Laboratorio I - Programación I

Guia de ejercicios

Tema 1 (Entradas, procesos y salidas)

Ejercicio 1-1: Ingresar dos números enteros, sumarlos y mostrar el resultado.

Ejemplo:

Si ingresamos 3 y 2 el programa mostrará: **“La suma entre 3 y 2 da como resultado 5”**

Ejercicio 1-2: ingresar 3 números y mostrar cuál de los tres es el mayor.

Ejemplo:

Si ingresamos 5, 9 y 3 el programa mostrará: **“El mayor de los números es el segundo”**

Ejercicio 1-3: ingresar 3 números y mostrar el número del medio, sólo si existe. En caso de que no exista también informarlo.

Ejemplo:

1 5 3 el 3 es del medio

5 1 5 no hay numero del medio

3 5 1 el 3 es del medio

3 1 5 el 3 es del medio

5 3 1 el 3 es del medio

Tema 2 (Estructuras repetitivas, máximos, mínimos, contadores)

Ejercicio 2-1: Ingresar 5 números enteros calcular y mostrar el promedio de los números. Probar la aplicación con distintos valores.

Ejemplo 1: 3 - 5 - 7 - 9 - 1

Ejemplo 2: 2 - 1 - 8 -1 - 2

Ejercicio 2-2:

Ingresar 10 números enteros, distintos de cero. Mostrar:

a. **Cantidad de pares e impares.**
b. **El menor número ingresado.**
c. **De los pares el mayor número ingresado.**
d. **Suma de los positivos.**
e. **Producto de los negativos.**

Ejercicio 2-3:

Debemos alquilar el servicio de transporte para llegar a Mar del Plata con un grupo de personas, de cada persona debemos obtener los siguientes datos:

número de cliente,

estado civil **('s' para soltero", 'c' para casado o 'v' viudo)**,

edad**(solo mayores de edad, más de 17)**,

temperatura corporal **(validar por favor)**

y genero**('f' para femenino, 'm' para masculino, 'o' para no binario).**

NOTA: El precio por pasajero es de $600.

Se debe informar (solo si corresponde):

a) La cantidad de personas con estado "viudo" de más de 60 años.

b) el número de cliente y edad de la mujer soltera más joven.

c) cuánto sale el viaje total sin descuento.

d) si hay más del 50% de los pasajeros que tiene más de 60 años , el precio final tiene un descuento del 25%, que solo mostramos si corresponde.

Ejercicio 2-4:

Ingresar 5 caracteres e informar cuantas letras p (minúsculas) se ingresaron.

Tema 3 (Funciones)

Ejercicio 3-1:

Crear una función que muestre por pantalla el número flotante que recibe como parámetro.

Ejercicio 3-2:

Crear una función que permita determinar si un número es par o no. La función retorna 1 en caso afirmativo y 0 en caso contrario. Probar en el main.

Ejercicio 3-3:

Crear una función que pida el ingreso de un entero y lo retorne.

Ejercicio 3-4:

Especializar la función anterior para que permita validar el entero ingresado en un rango determinado.

Ejercicio 3-5:

Realizar un programa en donde se puedan utilizar los prototipos de la función Sumar en sus 4 combinaciones.

int Sumar1(int, int);

int Sumar2(void);

void Sumar3(int, int);

void Sumar4(void);

Tema 4 (Bibliotecas - Recursividad)

Ejercicio 4-1:

Realizar un programa EN EL MAIN que permita calcular y mostrar el factorial de un número. **Utilizar la función PedirNumero de la ejercitación 3-4.**

Por ejemplo:

5! = 5*4*3*2*1 = 120

Ejercicio 4-2:

Realizar un programa que permita la carga de temperaturas en celsius y fahrenheit , validando que las temperaturas ingresadas estén entre el punto de congelación y ebullición del agua para cada tipo de escala.

Las validaciones se hacen en una biblioteca.

Las funciones de transformación de fahrenheit a celsius y de celsius a fahrenheit se hacen en otra biblioteca.

Tema 5 (Vectores -Vectores y funciones)

Ejercicio 5-1:

Pedir el ingreso de 5 números. Mostrarlos y calcular la sumatoria de los mismos.

Ejercicio 5-2:

Pedir el ingreso de 10 números enteros entre -1000 y 1000. Determinar:

a) Cantidad de positivos y negativos.
b) Sumatoria de los pares.
c) El mayor de los impares.
d) Listado de los números ingresados.
e) Listado de los números pares.
f) Listado de los números de las posiciones impares.

Anexo 5-2

g) Los números que se repiten
h) Los positivos creciente y los negativos de manera decreciente

Se deberán utilizar funciones y vectores.

