y a continuación del último carácter se añade un elemento más que contiene el carácter nulo ('`\0`'). El carácter nulo sirve como indicador de fin de cadena.

Ejemplo para declaración de cadenas:

- `char DireccionCasa[25];` //Array de 25 caracteres como máximo.
- `char nombre[21];` //arreglo para un máximo de 21 caracteres.

Los arrays de caracteres pueden inicializarse cuando se declaran. Puede hacerlo de 2 formas:

- A. Asigna valores a cada elemento o directamente mediante una cadena de caracteres.

Por ejemplo:

```
char nom[ ]={'A','l','i','c','i','a','\0'};
```

En este caso, no es necesario indicar el tamaño del arreglo. El compilador asume que el arreglo tiene tantos elementos como los valores iniciales brindados.

Otro ejemplo:

```
char c[15]={'M','a','r','t','a','\0'};
```

En este caso, se define un tamaño de 15 caracteres pero solo asigna 6 caracteres. Las posiciones restantes guardan un valor aleatorio

:

- B. Inicializa un vector de caracteres, asignado una cadena de texto directamente.

Por ejemplo:

```
char saludo[ ] = "Hello my friend";
```

En este caso, el carácter nulo se añade automáticamente luego del carácter 'd' del texto.

Otro ejemplo:

```
char flor[14] = "Orquidia";
```

Define un arreglo para 14 caracteres como máximo, pero asigna 9 posiciones (incluyendo el carácter nulo, que C lo incluye por defecto).

Por lo tanto, los arreglos de caracteres en C/C++ son de **tamaño fijo y longitud variable**. Lo que significa que cuando se declara un arreglo de caracteres, debe indicarse el tamaño máximo del vector (número máximo de caracteres, más el carácter nulo). Pero el arreglo podrá contener una cadena más corta.

El arreglo tendrá un tamaño fijo, pero su longitud será la de la cadena que contiene en cada momento.

Metodos de cin para manejo de cadenas de caracteres

cin.get()

Permite leer solo el primer carácter que ingresa el usuario cuando se le pide ingresar una palabra y después mostrarlo en pantalla. Este método requiere solamente declarar una variable tipo char.

Para leer una cadena, necesita a un ciclo, para capturar cada carácter y completar el texto en el vector.

```
1  #include <iostream>
2  using namespace std;
3  int main(void){
4      char cadena[10];
5      int i; //contador de posicion en arreglo cadena
6      for(i=0;i<9;i++) // máximo 9 caracteres
7          cin.get(cadena[i]); // se lee por teclado un carácter y
8                               // se guarda en arreglo
9      cadena[i]='\0'; // se debe añadir caracter nulo al final
10     cout << "cadena ingresada: " << cadena << endl;
11 }
```

cin.getline()

Lee una serie de caracteres desde el flujo de entrada y los almacena en un arreglo de caracteres. El formato de uso de getline es el siguiente:

```
cin.getline(cadena, num, carácter_final);
```

En donde **cadena** es el array donde se almacenan los caracteres de la cadena digitados por usuario.

Se leen caracteres hasta el final del flujo, hasta el primer carácter que coincide con el **carácter_final** especificado, el cual se descarta ó hasta que se han leído **num-1** caracteres.

getline añade el carácter nulo (\0) al final de la cadena extraída.

Ejemplo:

Dado el arreglo: `char cargo[15];`

La siguiente línea invoca a getline para leer por teclado a la cadena que se almacenara en este arreglo:

```
cin.getline(cargo, 15, '\n');
```

El carácter final se puede omitir, si no aparece se supone que sera '\n':

```
cin.getline(cargo, 50);
```

Guardar una lista de cadenas de caracteres

Al igual que puede guardar conjuntos de valores numéricos en matrices, tambien puede guardar conjuntos de cadenas de caracteres.

Una de las formas para guardar listas de cadenas de texto, es crear una matriz bidimensional (tabla) de tipo char:

```
char lista[n][m];
```

La primera dimensión [n] indica la cantidad de cadenas de caracteres que se pueden almacenar y la otra dimensión [m] indica la máxima longitud (tamaño) de caracteres que se podrá guardar en cada una de ellas.

