

**ALUMNO: Mendez Juan Cruz**

**DIVISION: 1-C**

**TURNO: Mañana**

**DNI: 40423224**

**MATERIA: Laboratorio 1**

**DATOS: Recuperatorio Parcial 1 Laboratorio 1**

**PROFESOR: German Scarafilo**

**ARCHIVO DRIVE:**

**El video quedo algo largo, no logre calcular bien el tiempo, los primeros 5 min muestran el funcionamiento del programa, eso puede saltearse si se desea, disculpas..**

**https://drive.google.com/file/d/1hNGm0v06TwXbrFGgOknUksnc0saTxeV8/view?usp=sharing**

**PRINCIPALES FUNCIONES:**

**ALTA CLIENTE:**

**//-------------------------------------------------------------------------------------------------------------------//**

**Funcion: generarCliente()**

**/\*\***

**\* @brief carga un cliente en un auxiliar, si se verifica devuelve ese cliente con sus campos cargados..**

**\* @param id\*: Referencia donde se almacena el id asignado**

**\* @return Retorna el cliente cargado..**

**\*/**

**eCliente generarCliente(int\* id);**

**//-------------------------------------------------------------------------------------------------------------------//**

**Funcion: cargarCliente()**

**/\*\***

**\* @brief carga un cliente en la lista de clientes..**

**\* @param list: Array de tipo cliente**

**\* @param range: Largo de la cadena de clientes**

**\* @param id\*: Referencia donde se almacena el id**

**\* @return Retorna -1 si hay error con los nulls , 0 si carga bien el cliente**

**\*/**

**int cargarCliente(eCliente listaClientes[] , int rango , int\* id);**

**//-------------------------------------------------------------------------------------------------------------------//**

**ALTA PEDIDO:**

**//-------------------------------------------------------------------------------------------------------------------//**

**Funcion: preCarga()**

**/\*\***

**\* @brief Se encarga de pre cargar un pedido, busca espacio vacio, si encuentra inicializa los kilos deplasticos en 0 y**

**\* modifica el estado del pedido en pendiente y la variable isEmpty en 0 (esta cargada)**

**\* @param lista: Listado de pedidos**

**\* @param rango: Largo de la lista de pedidos**

**\* @param \*id: Referencia de contador de Id pedidos(la incrementa y asigna)**

**\* @param idClienteSeleccionado: Id del cliente seleccionado, se asigna este al id de relacion**

**\* @return Retorna -1 si hubo algun error en la inicializacion , -2 si no hay mas lugar, 0 si cargo bien**

**\*/**

**int preCarga(ePedidos lista[] , int rango , int\* id , int idClienteSeleccionado);**

**//-------------------------------------------------------------------------------------------------------------------//**

**BAJA PEDIDO:**

**//-------------------------------------------------------------------------------------------------------------------//**

**Funcion: darDeBajaPedido()**

**/\*\***

**\* @brief Da de baja un pedido , recibe el id y borra ese pedido**

**\* @param lista: Listado de pedidos**

**\* @param rango: Largo de la lista de pedidos**

**\* @return Retorna -1 si hubo algun error en la eliminacion , 0 si elimino bien**

**\*/**

**int darDeBajaPedido(ePedidos lista[], int rango , int idAborrar);**

**//-------------------------------------------------------------------------------------------------------------------//**

**MODIFICACION CLIENTE:**

**//-------------------------------------------------------------------------------------------------------------------//**

**Funcion: modificarCliente()**

**/\*\***

**\* @brief Modifica la direccion o localidad del cliente**

**\* @param lista: Listado de clientes**

**\* @param rango: Largo de la lista de clientes**

**\* @param mensaje: Mensaje de pedido de datos**

**\* @param mensajeError: Mensaje de error**

**\* @return Retorna -1 si hubo algun error con nulls , 0 si modifica bien, -2 cancelo operacion, -3 el id ingresado no existe...**

**\*/**

**int modificarCliente(eCliente lista[] , int rango , char mensaje[] , char mensajeError[]);**

**//-------------------------------------------------------------------------------------------------------------------//**

**BAJA CLIENTE:**

**//-------------------------------------------------------------------------------------------------------------------//**

**Funcion: bajaCliente()**

**/\*\***

**\* @brief Llama varias funciones dentro para dar de baja un cliente y en tal caso sus pedidos si tiene**

**\* @param listaPedidos: Listado de pedidos**

**\* @param rangoPedidos: Largo de la lista de pedidos**

**\* @param listaClientes: Listado de clientes**

**\* @param rangoClientes: Largo de la lista de clientes**

**\* @return -1 si algo falla con los nulls, -2 si el id de cliente a cargar pedido no se encuentra en la lista , -3 si cancela la operacion**

**\* 0 si se dio de baja el cliente bien junto con sus pedidos , 1 se dio de baja el cliente pero no los pedidos por que no le cargo ninguno todavia..**

**\*/**

**int bajaCliente(ePedidos listaPedidos[] , int rangoPedidos , eCliente listaClientes[] , int rangoClientes);**

**//-------------------------------------------------------------------------------------------------------------------//**

**PRINCIPALES FUNCIONES FILTRO:**

**//-------------------------------------------------------------------------------------------------------------------//**

**Funcion: contadorDePedidosPendientesOprocesadosPorCliente()**

**/\*\***

**\* @brief Recibe un idCliente, lo busca y cuenta la cantidad de pedidos pendientes o procesados para ese cliente segun el orden dado, 1 (pendientes) 0 (procesados)**