Ejercicio 5-3:

Realizar un programa que permita el ingreso de 10 números enteros distintos de cero. La carga deberá ser aleatoria (todos los elementos se inicializan en cero por default). Determinar el promedio de los positivos, y del menor de los negativos la suma de los antecesores (Según la recta numérica de los reales, hasta llegar a cero).

Utilizar funciones y vectores.

ANEXO TEMA VECTORES:

ANEXO 5-1:

Armar un Menú de Opciones con las siguientes opciones

1-Inicializar

2-Cargar

3-Mostrar

4-Calcular Promedio

5-Ordenar

Al ingresar a cada opción del menú deberá imprimir en pantalla el nombre del mismo. Ej: Si se presiona la opción 1 mostrar por pantalla “Ud. ha seleccionado lo opción 1-Inicializar”

ANEXO 5-2:

Realizar un programa que desde el main llame o invoque a la función que muestre el Menú del punto anterior.

Utilizar funciones y bibliotecas.

ANEXO 5-3:

Crear un Array de 5 elementos de tipo numérico donde se cargarán edades.

Agregar la funcionalidad para la opción 1 del Menú del ejercicio anterior utilizando funciones, es decir una función que inicialice el Array.

Agregar la funcionalidad para la opción 2 del Menú utilizando funciones, es decir una función que le pida los números o edades al usuario para cargar el Array de forma secuencial.

ANEXO 5-4:

Modificar la funcionalidad de la opción 2 del Menú para que la carga del Array sea de forma aleatoria.

Agregar la funcionalidad de la opción 3 del Menú para mostrar por pantalla los elementos del Array, o sea las edades.

ANEXO 5-5:

Modificar la funcionalidad de la opción 3 del Menú de forma que existan 2 funciones una que muestre solamente un elemento o edad y otra que muestre todos los elementos o edades. La función que muestra todos los elementos o edades debe llamar a la que muestra solo uno.

ANEXO 5-6:

Realizar una función que calcule el promedio de los números ingresados en el Array y agregarlo a la opción de Menú 4. La misma deberá imprimir el resultado por pantalla.

ANEXO 5-7:

Modificar la función que calcula el promedio en la opción 4 del Menú, para que el resultado del promedio lo devuelva por retorno y sea main quien lo imprima por pantalla.

ANEXO 5-8:

Modificar la función que calcula el promedio en la opción 4 del Menú, para que devuelva el resultado del promedio por parámetro por referencia. El retorno de la misma debe indicar si funcionó correctamente.

Tema 6 (Ordenamiento BubbleSort - Cadena de caracteres)

Ejercicio 6-1:

Disponemos de dos variables numéricas (a y b). Realizar un algoritmo que permita el intercambio de valores de dichas variables.

Ejercicio 6-2:

Realizar un programa que permita el ingreso de 10 números enteros (positivos y negativos). Necesito generar un listado de los números positivos de manera creciente y negativos de manera decreciente. (Como máximo 5 estructuras repetitivas)

Ejemplo:

Listado 1 : 4, 5, 6, 10, 18, 29

Listado 2 : -1,-5,-9,-12

Ejercicio 6-3:

Pedirle al usuario su nombre y apellido (en variables separadas, una para el nombre y otra para el apellido). Ninguna de las dos variables se puede modificar.

Debemos lograr guardar en una tercer variable el apellido y el nombre con el siguiente formato:

Por ejemplo:

Si el nombre es juan ignacio y el apellido es gOmEz, la salida debería ser:

Gomez, **J**uan **I**gnacio

Tema 7 (ARRAYS PARALELOS)

Ejercicio 7-1:

Una empresa importadora que comercializa productos Apple, decide registrar de sus productos los siguientes datos:

- **idProducto** (numerico)
- **descripcion** (alfanumérico)
- **nacionalidad** (numérico, por el momento utilizaremos un define: EEUU - CHINA - OTRO)
- **tipo** (numérico, por el momento utilizaremos un define: IPHONE -MAC - IPAD - ACCESORIOS)
- **precio** (numérico decimal)

Realizar un programa que permita interactuar con un menú de usuarios con las siguientes opciones:

1. **ALTA** Producto: Se ingresan los datos de UN solo producto. Siempre y cuando haya espacio disponible en el array.
2. **BAJA** Producto: A partir del ingreso del ID. Si existe el producto desaparece de la lista, dejando espacio disponible para un nuevo producto.
3. **MODIFICACIÓN** Producto: A partir del ingreso del ID. Si existe se podrá modificar el precio o el tipo.
4. **LISTADO** Productos.
5. **LISTADO** ordenado por precio.
6. **LISTADO** ordenado por descripción.