Observe el siguiente ejemplo:

```
//Tabla para almacenar 4 cadenas caracteres
//cada una tendra una longitud max de 25 caracteres
char nom[4][25];
cout<<"Ingrese 4 cadenas de caracteres:\n";
for(int f=0;f<4;f++){
    cout<<"Cadena #"<<f+1<<" de la lista: ";
    cin.getline(nom[f],25); //Lee una cadena de caracteres
}
cout<<"\n\nCadenas ingresadas son:\n";
for(int f=0;f<4;f++) cout<<nom[f]<<endl;
```

III. MATERIALES Y EQUIPO

No.	Requerimiento	Cantidad
1	Memoria USB	1
2	Computadora con compilador de C++	1

IV. PROCEDIMIENTO

PARTE 1: Acceso a las posiciones de una matriz bidimensional

1. Proceda a crear una carpeta denominada **PALguia9_SUCARNET**, en el cual guardará cada uno de los siguientes códigos fuentes C++ que desarrollara en el resto de este procedimiento.
2. Prepara un nuevo archivo de código fuente y guárdelo en la carpeta del paso anterior, bajo el nombre **PAL_Guia9_E1**.
3. El programa a digitar a continuación le permitirá inicializar el valor de cada una de las posiciones de una matriz de 12 valores decimales, distribuidos en 2 dimensiones (3 filas por 4 columnas).

Luego se solicitan a usuario los 12 valores correctos. Finalmente, se le demuestra que los valores fueron ingresados apropiadamente a la matriz.

PAL_Guia9_E1.cpp

```
#include<iostream>
#include<conio.h>
#include<stdlib.h>
using namespace std;
main(){
    //declaracion e inicilizacion de valores de una matriz
    float n[3][4]={
        {0,0,0,0},
        {0,0,0,0},
        {0,0,0,0},
    };
};
```

```

//variables para uso de ciclos e indices de la matriz
int fil,col; //contador filas y columnas

//impresion del contenido inicial de la matriz
cout<<"La matriz n[3][4] contiene actualmente estos valores:";
for(fil=0;fil<3;fil++){ //recorre cada fila
    //accede a cada columna de fila(fil) actual
    cout<<"\n\nfila " <<fil<<": ";
    for(col=0;col<4;col=col+1)
        cout<<n[fil][col]<<"\t";
} //fin for fil
cout<<"\n\nPresiona cualquier tecla para continuar";
getch();
system("cls");

//Valores dados por usuario a una matriz
cout<<"\n\nDigite el valores de cada posicion de la matriz";
for(fil=0;fil<3;fil++){ //recorre cada fila
    cout<<"\n\nValores de fila " <<fil<<":\n";
    //recorre elementos de fila actual
    col=0; //columna 0
    do{
        cout<<"\nValor de posicion (" <<fil<<"," <<col<<")?? ";
        cin>>n[fil][col];
        col++; //incrementa contador columnas
    }while(col<4);
} //fin for fil
//imprime nuevamente contenido matriz con los valores
//dados por usuario
cout<<"\n\nLa matriz n[3][4] contiene ahora estos datos:";
for(fil=0;fil<3;fil++){ //recorre cada fila
    //accede a cada columna de la fila(fil) actual
    cout<<"\nfila " <<fil<<": ";
    for(col=0;col<4;col=col+1) cout<<n[fil][col]<<"\t";
} //fin for fil
cout<<"\n\nPresiona cualquier tecla para finalizar este programa";
getch();
} //fin main

```

4. Compila el programa y confirma que este funciona correctamente.
5. Haz una copia del archivo anterior bajo el nombre **PAL_Guia9_E1_modi**
6. Modifica el código original de tal forma que solamente se solicite al usuario el valor de cada posición de la matriz n y finalmente, se muestre cada uno de los siguientes resultados del análisis de valores de la matriz:
 - a) Cantidad de valores positivos almacenados
 - b) Porcentaje de valores negativos de todo el listado
 - c) Mayor valor de las posiciones de la última columna de la matriz
7. Compile el código modificado en el paso anterior y compruebe que se obtienen los resultados solicitados.