**\* @param lista: Listado de pedidos**

**\* @param rango: Largo de la lista de pedidos**

**\* @param idAcontar: IdCliente a encontrar**

**\* @return Retorna -1 si hubo algun error con los nulls , o sino la cantidad de pedidos pendientes para ese cliente**

**\*/**

**int contadorDePedidosPendientesOprocesadosPorCliente(ePedidos lista[] , int rango , int idAcontar , int orden);**

**//-------------------------------------------------------------------------------------------------------------------//**

**Funcion: acumuladorPlasticosPorCliente()**

**/\*\***

**\* @brief Recibe un idCliente, lo busca y acumula la cantidad de kilos de PP en todos sus pedidos**

**\* @param lista: Listado de pedidos**

**\* @param rango: Largo de la lista de pedidos**

**\* @param idAcontar: IdCliente a encontrar**

**\* @param \*acumulado: Referencia donde almacenara el acumulado**

**\* @return Retorna -1 si hubo algun error con los nulls , o sino la cantidad de kilos de PP de todos los pedidos del cliente**

**\*/**

**int acumuladorPlasticosPorCliente(ePedidos lista[] , int rango , int idAencontrar , float\* acumulado);**

**//-------------------------------------------------------------------------------------------------------------------//**

**Funcion: clienteConMasPedidos ()**

**/\*\***

**\* @brief Imprime los datos del cliente con mas pedidos, ya sea que esten procesados o no..**

**\* @param listaClientes: Listado de clientes**

**\* @param rangoClientes: Largo de la lista de clientes**

**\* @param listaPedidos: Listado de pedidos**

**\* @param rangoPedidos: Largo de la lista de pedidos**

**\* @return -1 si algo falla con los nulls, 0 si imprime y calcula de manera correcta , -2 si no hay pedidos..**

**\*/**

**int clienteConMasPedidos(eCliente listaClientes[] , int rangoClientes , ePedidos listaPedidos[] , int rangoPedidos);**

**//-------------------------------------------------------------------------------------------------------------------//**

**Funcion: BuscadorClienteConMasPedidosProcesados ()**

**/\*\***

**\* @brief Imprime los datos del cliente con mas pedidos procesados por el momento**

**\* @param listaClientes: Listado de clientes**

**\* @param rangoClientes: Largo de la lista de clientes**

**\* @param listaPedidos: Listado de pedidos**

**\* @param rangoPedidos: Largo de la lista de pedidos**

**\* @return -1 si algo falla con los nulls, 0 si imprime y calcula de manera correcta , -2 si no hay pedidos procesados..**

**\*/**

**int BuscadorClienteConMasPedidosProcesados(eCliente listaClientes[] , int rangoClientes , ePedidos listaPedidos[] , int rangoPedidos);**

**//-------------------------------------------------------------------------------------------------------------------//**

**Funcion: BuscadorClienteConMasPedidosPendientes ()**

**/\*\***

**\* @brief Imprime los datos del cliente con mas pedidos pendientes por el momento**

**\* @param listaClientes: Listado de clientes**

**\* @param rangoClientes: Largo de la lista de clientes**

**\* @param listaPedidos: Listado de pedidos**

**\* @param rangoPedidos: Largo de la lista de pedidos**

**\* @return -1 si algo falla con los nulls, 0 si imprime y calcula de manera correcta , -2 si no hay pedidos pendientes..**

**\*/**

**int BuscadorClienteConMasPedidosPendientes(eCliente listaClientes[] , int rangoClientes , ePedidos listaPedidos[] , int rangoPedidos);**

**//-------------------------------------------------------------------------------------------------------------------//**

**Funcion: pedidosPendientesPorLocalidad ()**

**/\*\***

**\* @brief Pide el ingreso de una localidad y devuelve por return la cantidad de pedidos pendientes para esa localidad**

**\* @param listaPedidos: Listado de pedidos**

**\* @param rangoPedidos: Largo de la lista de pedidos**

**\* @param listaClientes: Listado de clientes**

**\* @param rangoClientes: Largo de la lista de clientes**

**\* @return -1 si algo falla con los nulls, 0 si calcula de manera correcta , -2 si no se encuentra esa ubicacion,**

**\* -3 si no hay pedidos pendientes para esa localidad , retorna sino la cantidad de pedidos pendientes calculados**

**\*/**

**int pedidosPendientesPorLocalidad(ePedidos listaPedidos[] , int rangoPedidos , eCliente listaClientes[] , int rangoClientes , eLocalidad listaLocalidades[] , int rangoLocalidades);**

**//-------------------------------------------------------------------------------------------------------------------//**

**Funcion: mostrarPedidosProcesadosConClientes ()**

**/\*\***

**\* @brief Imprime la lista de pedidos procesados, junto con los datos del cliente relacionado y la cantidad de kilos de plasticos procesados**

**\* @param listaPedidos: Listado de pedidos**

**\* @param rangoPedidos: Largo de la lista de pedidos**

**\* @param listaClientes: Listado de clientes**

**\* @param rangoClientes: Largo de la lista de clientes**

**\* @return -1 si algo falla con los nulls, 0 si muestra de manera correcta , -2 si no hay pedidos procesados**

**\*/**

**int mostrarPedidosProcesadosConClientes(ePedidos listaPedidos[] , int rangoPedidos , eCliente listaClientes[] , int rangoClientes);**

**//-------------------------------------------------------------------------------------------------------------------//**