	UNIVERSIDAD DON BOSCO **FACULTAD DE INGENIERIA** **ESCUELA DE COMPUTACIÓN**
Ciclo II	**Programación de Algoritmos** **Guía de Laboratorio No. 8** **Arreglos unidimensionales (vectores)**

I. RESULTADOS DE APRENDIZAJE

Que el estudiante:

- Defina el concepto de arreglo y cómo utilizarlo en el lenguaje C++
- Determine su metodología para acceder a los elementos de un arreglo
- Resuelva una de las aplicaciones prácticas típicas con arreglos: *ordenamiento de elementos*.

II. INTRODUCCIÓN

¿Qué es un Arreglo?

Muchas aplicaciones requieren el procesamiento de múltiples datos que tienen características comunes, por ejemplo, un conjunto de datos numéricos, representados por *x1, x2, x3 x4… hasta un xN*.

En tales situaciones, es conveniente colocar los datos en un ***arreglo***, el cual se caracteriza porque todos sus elementos comparten un **mismo nombre** (en este ejemplo el nombre para la variable de arreglo seria **x**).

Los datos individuales pueden ser caracteres, números enteros, números de coma flotante de simple o de doble precisión (reales).

Debe tenerse muy en cuenta que todos los N elementos de un arreglo se caracterizan en que **cada uno debe tener el mismo nombre y también ser de un mismo tipo de dato**.

Definición de arreglo (array en inglés)

Un arreglo es un conjunto de variables del mismo tipo de datos que pueden ser referenciadas a través de un mismo nombre. La forma de identificar a un elemento determinado es a través de un índice.

Como se maneja a los elementos de un arreglo

Cada elemento (dato individual) del *arreglo* es referenciado mediante la especificación del **nombre del arreglo** seguido por **uno o más índices**. El número de índices utilizados determinan las **dimensiones del Arreglo**.

Dimensiones del arreglo

Al definir un arreglo, se le debe indicar el total de elementos N que lo formaran. A este total N se le conoce como **Dimensión del arreglo**, el cual permite indicar el total de posiciones que se almacenara dentro de la memoria de la PC.

Si un arreglo tiene solo una dimensión, se le llama **arreglo unidimensional** o también llamado **vector**, porque solo basta indicar un número, llamado "**índice**" para referenciar a una posición específica dentro del mismo. A

Si a un arreglo se le definen 2 o más dimensiones, se le conoce como "**arreglo multidimensional**" o también **matriz**, y necesitara tantos índices como dimensiones se le declaren al arreglo.

Si define 2 dimensiones para un arreglo, a este se le conoce como un **arreglo bidimensional**.

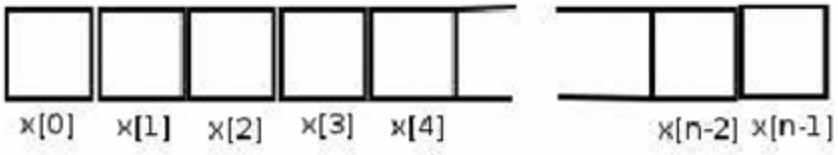
Al definir más de una dimensión al arreglo, le permite aplicar diversos métodos de clasificación para el almacenamiento de listas de datos más complejos.

Índices del arreglo

Cada índice se encierra entre corchetes [ ] y **debe ser expresado como un entero no negativo**. Así en un arreglo unidimensional llamado **x** de **"n"** elementos, para referenciar a cada uno de sus elementos se hará así:

x[0], x[1], x[2], x[3], ... Hasta x[n-1]

Tal como se ilustra en la siguiente figura:



Al observar el conteo de índices de la figura anterior, se debe tener muy claro que: **al declarar un arreglo dentro de Lenguaje C/C++, con una dimensión de N elementos, el primer índice disponible del arreglo comienza con 0 (cero) y el último índice es N-1 (uno menos del total N indicado para la dimensión)**.

Esta aclaración es válida para el conteo de índices de cada una de las dimensiones de un arreglo multidimensional. Observe los ejemplos siguientes:

Ejemplo 1:

- Para declarar un arreglo unidimensional llamado `MiLista` con un total (una dimensión única) de 5 elementos de tipo entero, utilizara esta instrucción:

```
int MiLista [5];
```

- Si desea **asignar el valor 956 a la cuarta posición de MiLista**, lo hará referenciando a este elemento con la siguiente instrucción:

```
MiLista[3]= 956; //índice 3 del arreglo
```

- Al final de esta asignación, el arreglo MiLista se verá en memoria así:

índice	0	1	2	**3**	4
Valor asignado				**956**	

Declaración de un arreglo

Los arreglos se declaran de igual forma que las variables ordinarias, excepto que cada nombre de arreglo debe acompañarse de una especificación de tamaño (es decir, el total de elementos por cada dimensión).