PARTE 2: Simulación de una base de datos con varias matrices

8. Prepare un nuevo archivo .cpp, para desarrollar ahí al siguiente código fuente y guardar el archivo.

PAL_Guia9_E2.cpp
<pre>// Programa para obtener los precios de 4 categorias de productos #include <iostream> using namespace std; #include <conio.h> #include <stdlib.h> #include <stdio.h> //total de .. #define CAT 4 //..categorias #define PRO 3 //.. productos por categorias main(){ //Categorias de productos char categorias[CAT][20]= { "Lamparas", "Escolares", "Herramientas", "PC" }; };</pre>

```

//Nombres de productos
//cada grupo de 3 elementos son productos de una categoria especifica
char nombres[CAT*PRO][50] = {
    "de cristal","de plastico","de barro",
    "Cuaderno","Bolson","Lapices de Colores",
    "Martillo","Serrucho","Taladro",
    "HP", "BenQ", "Toshiba"
};

//Precios de productos
//cada grupo de 3 precios son productos de una categoria especifica
float precios[CAT][PRO]= {
    {20, 15, 18},
    {1.5, 16.4, 3.0},
    {34, 26 , 45},
    {280, 300, 800}
};

//Variable de proceso
int j; //contadores en ciclos
int idcate,op1; //id de categoria y de num producto

//Genera el menu de Categorías de productos
cout<<"\t\tEmulacion de base de datos\n\n";
cout<<"\nLista de categorias de Productos\n";
for(j=0;j<CAT;j++)
    cout<<(j+1)<<"- "<<categorias[j]<<"\n";
cout<<"\nDigite numero del tipo de producto que comprara: ";
cin>>idcate;

switch(idcate){
    case 1: //Menu de precios de Lamparas
        cout<<"\n\nProductos de categoria: "
            <<categorias[idcate-1]<<"\n";

        for(j=0;j<PRO;j++)|
            //cout<<(idcate-1)*PRO+j<<endl;

```



```

        cout<<(j+1)<<" . "<<nombres[(idcate-1)*PRO+j]
            <<"\t $ " <<precios[idcate-1][j]<<endl;
    cout<<"\n";
    break;
    case 2: //Menu de precios de Area Escolar
    cout<<"\nProductos de categoria: "
        << categorias[idcate-1]<<"\n";
    for(j=0;j<PRO;j++)
        cout<< (j+1) <<" . "<<nombres[(idcate-1)*PRO+j]<<
            "\t $ " <<precios[idcate-1][j]<<endl;
    cout<<"\n";
    break;
    case 3: //Menu de precios de Herramientas
    cout<<"\nProductos de categoria: "<<
        categorias[idcate-1]<<"\n";
    for(j=0;j<PRO;j++)
        cout<<(j+1)<<"-Precio de " <<nombres[(idcate-1)*PRO+j]<<
            "\t $ " <<precios[idcate-1][j]<<endl;
    break;
    case 4: //Menu de precios de PC's
    cout<<"\nProductos de categoria: "<<
        categorias[idcate-1]<<"\n";
    for(j=0;j<PRO;j++)
    cout << (j+1) <<" . "<<categorias[idcate-1]<<" marca " <<
nombres[(idcate-1)*PRO+j]<<"\t $ " <<precios[idcate-1][j]<<"\n\n";
    break;
    default :
        idcate=0;
} //fin switch idcate
if(idcate!=0){
    cout<<"Su opcion es?\t";
    cin>>op1;
    cout<<"\nPrecio de " <<nombres[(idcate-1)*PRO+(op1-1)]<<
        " es de $" <<precios[idcate-1][op1-1]<<"\n\n";
}else
    cout<<"Error!!! opcion invalida\n\n";
    cout<<"Fin Programa...\n\n";

```

```
    getch( );  
} //fin main
```

9. Guarde el código anterior, proceda a compilarlo y corregir errores sintácticos (si los tiene).
10. Ejecute el programa y haga la selección de una categoría y de un producto de la misma para ver así su precio.
11. Sin cerrar la ventana de resultados, compare el resultado con la declaración de las matrices definidas en el código, para identificar el uso de las filas y columnas que se da a cada una.

V. ANÁLISIS DE RESULTADOS

Elabore el código fuente de C++ que solucionen a cada uno de los problemas a continuación:

PROBLEMA 1:

Haga una copia del archivo **PAL_Guia9_E2.cpp** del procedimiento, para luego realizar los siguientes cambios en este archivo.