Tema 8

Ejercicio 8-1:

Crear la estructura Jugador (nombre, goles metidos, partidos jugados, promedio de goles).
Crear una función que permita cargar los datos de un jugador y otra que los muestre.
Una tercera función calculará el promedio de goles.

Ejercicio 8-2:

Realizar el ejercicio 7-1 utilizando estructuras.

Clase 9

Ejercicio 9-1:

Una empresa importadora que comercializa productos Apple, decide registrar de sus productos los siguientes datos:

- **idProducto** (numerico)
- **descripcion** (alfanumérico)
- **nacionalidad** (numérico, por el momento utilizaremos un define: EEUU - CHINA - OTRO)
- **tipo** (numérico, por el momento utilizaremos un define: IPHONE -MAC - IPAD - ACCESORIOS)
- **precio** (numérico decimal)

Realizar un programa que permita interactuar con un menú de usuarios con las siguientes opciones:

1. **ALTA** Producto: Se ingresan los datos de UN solo producto. Siempre y cuando haya espacio disponible en el array.
2. **BAJA** Producto: A partir del ingreso del ID. Si existe el producto desaparece de la lista, dejando espacio disponible para un nuevo producto.

3. **MODIFICACIÓN** Producto: A partir del ingreso del ID. Si existe se podrá modificar el precio o el tipo.
4. **LISTADO** Productos.
5. **LISTADO** ordenado por precio.
6. **LISTADO** ordenado por descripción.

   Agregar los siguientes informes:

7. El/los productos más caros.
8. Precio promedio por tipo de producto.

Clase 10

Ejercicio 10-1:

Dada la estructura eAlumno (legajo, nombre, nota1, nota2, promedio), de qué manera la podría relacionar con la estructura eCurso (idCurso, descripcion, duracion), teniendo en cuenta que un alumno puede cursar solo un curso

Ejercicio 10-2:

Crear las siguientes estructuras en c, y definir la manera mas optima de relacionarlas.

1. Producto-Proveedor (Donde el producto solo puede tener un único proveedor)

Producto	Proveedor
idProducto (int)	idProveedor(int)
descripcion(string)	razonSocial(string)
precio(float)	direccion(eDireccion)

2. Dueño-Mascota (Donde la mascota solo puede tener un único dueño)

Dueño	Mascota
idDueño(int)	idMascota(int)
nombre(string)	nombre(string)
edad (int)	raza(eRaza)
sexo(char)	sexo(char)

3. Alumno-Localidad (Donde un alumno solo vive en una localidad)

Alumno	Localidad
idAlumno(int)	idLocalida(int)
nombre(string)	descripcion(string)
	codigoPostal(int)

Ejercicio 10-3:

Una empresa importadora que comercializa productos Apple, decide registrar de sus productos los siguientes datos:

- **idProducto** (numerico)
- **descripcion** (alfanumérico)
- **nacionalidad** (numérico, por el momento utilizaremos un define: EEUU - CHINA - OTRO)
- **tipo** (numérico)
- **precio** (numérico decimal)

Agregar la estructura etipoProducto, que definirá los siguientes campos:

- **idTipo** (numérico)
- **descripcionTipo**(alfanumérico)

Para esta estructura en principio trabajaremos con datos hardcodeados:

idTipo	descripcionTipo
1000	Iphone
1001	Ipad
1002	Mac
1003	Accesorios

Realizar un programa que permita interactuar con un menú de usuarios con las siguientes opciones:

1. **ALTA** Producto: Se ingresan los datos de UN solo producto. Siempre y cuando haya espacio disponible en el array. Al momento de dar de alta el producto, el usuario podrá elegir el tipo de producto, de una lista que se le desplegará en pantalla.
2. **BAJA** Producto: A partir del ingreso del ID. Si existe el producto desaparece de la lista, dejando espacio disponible para un nuevo producto.
3. **MODIFICACIÓN** Producto: A partir del ingreso del ID. Si existe se podrá modificar el precio o el tipo. Si modifica el tipo de producto, se utilizara el mismo criterio que para dar de alta.
4. **LISTADO** Productos.
5. **LISTADO** ordenado por precio.
6. **LISTADO** ordenado por descripción.

   Agregar los siguientes informes:

7. El/los productos más caros.
8. Precio promedio por tipo de producto. Se deberá mostrar la descripción del tipo y a continuación el precio promedio.