El tamaño de cada dimensión se especifica con una expresión entera positiva encerrada entre corchetes [ ].

En términos generales, la definición de un arreglo puede expresarse como:

`TipoDato` *`nombre-arreglo`* `[expresión] [expresion2]... [expresionD]`

En donde:

- **`TipoDato`** es el tipo de datos común para todos los elementos del arreglo
- *`nombre-arreglo`* es el nombre del arreglo
- `[expresión][expresión2][expresión3]`**... hasta** `[expresionD]` es una o más expresiones enteras positivas que indican el total de elementos por cada dimensión del arreglo.

El primer valor **expresión** es obligatorio para definir un arreglo unidimensional. Si se utiliza además la expresion2, se define un arreglo bidimensional (2 dimensiones). Si utiliza el resto de expresiones, crea un arreglo multidimensional o matriz.

Ejemplos sobre definición de arreglos y su respectivo significado respectivo:

`int A[100];`	Arreglo unidimensional A de 100 enteros. Para utilizarlos se referencian como A[0], A[1], y así sucesivamente hasta las posiciones A[98] y A[99]
`float raices[5];`	arreglo llamado raíces que tiene 5 valores de comas flotantes (valores reales)
`char texto[80];`	arreglo de 80 caracteres llamado texto */

Asignando valores a las posiciones del arreglo

Un arreglo puede almacenar un valor por cada una de sus posiciones.

Existen 3 métodos para almacenar los valores en las posiciones del arreglo:

a) Inicializar todas las posiciones de un arreglo al momento de declararlo, con los valores definidos por el programador.

b) Asignar un valor fijo, o de una variable o el resultado de un cálculo que genere un valor del mismo tipo de dato asignado al arreglo

c) Solicite al usuario el valor para una posición del arreglo

III. MATERIALES Y EQUIPO

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

IV. PROCEDIMIENTO

Parte 1: Métodos para acceder a un arreglo

- **Método 1**: solicitando al usuario valor para una posición
- **Método 2**: inicializando posiciones al momento de declarar la matriz
- **Método 3**: asignando valor desde: a) de una variable o sino b) un proceso/cálculo

1. Cree una carpeta denominada **PALGuia8 _CARNET**, en el cual se guardará los códigos fuente (cpp) de los ejemplos del procedimiento a continuación.

Ejercicio 1: **PAL_Guia8_E1**

Método 1: solicitando al usuario los datos del arreglo

```
#include <iostream>
using namespace std;
#include <stdlib.h>
#include <conio.h>
#define tam 20 //define el total maximo de elementos

main(){ //Llenado y Salida de arreglo Unidimensional
  int n; //total elementos a ingresar
```

```
  int arreglo[tam]; //arreglo de 20 elementos
  int j=0; //contador de ciclo e indice del arreglo

  cout<<"  INICIO: Datos del arreglo."<<endl;
  cout<<endl<<"Valores actuales del arreglo:"<<endl;

  for(j=0;j<tam; j++) cout<<"Elemento["<<j<<"]=\t "<<arreglo[j]<<endl;
  system("pause");

  cout<<endl<<"LLENADO DEL ARREGLO"<<endl;
  cout<<"Cuantos elementos tendra arreglo(como Max 20):\t";
  cin>>n;

  if((n>0)&&(n<=tam)){
    cout<<"\n\nIngreso de sus "<<n<<" datos hacia el arreglo"<<endl;
    for(j=0; j<n; j++){
      cout<<endl<<"Ingrese el elemento ["<<j<<"]:\t";
      cin>>arreglo[j];
    }//fin for j

    system("pause");
    system("cls");

    cout<<"\tVista de datos almacenados en arreglo"<<endl;
    for(j=0;j<tam; j++)
       cout<<"Elemento ["<<j+1<<"] = "<<arreglo[j]<<endl;
  }else
    cout<<"*** ERROR: Entrada no Valida ***";

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

Ejercicio 2: **PAL_Guia8_E2**

+ **Método 2:** inicializando posiciones al momento de declarar la matriz

+ **Método 3**: asignando valor desde: a) de una variable o sino b) un proceso/cálculo

```
#include <iostream>
using namespace std;
#include <conio.h>
main(){
    // inicializa posiciones al momento de declarar la matriz
    float venta[]={62, 0, 70, 63.5, 0, 83.4};
    int i; //contador para indice del arreglo
    float prom=0; //valor ventas promedios del semestre
    float mayo=76.9; //guarda venta de mes de mayo

    venta[1]=56.4; //asigna valor a venta de febrero
    venta[4]=mayo; //asigna valor desde la variable mayo


    //calcula venta promedio de 1er trimestre
    prom=(venta[0]+venta[1]+venta[2])/3;

    //mostrando valores almacenados en el vector venta
    cout<<endl<<"Ventas mensuales del semestre:"<<endl;
    for(i=0;i<6;i++)
       cout<<endl<<"mes"<<i+1<<" :$"<<venta[i];
    getch();
    //resultado de operar valores de un arreglo
    cout<<endl<<endl<<"Venta promedio de 1er trimestre es de $";
    cout<<prom;

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

Parte 2: Operaciones de ordenamiento con los elementos de un arreglo

Ejercicio 3: **PAL_Guia8_E3**

Ordenar en forma ascendente los datos de un vector, utilizando metodo de burbuja

```
#include <iostream>
using namespace std;
#include <conio.h>
#include <stdlib.h>
```

```
main(){  //Ordenamiento rapido de un arreglo
 int n; //total elementos de arreglo

 int i,j, k; //contadores de indices de arreglo
 int auxiliar; //variable para intercambio de 2 valores enteros

 cout<<"PROGRAMA PARA ORDENAR ELEMENTOS DE ARREGLO\t";
 cout<<"//ORDEN ASCENDENTE\\"<<endl;

//Solicita total de elementos del arreglo

//Ciclo infinito al colocar valor logico verdadero (1) como condicion
 while(1){
  cout<<"\nDe cuantos elementos quiere su arreglo[1-20max]\t"; cin>>n;
  if(!(n>0 && n<=20)){
cout<<"\t***> ERROR, total de elementos debe estar entre 1 hasta 20
<***\n";
    getch();
  }else
    break; //sale del ciclo infinito while(1)
 }// fin while(1)

 system("cls");
 int arreglo[n]; //crear un arreglo de n elementos enteros
 cout<<"Ingreso los "<<n<<" datos del arreglo:"<<endl;
//recorre arreglo para solicitar c/posicion a usuario
 for(i=0; i<n; i++){
   cout<<endl<<"Ingrese el elemento ("<<i+1<<"): ";
  cin>>arreglo[i];
 }//fin for i

 /*
 Ordena arreglo de forma ascendente, camparando posicion [i] con
 el resto de posiciones del arreglo
 */
 for(i=0;i<n-1;i++){
  for(j=i+1;j<n;j++){
```

```
   //intercambia valor de posicion [i] con posicion [j]
   if(arreglo[i]>arreglo[j]){
    auxiliar=arreglo[i];
    arreglo[i]=arreglo[j];
    arreglo[j]=auxiliar;
   }//fin if
  }//fin for j
 }//fin for i

 cout<<"\n\narreglo ya ordenado de forma ASCENDENTE"<<endl;
 j=0; //valor indice inicial [0]
 do{ //inicio ciclo do-while
  cout<<"\nelemento ("<<j+1<<")= "<<arreglo[j]<<endl;
  j++; //incrementa variable conteo del ciclo do-while
 }while(j<n); //fin ciclo do-while

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

Ejercicio 4: **PAL_Guia8_E4**

Ordena en forma ascendente los datos de un vector cuyos valores son dados por el programador. Se utiliza una variante del metodo de burbuja, basado en la cantidad de comparaciones a ejecutar

```
#include<iostream>
using namespace std;
#include<conio.h>

main(){
   int T=10; //total elementos de arreglo
   int n[]={10, 4, 5, 1, 3, 8, 2, 7, 9, 6};

   int i, j; //contadores de indices de arreglo
   int k;//total de comparaciones que se haran
   int aux; //variable para intercambio de 2 valores enteros

   i=0; //indice fijo para iniciar comparacion
```

```
//total comparaciones a realizar entre posiciones
   for(k=T-1;k>-1;k--){
     for(j=i+1;j<T;j++){//controla 2do indice para comparar
       if(n[i]>n[j]){ //determina si hace cambio de posiciones
         //hace intercambio de posiciones
         aux =n[i];
         n[i]=n[j];
         n[j]=aux;
       }
     }//fin for j
     i++; //cambia posicion de inicio de comparacion
   }//for k