- Borre todo el código, excepto la declaración de las matrices y variables
- Elimine la inicialización de valores que se hace a las 3 matrices.
- Ahora redacte un nuevo código que muestre el siguiente menú de opciones al usuario:
 - a) Agregar categoría
 - b) Agregar producto a una categoría
 - c) Ver lista de productos
 - d) Salir
- Tome en cuenta lo siguiente:
 - * Pueden agregarse hasta 20 categorías diferentes
 - * Cada categoría puede tener hasta 10 productos diferentes como máximo
 - * Cuando usuario elija una opción de menú, este se borra para mostrar el dialogo correspondiente
- Al finalizar la acción del menú elegida, debe mostrarse nuevamente el menú

PROBLEMA 2:

Elabore un programa que permita llevar un control de planilla de los sueldos de N empleados de la empresa Calipso SA. Se debe mostrar el siguiente menú de manera continua:

- a) Ingrese los datos (Nombre y sueldo base) de un nuevo empleado
- b) Ver informe de un empleado: muestra nombre, sueldo base y sueldo neto.
- c) Ver la planilla de empleados:

Se debe presentar como una tabla conteniendo los siguientes datos:

- ✓ Por c/empleado: #correlativo de empleado, nombre, monto de sueldo base, suma de los descuentos de ley (renta [10%] e ISSS [3.5%]) y sueldo final.
 - ✓ Monto de planilla completa a pagar.
- d) Finalizar aplicación

PROBLEMA 3:

Desarrolle una aplicación que permita registrar la cantidad de votos de una elección presidencial ficticia desarrollada en El Salvador, en la que participan 4 partidos políticos.

El conteo de votos debe limitarse a solamente los 9 municipios que conforman el departamento de Cabañas. Finalmente, muestre a la autoridad electoral los siguientes resultados específicos:

- a) Cantidad acumulada de votos que obtuvo cada partido entre los siguientes municipios: Sensuntepeque (su cabecera municipal), Cinquera y San Isidro.
- b) Cantidad y porcentaje de votos que obtuvo cada partido político en todo el departamento de Cabañas.

PROBLEMA 4:

Cree un programa que sea capaz de calcular las ventas semanales de un mes de un negocio de comida rápida, además deberá desplegar en pantalla la venta total de todo el mes.

Salida del programa:

Ventas de cuatro semanas				
	SEM1	SEM2	SEM3	SEM4
(L)	123.50	234.60	345.45	321.40
(M)	345.00	456.65	123.50	234.60
(M)	345.45	321.40	345.00	456.65
(J)	123.50	234.60	345.45	321.40
(V)	345.00	456.65	123.50	234.60
(S)	345.45	321.40	345.00	456.65
(D)	0.00	0.00	0.00	0.00
	1627.90	2025.30	1627.90	2025.30
Venta total del mes:			7306.40	

Guía de Laboratorio No. 9

RÚBRICA DE EVALUACIÓN

Actividad a evaluar: ANÁLISIS DE RESULTADOS

Formar grupos de **entre 3 a 4 estudiantes**, llenar esta hoja de evaluación y entregarla a su docente.
Instructor seleccionará 2 problemas del análisis de resultados para ser resueltos por el grupo de estudiantes.

Lista de Integrantes:

CARNET 1	CARNET 2	CARNET 3	CARNET 4	CARNET 5

Problemas a resolver:

Criterio a evaluar	¿Prob 1?	¿Prob 2?	PROM.
(30%) Se define un menú, el cual muestra las acciones que usuario puede hacer con el programa Menú se muestra continuamente, hasta que usuario selecciona finalizar aplicación Cada opción de menú funciona apropiadamente Código fuente se logra compilar y se obtiene a c/u de los resultados solicitados			
(45%) Define la cantidad y tipos de datos de los arreglos requeridos en el problema. Acceso correcto a las posiciones de cada una de las matrices utilizadas en el programa Cumple las restricciones dadas por el problema			
(25%) Aplica un lenguaje conciso para comunicarse con usuario Documenta el código fuente de manera apropiada			
			Nota: