Trabajo Práctico 3

Objetivos:

- **Comprender el concepto de dato estructurado estático**
- **Trabajar con ciclos repetitivos**
- **Trabajar con arreglos unidimensionales y bidimensionales**
- **Emplear datos estructurados con diferentes tipos de datos**
- **Comprender los conceptos de búsqueda y ordenamiento**

Problemas propuestos: Arreglos Unidimensionales de variables simples

Resolver los siguientes ejercicios sin utilizar funciones para la carga y presentación por pantalla de los arreglos correspondientes.

1. Realizar un programa que permita cargar un arreglo cuyos elementos sean N números enteros y luego permita presentarlo por pantalla.
2. Realizar un programa que permita cargar un arreglo cuyos elementos sean N caracteres y luego permita presentarlo por pantalla.
3. Realizar un programa que permita cargar un arreglo cuyos elementos sean N números de punto flotante y luego permita presentarlo por pantalla.
4. Cargar caracteres en un arreglo de orden N. Luego contabilizar cuántos caracteres del arreglo son alfabéticos, cuántos son dígitos y cuántos de otras categorías.
5. Dado un arreglo de orden N, realizar un programa que permita ingresar números enteros a dicho arreglo. Controlar que la cantidad de elementos ingresados no supere el orden permitido del arreglo. Luego mostrar por pantalla el arreglo ingresado, el valor del promedio de los elementos ingresados, la cantidad de elementos ingresados que son mayores al promedio y la cantidad de elementos ingresados que sean menores al promedio.
6. Dado un arreglo de orden N, realizar un programa que permita ingresar números enteros a dicho arreglo. Controlar que la cantidad de elementos ingresados no supere el orden permitido del arreglo. Luego contar los elementos del arreglo que sean mayores a un número entero ingresado por teclado. Finalmente, mostrar por pantalla el arreglo ingresado, el número ingresado y la cantidad de elementos del arreglo que son mayores al número ingresado.

Resolver los siguientes ejercicios utilizando funciones, en el caso que sea necesario, para:

➔ Controlar que el orden del arreglo a cargar sea un entero positivo y que no supere el valor máximo que puede contener dicho arreglo.

➔ Cargar el arreglo.

➔ Mostrar el arreglo.

➔ Resolver lo solicitado en cada problema.

7. Realizar un programa que permita cargar un arreglo que contenga una cantidad N de caracteres alfabéticos y luego lo presente por pantalla. Recordar que debe resolverse empleando una función para cargar el arreglo y controlar los caracteres que se ingresan y otra función para presentar por pantalla el arreglo resultante.
8. Realizar un programa que permita determinar cuántas veces se encuentra el valor x en un arreglo. Para esto se deberá:
    - ingresar números enteros en un arreglo de orden N
    - ingresar un valor x
    - mostrar por pantalla el arreglo ingresado
    - mostrar el elemento buscado x y la cantidad de veces que se encontró el elemento buscado. En caso de que el vector no contenga un elemento de valor igual a x, mostrar el mensaje correspondiente.
9. Dado un arreglo V de orden N que contiene enteros positivos, generar un arreglo P de orden M que contenga los elementos de V que sean números primos. Si el arreglo V no tiene números primos entonces presentar un mensaje por pantalla.

    Utilizar funciones para
    - Controlar que el orden del arreglo que se va a cargar sea un entero positivo y que no supere el valor máximo que puede contener el arreglo.
    - Controlar que los valores ingresados sean positivos.
    - Determinar si un número es primo o no.
    - Armar el nuevo vector.
10. Realizar un programa que permita ingresar números enteros en un arreglo de orden N, luego **desplazar sus elementos X lugares a la derecha** (tener en cuenta que al mover los elementos unba posición hacia la derecha, el último elemento del arreglo pasará a ser el primer elemento del arreglo). Mostrar por pantalla el arreglo ingresado y el arreglo modificado. Por ejemplo, si se ingresa el siguiente arreglo: (3, 0, 8, 2, 1, 7) y X=3, entonces el arreglo modificado debe quedar del siguiente modo: (2, 1, 7, 3, 0, 8).
11. Dado un arreglo de N elementos enteros, **se debe guardar sus elementos en otro arreglo** de orden N pero en sentido invertido. Ej: Arreglo dado: **V**=(2,4,7,9,5,6); Arreglo invertido: **W**=(6,5,9,7,4,2).
12. Dado un arreglo de N elementos enteros, **se debe invertir los elementos del mismo** y presentar por pantalla el arreglo resultante. **No debe usar un arreglo auxiliar.** Ej: El Arreglo dado: **V**=(2,4,7,9,5,6) se convierte en: **V**=(6,5,9,7,4,2).

Ordenamiento y búsqueda en arreglos unidmensionales

13. Realizar un programa que permita cargar un arreglo V de orden N de números enteros y luego presente por pantalla el arreglo ingresado y genere otro arreglo P que contenga todos los números pares de V, y lo presente por pantalla. En caso de que V no tenga números pares mostrar el correspondiente mensaje por pantalla.
14. Realizar un programa que permita cargar un arreglo V de orden N de números enteros y luego presente por pantalla el arreglo ingresado y genere otro arreglo P que contenga todos los números pares de V, y lo presente por pantalla ordenado de forma ascendente. En caso de que V no tenga números pares mostrar el correspondiente mensaje por pantalla.
15. Realizar un programa que permita cargar un arreglo de orden N de números enteros. Luego ordenar sus elementos en forma descendente y presentarlos por pantalla. **No utilizar un arreglo auxiliar para el ordenamiento.**

16. Ingresar en un arreglo de caracteres el nombre de una persona, ingresar cada letra en un elemento del arreglo. Luego, ingresar un carácter por teclado y comprobar si dicho carácter pertenece al nombre ingresado. Finalmente, mostrar el arreglo y el mensaje correspondiente.
17. Cargar un arreglo con N números enteros, luego buscar el menor elemento impar contenido en el mismo. En caso de no encontrar elementos impares presentar un mensaje por pantalla.
18. Diseñar un programa que emplee un arreglo de números enteros de orden N y que cuente con un menú interactivo con las siguientes opciones:
    - a.Cargar arreglo de enteros
    - b.Mostrar arreglo de enteros
    - c.Mostrar el valor del mayor elemento del arreglo
    - d.Mostrar el valor del menor elemento del arreglo.

Tener en cuenta que el usuario puede presionar la opción b, c o d sin haber cargado previamente el arreglo (opción a), por lo que deberá controlar esta situación. Además, controlar que el valor de N, ingresado por el usuario, sea un entero positivo y se encuentre en el rango permitido.

Resolver los siguientes ejercicios utilizando las funciones que crea necesarias de la librería ctype.h:https://www.tutorialspoint.com/c_standard_library/ctype_h.htm

19. Ingresar un arreglo de caracteres de orden N, luego convertir los caracteres alfabéticos en mayúscula a minúscula y viceversa. **Presentar el arreglo original y el modificado** por pantalla. **No utilizar un arreglo auxiliar**
20. Cargar caracteres en un arreglo de orden N. Luego contabilizar, usando funciones, cuántos caracteres del arreglo son alfabéticos, cuántos son dígitos y cuántos son de otras categorías.
21. ***Para pensar:*** Escribir un programa que, dada una frase de tamaño máximo N, determine si es un palíndromo o no. Un palíndromo es una frase que, atendiendo sólo a sus letras e ignorando los espacios, acentos, signos de puntuación y tipo de letra (mayúscula o minúscula) expresa lo mismo leída de izquierda a derecha que de derecha a izquierda. “dabale arroz a la zorra el abad”.

    Utilizar funciones para:

    - Controlar que el orden del arreglo que se va a cargar sea un entero positivo y que no supere el valor máximo que puede contener el arreglo.
    - Ingresar la frase caracter a caracter en un arreglo.
    - Evaluar si la frase es un palíndromo.

22. ***Para Pensar:*** En una pila de platos, los platos van poniéndose uno sobre el otro, y si se quiere sacar uno, se saca primero el último que se ha puesto. Realizar un programa que permita simular una pila de números usando un arreglo unidimensional y un menú interactivo que muestre en pantalla las siguientes operaciones:
    - Apilar (push): Añadir un elemento en la parte superior de la pila, es decir que debe ingresar un número por teclado y luego agregarlo en el vector a continuación del último número agregado (se debe controlar previamente que exista lugar en el arreglo, en caso contrario deberá mostrar el mensaje correspondiente).
    - Desapilar (pop): Leer y retirar el elemento superior de la pila, es decir que debe mostrar en pantalla el último número que fué ingresado en el vector y luego deberá eliminarlo, dejando disponible esa posición para almacenar otro número (se debe controlar previamente que la pila no se encuentre vacía, en caso contrario deberá mostrar el mensaje correspondiente).
    - Leer último (top o peek): Leer el elemento superior de la pila sin retirarlo, es decir que debe mostrar en pantalla el último número que fué ingresado en el vector (se debe controlar previamente que la pila no se encuentre vacía, en caso contrario deberá mostrar el mensaje correspondiente).
    - Mostrar el arreglo en pantalla.
    - Salir.

Problemas propuestos: Arreglos Unidimensionales de estructuras

23. Realizar un programa que permita ingresar los datos de las fechas de nacimiento de una cantidad N personas en un arreglo de estructuras. Luego presentar el arreglo por pantalla. Resolver el ejercicio sin utilizar funciones para la carga y presentación por pantalla del arreglo correspondiente. Utilizar funciones únicamente para la carga y presentación de cada estructura. Los campos de la estructura deben almacenar los siguientes datos: día, mes, año y nombre (inicial):

```
typedef struct{
        int dia, mes, anio;
        char nombre;
}datos;
```

Resolver los siguientes ejercicios utilizando funciones para la carga y presentación por pantalla de los arreglos y para la realización de controles de datos y toda otra operación que se indique.

24. Resolver el problema anterior pero ahora empleando funciones para:
    - solicitar los datos de cada persona
    - cargar el arreglo llamando a la función del punto anterior
    - mostrar los datos de cada persona
    - mostrar el arreglo correspondiente llamando a la función del punto anterior

25. Ingresar los datos de una cierta cantidad de monitores en un arreglo de estructuras. Luego, presentar por pantalla los datos de los monitores en donde el precio sea menor o igual a un precio de búsqueda ingresado por teclado. Los campos de la estructura deben almacenar los siguientes datos: inicial de la marca, precio y cantidad en stock:

```
typedef struct{
        char marca;
        float precio;
        int stock;
}monitores;
```

Utilizar funciones para:

- Controlar que el orden del arreglo sea un entero positivo y menor que el valor máximo de elementos que pueden contener el arreglo.
- Solicitar al usuario los datos de cada uno de los monitores.
- Cargar el arreglo que contiene los monitores llamando a la función del ítem anterior.
- Presentar por pantalla los datos de cada monitor.
- Buscar y presentar por pantalla los datos de los monitores que cumplan que su precio sea menor o igual a un precio de búsqueda ingresado por teclado.

26. Los datos de un televisor se almacenan usando la siguiente estructura:

```
typedef struct{
        char marca;
        float precio;
        int cantidad;
}televisores;
```

Realizar un programa que permita ingresar los datos de N televisores en un arreglo. Presentar por pantalla el arreglo y luego permitir la modificación del precio en los casos en donde exista coincidencia entre una inicial de una marca ingresada por teclado y la inicial registrada para el televisor. Al realizar la búsqueda:

Si no existe en el arreglo un elemento con esa inicial presentar por pantalla el mensaje correspondiente.

Si se encuentra coincidencia (una o más), se solicitará el ingreso del nuevo precio. Se debe presentar por pantalla el contenido de dicho arreglo antes y después de la modificación.

Utilizar funciones para:

- Almacenar los datos de los televisores en el arreglo. Se sugiere además el uso de una función para solicitar los datos de cada televisor.
- Realizar la búsqueda de las coincidencias y si es que hubiera, realizar el reemplazo del valor del precio.
- Mostrar el arreglo de televisores.

27. Se desea realizar un programa que permita ingresar en un arreglo los datos de N empleados, para

luego poder listarlos ordenados alfabéticamente por apellido en forma ascendente.

Utilizar funciones para:

- Cargar la lista de empleados.
- Solicitar los datos de un empleado en particular.
- Mostrar los datos de un empleado en particular.
- Ordenar la lista de empleados.
- Mostrar la lista de empleados.

Para cada elemento de la lista debe usar una estructura que contenga: la inicial del apellido, la inicial del nombre y el número de documento. Al cargar cada empleado, la inicial del apellido y la inicial del nombre deben ser caracteres alfabéticos y guardarse en mayúscula. No utilizar un arreglo auxiliar para el ordenamiento.