   //muestra listado de posiciones de vector ya ordenados
   cout<<"Lista ordenada de valores es: \n";
   for(i=0;i<T;i++) cout<<n[i]<<"\t";

   getch();
}
```

Ejercicio 5: **PAL_Guia8_E5**

Ordena en forma ascendente los datos de un vector cuyos valores son dados por el programador.

Similar al ejemplo anterior, pero este compara parejas adyacentes, buscando que el dato mayor se ubique al final, el siguiente menor en la penúltima posición y así sucesivamente hasta ordenar el vector.

```
#include<iostream>
using namespace std;
#include<conio.h>
main(){
     int T=10;//total elementos de arreglo
     int n[]={10,4,5,1,3,8,2,7,9,6};

     int i; //contador de indices de arreglo
     int k;//total de comparaciones que se haran
     int aux; //variable para intercambio de 2 valores enteros

for(k=T-1;k>=1;k--){//total de comparaciones
```

```
  for(i=0;i<k;i++){//controla 2do indice para comparar
      //compara parejas consecutivas de posiciones
      if(n[i]>n[i+1]){
      //hace intercambio de posiciones
      aux=n[i];
      n[i]=n[i+1];
      n[i+1]=aux;
      }
  }//fin for i
}//for k

   //muestra listado de posiciones de vector ya ordenados
   cout<<"Lista ordenada de valores es: \n";
   for(i=0;i<T;i++) cout<<n[i]<<"\t";
   getch();
}
```

Ejercicio 6: **PAL_Guia8_E6**

Manejo de un arreglo de tipo carácter (char)

```
#include<iostream>
using namespace std;
#include<conio.h>

main(){
char nombre[25];
int edad;
cout<<"Digite nombre completo de alumno:";
//captura cadenas con espacios
cin.getline(nombre,25);

//captura cadenas sin espacios
//cin>>nombre;
```

```
cout<<"Digite la edad del alumno: ";
cin>>edad;

if (edad>=18)
    cout<<"\n\tEl alumno "<<nombre<<" es mayor de edad";
else
    cout<<"\n\tEl alumno "<<nombre<<" es menor de edad";

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

V. ANÁLISIS DE RESULTADOS

Elaborar el código fuente C++ que solucione a cada uno de los siguientes problemas:

Problema 1

Solicite al usuario un listado de 12 números enteros de todo tipo (positivos, negativos, ceros), para luego ver en pantalla los siguientes resultados:

a) Porcentaje de valores pares ingresados.

b) Listado de solamente los valores positivos, presentados de manera descendente

c) Amplitud/Rango de la serie de valores negativos ingresados.

(*) En caso que literales a) y b) no tengan respuesta, debe ser informado apropiadamente a usuario.

Problema 2

El contador del almacén "El Buen Vestir" cuenta con el monto de las ventas mensuales de todo el año 2016. Con esta información, solicita una aplicación informática que le determine cada uno de los siguientes resultados:

- Monto ($) de la venta anual promedio.
- Porcentaje de meses del año en los cuales la venta fue *Baja* (menos del 50% del promedio anual)
- Listado de los nombres de los meses (enero, febrero, etc.) en los cuales la venta fue *Excelente* (monto de venta del mas del 75% del promedio anual de ventas).

Problema 3

Escriba un programa que declare a 2 arreglos con 15 enteros cada uno.

Luego, en el primer vector, asigne a sus posiciones todo tipo de valores enteros aleatorios, definidos estáticamente por usted mismo, que incluyan valores repetidos al azar.

Finalmente, el programa llenará las posiciones del segundo vector con la copia de los valores del 1er vector, pero ordenados de manera ascendente y sin repetir ninguno de sus valores.

Para demostrar su funcionamiento, deberá imprimirse en pantalla a los valores finales de ambos vectores.

Observe un ejemplo:

Vector 1:	14	8	2	3	14	8	14	2	14	3	10	5	10	8	2
Vector 2:	2	3	5	8	10	14									

Problema 4

Cree una aplicación que solicite a un estadista un listado de N números enteros exclusivamente positivos.

