RECUPERATORIO PRIMER PARCIAL DE LABORATORIO I

“BICICLETERIA”

ALUMNO: GASTON ALEJANDRO GAYAN

DNI: 33.300.831

FECHA DE PRESENTACION: 05 DE JULO DE 2021

1. PRESENTACION DEL PROYECTO

El presente proyecto pretende satisfacer las necesidades de una bicicletería que solicita mantener un control de los trabajos que realiza por día.

Para lograr el objetivo, se implementaron 3 estructuras macro, que son Trabajos, Clientes y Servicios y una 4ta estructura del tipo Fecha vinculada directamente con la lista de Trabajos.

Se utilizaron funciones para efectuar un Alta del trabajo realizado, Baja, modificación de la bicicleta o del servicio prestado y los informes, tales como “Listar todos los trabajos ordenados por año y ante igualdad de año por marca de bicicleta”, “Informe de cantidad de bicicletas de color rojo de acuerdo al servicio seleccionado por el usuario”, entre otras.

Cada Biblioteca contiene las funciones pertenecientes a cada estructura a fin de optimizar los recursos del CPU y obtener una mayor fluidez de datos.

A continuación, se presentan Bibliotecas de las estructuras mencionadas a párrafo primero.-

1. RELACIONES DE LAS ESTRUCTURAS UTILIZADAS

|  |
| --- |
| **eTrabajos** |
| Int idTrabajo |
| Int idServicio |
| Int idCliente |
| eFecha fecha |
| Int isEmpty |

|  |
| --- |
| **eServicios** |
| Int idServicio |
| Char descripción[50] |
| Float precio |
| Int isEmpty |

|  |
| --- |
| **eClientes** |
| Int idCliente |
| Char nombre[51] |
| Char apellido[51] |
| Int isEmpty |

|  |
| --- |
| **eFecha** |
| Int dia |
| Int mes |
| Int anio |

1. MENU DE OPCIONES:

1. ALTA TRABAJO

2. MODIFICAR TRABAJO

3. BAJA TRABAJO

4. LISTAR TRABAJO

5. LISTAR SERVICIOS

6. TOTAL

7. LISTADO ORDENADO POR MARCA DE LA BICICLETA

8. SERVICIO CON MAS TRABAJOS REALIZADOS

9. CANTIDAD DE BICICLETAS DE COLOR ROJO QUE REALIZARON UN SERVICIO

10.LISTADO DISCRIMINADO POR CLIENTE

11.SUBTOTALES POR CLIENTES

0. SALIR

1. BIBLIOTECA TRABAJOS.H (FUNCIONES MAS IMPORTANTES)

/\*\*

\* @fn int InicializarTrabajos(eTrabajos[], int)

\* @brief Inicializa el vector de trabajos,

\* colocando en el campo isEmpty el valor 1 (VACIO)

\*

\* @param trabajos

\* @param tam

\* @return

\*/

**int** **InicializarTrabajos**(eTrabajos trabajos[], **int** tam);

/\*\*

\* @fn int get\_Id(void)

\* @brief ID AUTONUMERICO

\*

\* @return DEVUELVE UN ID ESTATICO INCREMENTADO

\* POR CADA VEZ QUE SE EJECUTA LA FUNCION

\*/

**int** **get\_Id**(**void**);

/\*\*

\* @fn int BuscarLibreTrabajo(eTrabajos[], int)

\* @brief BUSCA INDICE EN EL ARRAY QUE NO SE ENCUETRE OCUPADO

\*

\* @param trabajo

\* @param tam

\* @return INDICE (EXITO) -1 (ERROR)

\*/

**int** **BuscarLibreTrabajo**(eTrabajos trabajo[], **int** tam){

/\*\*

\* @fn int AltaTrabajos(eTrabajos[], int, eServicios[], int, eBicicletas[], int, eClientes[], int)

