Algoritmos y Estructura de Datos – Final	14/02/2022	Nota:
Nombre y Apellido:	Legajo:	Prof cursada:

Un comercio de electrodomésticos necesita realizar un sistema para el control de venta de sus productos en un mes determinado. Se sabe que vende exactamente 2500 electrodomésticos distintos. Cuenta con:

A) Una **array** que contiene el detalle de cada uno de los **2500** electrodomésticos. Estos se encuentran ya cargados. Cada electrodoméstico ocupa una posición en el array y están **ordenados por código de electrodoméstico ascendente**. El código de electrodoméstico es una cadena de 6 letras aleatorias. El struct que representa a cada electrodoméstico en el array es el siguiente:

```
struct Electrodomestico {
        char codigoElectrodomestico[6 + 1];
        char nombreElectrodomestico[250 + 1];
        float precioUnitario;
};
```

B) Un **archivo** *ventas.dat* que contiene los electrodomésticos que compran los clientes. Cada registro contiene:

```
struct Venta {
        char codigoElectrodomestico[6 + 1];
        int diaDelMes; // valores entre 1 y 31
        int cantidadVendida;
        int dniCliente;
        char nombreCliente[100 + 1];
};
```

Se requiere poder analizar las ventas en un día dado y para un electrodoméstico específico. Para ello, se pide armar una **matriz** de datos. Las filas de la matriz deben representar los días del mes (31 días máximo). Las columnas representan a cada electrodoméstico. La estructura de cada celda de la matriz contiene:

o Cantidad vendida del electrodoméstico en ese día	o Lista de clientes que lo compraron en ese día

La estructura del info de cada nodo de la lista de clientes:

o DNI del cliente	o Nombre del cliente

<u>Resuleva los siguientes puntos:</u>

1) (**1 punto)** Codifique la definición de los **tipos de datos** a utilizar (matriz y lista de clientes de cada día-electrodoméstico)

2) (3 puntos) Desarrolle:

a) ***int obtenerPosFilaEnMatriz(int diaDelMes)*** que recibe un día del mes y retorna la posición de la fila en la matriz correspondiente. El día 1 se encontrará en la fila 0, el día 2 en la fila 1, y así sucesivamente hasta el día 31 en la fila 30.

b) ***int obtenerPosColumnaEnMatriz(Electrodomestico electrodomesticos[], char codigoElectrodomestico[6 + 1])*** que retorna la posición de la columna en la matriz según el código del electrodoméstico. Para esto usa el código del electrodoméstico para buscar su posición en el array de electrodomésticos. La posición encontrada es la misma que se utiliza para la columna de la matriz. Siempre se invoca con códigos que existen en el array.

3) (4 puntos) Desarrolle el procedimiento **cargarMatriz** que debe cumplir con lo siguiente:

- Recibe 3 parámetros: el archivo de ventas ya abierto, el array de electrodomésticos y la matriz (cada una de sus celdas ya se encuentra inicializada: cantidad vendida en 0 y lista de clientes en NULL).
- Se debe leer secuencialmente el archivo de ventas.

4) (2 puntos) Desarrolle el procedimiento **mostrarClientes** que recibe por parámetro la matriz (luego de ser cargada en el punto 3), un número de fila y un número de columna, y muestra por pantalla, para esa fila-columna, la cantidad de ventas realizadas y todos los DNI de clientes que realizaron las compras. La lista de clientes NO debe ser destruida.

BONUS) (1 punto) Qué estructura de datos hubiese utilizado en el caso de no conocer la cantidad total de electrodomésticos que vende el comercio (cantidad de electrodomésticos variable)? El punto se obtiene si se define y justifica correctamente y sin ambigüedades la estructura a utilizar.

Puede utilizar sin desarrollar cualquiera de las funciones/procedimientos a continuación. Recuerde redefinir los tipos de datos genéricos en las firmas de las funciones que utiliza por los tipos de su examen.

Arrays

```
void agregar(TipoDeDatoElemento arr[], int& len, TipoDeDatoElemento v)
int insertarOrdenado(TipoDeDatoElemento arr[], int& len, TipoDeDatoElemento v)
int buscar(TipoDeDatoElemento arr[], int len, TipoDeLaClaveDeBusqueda v)
int busquedaBinaria(TipoDeDatoElemento arr[], int len, TipoDeLaClaveDeBusqueda v, bool& enc)
int buscaEInsertaOrdenado(TipoDeDatoElemento arr[], int& len, TipoDeDatoElemento v, bool&
enc)
void ordenar(TipoDeDatoElemento arr[], int len)
void eliminar(TipoDeDatoElemento arr[], int& len, int pos)
```

Listas simplemente enlazadas

```
void insertaPrimero(TipoNodo*& raiz, TipoInfo v)
TipoNodo* agregarAlFinal(TipoNodo*& raiz, TipoInfo v)
TipoNodo* insertarOrdenado(TipoNodo*& raiz, TipoInfo v)
TipoNodo* buscar(TipoNodo* raiz, TipoDeLaClaveDeBusqueda v)
TipoNodo* buscaEInsertaOrdenado(TipoNodo*& raiz, TipoInfo v, bool& enc)
void ordenar(TipoNodo*& raiz)
TipoInfo suprimePrimero(TipoNodo*& raiz)
void eliminar(TipoNodo*& raiz,  TipoDeLaClaveDeBusqueda v)
void liberar(TipoNodo*& raiz)
```