Tomar en cuenta que el usuario puede equivocarse al ingresar algún dato (ingresar ceros o negativos). Cuando esto ocurra, se debe solicitar el valor nuevamente hasta que este sea correcto, y luego, continuar con el ingreso del resto de datos.

Luego del ingreso de los valores, debe mostrar en pantalla a los siguientes resultados:

a) Promedio general de la serie ingresada ($\bar{X}$) b) Desviación típica

La desviación típica (σ) de una serie de N números positivos se calcula con la siguiente fórmula:

$$\sigma = \sqrt{\frac{\sum_{i=1}^{N}(X_i - \bar{X})^2}{N}} = \sqrt{\frac{(X_1 - \bar{X})^2 + (X_2 - \bar{X})^2 + \cdots + (X_N - \bar{X})^2}{N}}$$

(*) Cualquier tipo de ciclo utilizado en el cod. fuente, puede ser usado solo una vez.

Problema 5

Redacte un programa que llene a una matriz con un listado de 20 números enteros generados aleatoriamente por la misma aplicación (es decir, no serán definidos estáticamente por el programador ni ingresados por el usuario).

Cada número generado debe variar solamente entre -100 hasta 100, excepto el cero.

Luego se mostrarán los siguientes resultados:

a) Lista completa de los números aleatorios almacenados en la matriz, pero presentados de manera descendente.

b) Porcentaje de números impares, que se generaron en el listado aleatorio. En caso que no se genere ningún número impar, debe informarse al usuario apropiadamente.

Problema 6

Ayude a un estudiante de Estadística 1 a generar un listado de N valores aleatorios, c/u de los cuales estará ubicado en un rango de solamente 1 hasta 15.

Finalmente, debe mostrársele el listado de números diferentes almacenados en el vector (que fueron generados por su sistema), así como la frecuencia de aparición de cada valor diferente en la misma.

Observe un ejemplo:

Si el estudiante desea generar un listado de 13 números aleatorios, la aplicación le muestra en pantalla al siguiente listado de 13 valores aleatorios (que fueron generados bajo el rango dado al inicio del problema):

3 6 3 6 5 1 6 6 3 12 5 1 5

Finalmente, le mostrara el total de veces (frecuencia) que cada valor se generó en el listado.

Para este ejemplo, la tabla de frecuencias sera:

Numero	Frecuencia
1	2
3	3
5	3
6	4
12	1

Problema 7

El almacén *La Milagrosa* requiere un programa que evalue sus ventas de todo el año pasado.

El negocio indica que su contador general conoce los montos de ventas mensuales correspondientes. Finalmente, esta persona desea que la aplicación le genere los siguientes resultados:

- Valor del promedio de venta anual.
- Listar el/los nombres de los meses en los cuales se alcanzaron los siguientes niveles de ventas:

(*) Nivel ventas	**Descripción**
Mínimas	Menos del 40% del promedio anual de ventas
Regular	Entre el 40% a 75% del promedio de ventas
Excelente	Más del 75% del promedio mensual

(*) Importante: en caso que un nivel de ventas no se haya alcanzado en todo el año, debe ser informado al contador.

Guía de Laboratorio No. 8
RÚBRICA DE EVALUACIÓN

Actividad a evaluar: ANÁLISIS DE RESULTADOS

Formar grupos de entre 3 a 5 estudiantes, llenar esta hoja de evaluación y entregarla a su docente.

Su instructor seleccionará 3 problemas del análisis de resultados, para ser resueltos apropiadamente por el grupo.

Lista de Integrantes:

CARNET 1	CARNET 2	CARNET 3	CARNET 4	CARNET 5

Problemas a resolver:

Criterio a evaluar	¿Prob 1?	¿Prob 2?	¿Prob 3?	PROMEDIO
(25%) Define la cantidad y tipos de datos apropiados para los arreglos requeridos en el problema.				
(35%) • Cumple las restricciones dadas en la redacción • La solución se redacta SOLAMENTE en la función main ( ), porque aun no se han visto la creación de funciones propias. • Lectura/Impresión de datos se hace con C++ (no usa C nativo)				
(25%) Código fuente se logra compilar y se obtiene a c/u de los resultados solicitados				
(15%) Documenta apropiadamente a cada ejercicio solicitado				
			Nota:	