\* @brief DA DE ALTA UN NUEVO TRABAJO SIEMPRE QUE HAYA LUGAR EN EL ARRAY

\*

\* @param trabajo

\* @param tamTrabajo

\* @param servicio

\* @param tamServicio

\* @param bicicleta

\* @param tamBicicleta

\* @param cliente

\* @param tamCliente

\* @return -1 ERROR 1 EXITO

\*/

**int** **AltaTrabajos**(eTrabajos trabajo[], **int** tamTrabajo, eServicios servicio[], **int** tamServicio, eBicicletas bicicleta[],**int** tamBicicleta, eClientes cliente[], **int** tamCliente);

/\*\*

\* @fn int Modificacion(eTrabajos[], int, eServicios[], int, eBicicletas[], int, eClientes[], int)

\* @brief MODIFICA UN TRABAJO, PUDIENDO SOLO MODIFICAR EL SERVICIO O LA BICICLETA

\*

\* @param trabajo

\* @param tam

\* @param servicio

\* @param tamServicio

\* @param bicicleta

\* @param tamBicicleta

\* @param cliente

\* @param tamCliente

\* @return 0 ERROR 1 EXITO

\*/

**int** **Modificacion**(eTrabajos trabajo[], **int** tam, eServicios servicio[],**int** tamServicio, eBicicletas bicicleta[],**int** tamBicicleta, eClientes cliente[], **int** tamCliente);

/\*\*

\* @fn int BajaTrabajos(eTrabajos[], int, eServicios[], int, eBicicletas[], int, eClientes[], int)

\* @brief BAJA LOGICA DEL TRABAJO, DESVINCULANDOLO CAMBIANDO SU VARIABLE ISEMPTY POR VACIO

\*

\* @param trabajo

\* @param tam

\* @param servicio

\* @param tamServicio

\* @param bicicleta

\* @param tamBicicleta

\* @param cliente

\* @param tamCliente

\* @return 0 ERROR 1 EXITO

\*/

**int** **BajaTrabajos**(eTrabajos trabajo[],**int** tam, eServicios servicio[], **int** tamServicio, eBicicletas bicicleta[],**int** tamBicicleta, eClientes cliente[], **int** tamCliente);

/\*\*

\* @fn int ListarTrabajos(eTrabajos[], int, eServicios[], int, eBicicletas[], int, eClientes[], int)

\* @brief LISTA TOTALIDAD DE TRABAJOS RELACIONADOS CON TODAS LAS ESTRUCTURAS

\*

\* @param trabajo

\* @param tam

\* @param servicio

\* @param tamServicio

\* @param bicicleta

\* @param tamBicicleta

\* @param cliente

\* @param tamCliente

\* @return

\*/

**int** **ListarTrabajos**(eTrabajos trabajo[], **int** tam, eServicios servicio[],**int** tamServicio, eBicicletas bicicleta[],**int** tamBicicleta, eClientes cliente[], **int** tamCliente){

/\*\*

\* @fn void MostrarUnTrabajo(eTrabajos, eServicios[], int, eBicicletas[], int, eClientes[], int, int)

\* @brief MUESTRA UN SOLO TRABAJO RELACIONADO CON TODAS LAS ESTRUCTURAS

\*

\* @param trabajo

\* @param servicio

\* @param tamServicio

\* @param bicicleta

\* @param tamBicicleta

\* @param cliente

\* @param tamCliente

\* @param condicion

\*/

**void** **MostrarUnTrabajo**(eTrabajos trabajo, eServicios servicio[], **int** tamServicio, eBicicletas bicicleta[], **int** tamBicicleta, eClientes cliente[], **int** tamCliente, **int** condicion);

/\*\*

\* @fn int OrdenarTrabajos(eTrabajos[], int, eBicicletas[], int, int)

\* @brief

\* EN CASO DE CONDICION 1 ORDENA LOS TRABAJOS PRIMERO POR AÑO,

EN CASO DE IGUALDAD, POR MARCA DE BICICLETA

\* EN CASO DE CONDICION 0 ORDENA SOLO POR MARCA DE BICICLETA

\* @param trabajo

\* @param tamTrabajo

\* @param bicicleta

\* @param tamBicicleta

\* @param condicion

\* @return

\*/

**int** **OrdenarTrabajos**(eTrabajos trabajo[],**int** tamTrabajo, eBicicletas bicicleta[], **int** tamBicicleta, **int** condicion);