Se agregan los siguientes listados:

9. El listado de todos los productos con la descripción del tipo.
10. Por cada tipo la lista de productos.

Clase 11

Ejercicio 11-1:

Una empresa importadora que comercializa productos Apple, decide registrar de sus productos los siguientes datos:

- **idProducto** (numerico)
- **descripcion** (alfanumérico)
- **nacionalidad** (numerico)
- **tipo** (numérico)
- **precio** (numérico decimal)

Agregar la estructura etipoProducto, que definirá los siguientes campos:

- **idTipo** (numérico)
- **descripcionTipo**(alfanumérico)

Para esta estructura en principio trabajaremos con datos hardcodeados:

idTipo	descripcionTipo
1000	Iphone

1001	Ipad
1002	Mac
1003	Accesorios

Agregar la estructura eNacionalidad, que definirá los siguientes campos:

- **idNacionalidad** (numérico)
- **descripcionNacionalidad**(alfanumérico)

Para esta estructura en principio trabajaremos con datos hardcodeados:

idNacionalidad	descripcionNacionalidad
1	EEUU
2	CHINA
3	OTRO

Realizar un programa que permita interactuar con un menú de usuarios con las siguientes opciones:

1. **ALTA** Producto: Se ingresan los datos de UN solo producto. Siempre y cuando haya espacio disponible en el array. Al momento de dar de alta el producto, el usuario podrá elegir el tipo de producto, de una lista que se le desplegará en pantalla.
2. **BAJA** Producto: A partir del ingreso del ID. Si existe el producto desaparece de la lista, dejando espacio disponible para un nuevo producto.
3. **MODIFICACIÓN** Producto: A partir del ingreso del ID. Si existe se podrá modificar el precio o el tipo. Si modifica el tipo de producto, se utilizará el mismo criterio que para dar de alta.
4. **LISTADO** Productos.
5. **LISTADO** ordenado por precio.
6. **LISTADO** ordenado por descripción.

   Agregar los siguientes informes:

7. El/los productos más caros.
8. Precio promedio por tipo de producto. Se deberá mostrar la descripción del tipo y a continuación el precio promedio.
9. El listado de todos los productos con la descripción del tipo.
10. Por cada tipo la lista de productos.

Se agregan los siguientes listados:

11. El/los tipos de productos con mas productos elaborados.

Clase 12

Ejercicio 12-1:

Agregar al ejercicio 11-1 las siguientes consultas:

12. La nacionalidad que ***solo*** fabrica Iphone.
13. Los productos ordenados por nacionalidad alfabéticamente.
14. La nacionalidad con más tipos de productos fabricados.
15. El precio promedio de productos por nacionalidad

Ejercicio 12-2:

Agregar al ejercicio 12-1 las siguientes TAREAS:

16. Alta,baja y modificación de TIPO

17. Alta,baja y modificación de NACIONALIDAD

Clase 13

Ejercicio 13-1:

Al momento de realizar la baja de la nacionalidad o del tipo, tener en cuenta que los productos al depender de estas, deberían ser eliminados también. A esto lo denominamos borrado en cascada.

Clase 14

Ejercicio 14-1:

Agregar la estructura eFabricanteDePantalla (idFabricante y descripcion).

Esta estructura se relaciona con el tipo de producto. Un fabricante puede producir pantallas para varios tipos, pero un tipo de producto solo tendrá un fabricante de pantalla. Solo se trabajará con los datos hardcodeados de la estructura fabricante.

IdFabricante	Descripcion
1	Foxconn
2	Sharp
3	Japan Display Inc

Ejercicio 14-2:

El programa deberá mostrar en otra consulta los productos cuyas pantallas hayan sido fabricadas por un fabricante que ingresó el usuario (validar).

Clase 15 (Punteros)

Ejercicio 15-1:

Realizar el ordenamiento de un vector de enteros. Para ello deberán crear una función que se encargue de realizar el intercambio de los dos valores que se van a ordenar (función swap), que solo puede recibir dos parámetros.

Ejercicio 15-2:

Realizar una función que reciba como parámetro un puntero a entero. La función le pedirá al usuario que cargue un valor por medio del puntero. Retornará si pudo cargarlo o no.

Ejercicio 15-3: (No entregar)

Modificar la biblioteca input.h para que a partir de ahora trabaje con punteros. El retorno de cada función será exclusivamente para el manejo de estados.

Ejercicio 15-4: (No entregar)

Refactorizar el TP2 para que trabaje con punteros.

Ejercicio 15-5

Escriba una función que reciba dos números enteros y los devuelva ordenados. Es decir que en el primer parámetro debe devolver el menor valor y en el 2do. el mayor.