Como sugerencia puede usar funciones con estas cabeceras:

```
void solicitarDatosEmpleado(struct Empleado *emp);
void mostrarDatosEmpleado(struct Empleado emp);
void cargarListaEmpleados(struct Empleado lista[], int numEmpleados);
void ordenarListaEmpleados(struct Empleado lista[], int numEmpleados);
void mostrarListaEmpleados(struct Empleado lista[], int numEmpleados);
```

28. Ingresar los datos de N Instrumentos en un arreglo, luego mostrar todos los instrumentos registrados ordenados por marca de forma ascendente.

    Utilizar la siguiente estructura:

```
typedef struct{
      int codigo;    // Código del instrumento
      char marca;    // Solo la inicial
      char modelo;   // Solo la inicial
      char frecuencia;
}instrumento;
```

    Utilizar funciones para:

    - Cargar el vector de instrumentos.
    - Ordenar los instrumentos (arreglo unidimensional de instrumentos) por el campo marca.
    - Mostrar el arreglo de instrumentos.

29. Con el arreglo del ejercicio anterior, implementar una función para mostrar los instrumentos agrupados por frecuencia de calibración. Deberá solicitar al usuario que ingrese, mediante un submenú, qué frecuencia de calibración desea consultar, puede ser A-ANUAL, B-SEMESTRAL, C-TRIMESTRAL y luego mostrar por pantalla todos los instrumentos que corresponden a la misma. Utilizar funciones para:
    - La lectura de la frecuencia por la cual agrupar.
    - Buscar todos los instrumentos que corresponden a una frecuencia ingresada.

Como sugerencia puede usar una función con esta cabecera:

```
void mostrarInstrumentosPorFrec(instrumento lista[], int orden, char frec);
```

Problemas propuestos: Arreglos Bidimensionales

30. Realizar un programa que permita cargar un arreglo bidimensional con N filas y M columnas cuyos elementos sean números enteros y luego lo presente por pantalla. Resolver sin utilizar funciones para la carga y presentación del arreglo.

    Ejemplo de un arreglo bidimensional de 4 columnas por 3 filas:

    4 8 -2 1

    0 3 6 -7

    -9 2 5 3

Resolver los siguientes ejercicios utilizando funciones para la carga y presentación por pantalla de los arreglos y para la realización de controles de datos y toda otra operación que se indique.

31. Realizar un programa que permita cargar un arreglo bidimensional con N filas y M columnas cuyos elementos sean números enteros y luego lo presente por pantalla recorriendolo por filas. Utilizar una función para controlar que los valores de la cantidad de filas y columnas sean mayores que 1 y menores que el máximo definido para el arreglo.
32. Realizar un programa que permita cargar un arreglo bidimensional con N filas y M columnas cuyos elementos sean caracteres y luego lo presente por pantalla recorriendolo por columnas. Utilizar una función para controlar que los valores de la cantidad de filas y columnas sean mayores que 1 y menores que el máximo definido para el arreglo.
33. Se desea mantener en una tabla la lista de empleados y la recaudación semanal de cada uno de ellos, considerando los días de lunes a sábado, como se muestra en el ejemplo. Realizar un programa para poder cargar la recaudación de la semana de cada empleado y luego poder consultarla. Utilizar un arreglo bidimensional de orden M x N de números reales, donde M representa la cantidad de empleados y N representa los días.

    Ejemplo para 4 empleados:

	Lunes	Martes	Miérc.	Jueves	Viernes	Sábado
Empleado 1	5125,00	1234,40	345,00	12345,00	0,00	12345,00
Empleado 2	5125,00	4234,40	345,00	1864,40	0,00	1245,00
Empleado 3	4234,40	1864,40	2300,00	2300,00	0,00	2345,00

Empleado 4	5125,00	135,40	345,00	2300,76	0,00	7345,00

34. Modificar el programa anterior agregando un menú para realizar las diferentes operaciones:
    - Cargar el arreglo con la recaudación de la semana de cada empleado pero ahora controlando que los valores de la recaudación de la semana sean valores positivos.
    - Mostrar el arreglo.
    - Calcular y mostrar la recaudación semanal de un empleado en particular.
    - Calcular cuál fue el día de la semana que más recaudó y el monto.