IMPORTANTE:

- **Crear un proyecto con su Nombre y Apellido.**
- **Realizar todas las funciones que se indican.**
- **Añadir comentarios a su código identificando cada inciso.**

Como parte del proceso previo al TP Final de la cátedra Laboratorio 2, se nos ha encomendado realizar un pequeño módulo para organizar información existente en un archivo de Compras de Clientes. El archivo consta de la siguiente estructura de datos:

```
typedef struct{ int
   NroCliente;char
   NyA[30]; int
   NroPedido;
    float MontoCompra;
 } RegistroCompraCliente;
```

Contaremos con una lista simplemente vinculada con la siguiente estructura:

```
typedef struct{ int
   NroCliente;char
   NyA[30];
} Cliente;

typedef struct{ int
   NroPedido;
   Cliente cliente;
   float MontoCompra;
   struct NodoPedido * siguiente;
} NodoPedido;
```

Se nos pide desarrollar la siguiente funcionalidad:

1. Crear la función (crearNodoPedido), que reciba como parámetro un registro de tipo RegistroCompraCliente, y devuelva un nodo de tipo nodoPedido.

2. Crear la función (agregarNodoPedido), que reciba la lista y un registro del tipo RegistroCompraCliente y agregue ese nodo al final de la lista. Retornar la lista.

3. Crear una función (crearListaPedidos) que permita recorrer el archivo y crear la lista de pedidos. La función puede recibir como parámetro el nombre del archivo o puede no recibiré ese parámetro si así lo prefiere. Retornar la lista.

4. Crear una función (mostrarLista) que permita recorrer y mostrar la lista. La misma debe ser recursiva.

5. Crear la función (liberarLista), que reciba la lista y hacer lo indicado. La función no debe retornar nada (no importa que la cabecera no quede en NULL).

6. Crear la función (ingresarPedido), la cual recibirá la lista y solicitará al usuario los datos del pedido para finalmente agregar ese pedido a la lista y retornar la lista. Modularizar de ser conveniente.

7. Crear la función recursiva (calcularTotalMontoCompra), la cual retornará un float con el monto total de las compras y recibirá por parámetro la lista.

8. Crear la función (copiarPedidosMayores5000) la cual recibirá como parámetros la lista y la cantidad de pedidos (como puntero) y a partir de esa info, deberá crear un arreglo dinámico de tipo RegistroCompraCliente con todos los pedidos de monto superior a 5000. Retornar un puntero al arreglo creado y actualizar el puntero recibido como parámetro (para saber cuántos pedidos se almacenaron el arreglo). Modularizar si lo considera necesario.

9. Crear la función (mostrarPedidos) que reciba el arreglo dinámico y la cantidad de pedidos cargados en el mismo para finalmente mostrar esos pedidos. La función no debe retornar nada.

10. Crear la función (armaFilaPedidos) la cual debe armar una FILA (filaPedidos) con todos los pedidos (leídos desde el archivo), cuyos montos seas mayores a 20000. La fila debe ser implementada mediante listas dobles.

11. Crear una función (buscaPedido), la cual deberá recorrer la fila en buscar del pedido (si se encuentra), cuyo nro de pedido sea: 20799. En caso de encontrarlo, mostrar la info de ese nodo por pantalla.

12. Finalmente desarrollar el main con la invocación a cada ejercicio.