Ejercicio 15-6

Realizar una función que reciba como parámetros dos vectores. La función deberá determinar si el segundo vector está contenido en el primero.

Clase 16 (Punteros a funciones)

Ejercicio 16-1:

Realizar una función que reciba un puntero a char y dos char. La función deberá reemplazar en la cadena cada ocurrencia del primer carácter recibido, por el segundo. Retornando la cantidad de reemplazos o -1 en caso de error.

Ejercicio 16-2:

Realizar una función que reciba como parámetros un array de enteros y un entero por referencia. La función calculará el valor máximo de ese array y utilizará el valor por referencia para retornar ese valor.

Ejercicio 16-3:

Realizar una función que respete el siguiente prototipo:

int Calculador(int, int(*pFunc)(int, int), int, int*);

Dicha función deberá recibir dos operandos y un operador representado mediante la función pasada como parámetro. El puntero a entero, servirá como parámetro de salida y guardará el resultado de la operación.

Ejercicio 16-4:

Realizar una función que respete el siguiente prototipo:

int Ordenador(eProducto*,int , int(*pFunc)(eProducto*, eProducto*));

La función Ordenador recibirá como parámetros:

a. el array de productos, dado por el puntero a eProducto
b. la cantidad de elementos del array
c. el criterio de ordenamiento, dado por la función pFunc.

Veamos un ejemplo de la función criterio:

int CompararPorMarca(eProducto*,eProducto*);

int ComparaPorPrecio(eProducto*,eProducto*);

Clase 17 (Memoria Dinamica)

Ejercicio 17-1:

Realizar una función que retorne un puntero a eProducto. Dentro de esta función deberá crear un producto hardcodeado, un puntero y se apuntará ese puntero al producto. Mostrar el producto cargado 5 veces desde el main. Observar el comportamiento del programa.

Ejercicio 17-2:

Realizar una función que reciba como parámetro un array de enteros, su tamaño y un entero. La función se encargará de buscar el valor entero y borrará la primera ocurrencia del mismo. El array deberá reestructurarse dinámicamente.

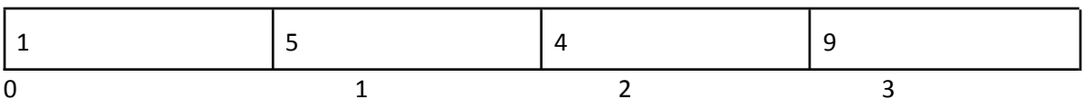
valor = 2

5	4	2	1	3
0	1	2	3	4

5	4	1	3
0	1	2	3

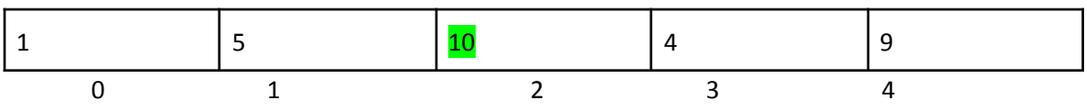
Ejercicio 17-3:

Realizar una función que reciba como parámetro un array de enteros, su tamaño, un valor entero y un índice. La función se encargará de insertar el valor entero en el índice especificado. El array deberá reestructurarse dinámicamente.



indice=2

valor = 10



Clase 18 (Archivos)

Ejercicio 18-0:

Guardar en una variable nombre y apellido. Guardar el dato en un archivo de texto, leerlo y luego mostrarlo.

Ejercicio 18-1:

Crear una función que reciba como parámetros un array con 5 nombres (que estarán previamente hardcodearlos). La función deberá guardar la lista de nombres en un archivo de texto.

```
int EscribirNombresTXT(char* nombres, char* path);
```

Otra función se encargará de la lectura de dicha lista.

```
int LeerNombresTXT(char* nombres, char* path);
```

Ejercicio 18-2:

Crear la estructura alumno (legajo, nombre, nota), un array de este tipo y hardcodearlo con 3 datos. Escribir en un archivo de texto los datos del array, separando cada campo con una coma y cada alumno con un salto de línea. Mostrar los datos luego de haber leído los datos desde el archivo, en un array de cadenas.

Ejercicio 18-3:

Escribir y leer un archivo binario con 5 números enteros. Mostrar los datos luego de la lectura del archivo

Ejercicio 18-4:

Escribir y leer un archivo binario con 5 alumnos (utilizar la estructura del ejercicio 18-2). Mostrar los datos luego de la lectura del archivo.

Ejercicio 18-5:

Leer un archivo CSV (generado en mockaroo). Parsearlo y guardarlo en un array del tipo de dato correspondiente. Mostrar los datos por pantalla.