/\*\*

\* @fn int TotalTrabajos(eTrabajos[], int, eServicios[], int)

\* @brief SUMA TOTAL DE TODOS LOS TRABAJOS REALIZADOS RELACIONANDO LOS

\* DATOS CON EL ARRAY DE SERVICIOS Y OBTENER PRECIO

\*

\* @param trabajo

\* @param tamTrabajo

\* @param servicio

\* @param tamServicio

\* @return 0 ERROR 1 EXITO

\*/

**int** **TotalTrabajos**(eTrabajos trabajo[], **int** tamTrabajo, eServicios servicio[],**int** tamServicio);

/\*\*

\* @fn int trabajos\_contarServicios(eTrabajos[], int, eServicios[], int)

\* @brief CUENTA LA CANTIDAD DE SERVICIOS A FIN DE BRINDAR AL USUARIO EL SERVICIO CON MAS TRABAJOS REALIZADOS

\*

\* @param trabajo

\* @param tam

\* @param servicio

\* @param tamServicio

\* @return 0 ERROR 1 EXITO

\*/

**int** **trabajos\_contarServicios**(eTrabajos trabajo[], **int** tam, eServicios servicio[], **int** tamServicio);

/\*\*

\* @fn int trabajos\_contarBiciRoja(eTrabajos[], int, eServicios[], int, eBicicletas[], int)

\* @brief CUENTA CANTIDAD DE TRABAJOS REALIZADOS A BICICLETAS DE COLOR ROJO, FILTRANDO POR EL ID DEL SERVICIO QUE INGRESA EL USUARIO

\*

\* @param trabajo

\* @param tam

\* @param servicio

\* @param tamServicio

\* @param bicicleta

\* @param tamBicicleta

\* @return 0 ERROR 1 EXITO

\*/

**int** **trabajos\_contarBiciRoja**(eTrabajos trabajo[], **int** tam, eServicios servicio[], **int** tamServicio, eBicicletas bicicleta[], **int** tamBicicleta);

/\*\*

\* @fn int listadoPorCliente(eTrabajos[], int, eClientes[], int, eBicicletas[], int, eServicios[], int)

\* @brief LISTADO DISCRIMINADO POR CLIENTE, EL USUARIO INGRESA EL ID DEL CLIENTE A BUSCAR Y SE DETALLAN LOS TRABAJOS REALIZADOS AL MISMO

\*

\* @param trabajo

\* @param tamTrabajo

\* @param cliente

\* @param tamCliente

\* @param bicicleta

\* @param tamBicicleta

\* @param servicio

\* @param tamServicio

\* @return 0 ERROR 1 EXITO

\*/

**int** **listadoPorCliente**(eTrabajos trabajo[], **int** tamTrabajo, eClientes cliente[], **int** tamCliente, eBicicletas bicicleta[], **int** tamBicicleta, eServicios servicio[], **int** tamServicio){

/\*\*

\* @fn int totalTrabajosPorCliente(eTrabajos[], int, eClientes[], int, eBicicletas[], int, eServicios[], int)

\* @brief BUSCA TRABAJOS REALIZADOS A CADA CLIENTE Y EN CASO DE QUE EXISTA,

\* ACUMULA LOS PRECIOS DE LOS SERVICIOS BRINDADOS EN LOS DIFERENTES TRABAJOS

\*

\* @param trabajo

\* @param tamTrabajo

\* @param cliente

\* @param tamCliente

\* @param bicicleta

\* @param tamBicicleta

\* @param servicio

\* @param tamServicio

\* @return 0 ERROR 1 EXITO

\*/

**int** **totalTrabajosPorCliente**(eTrabajos trabajo[], **int** tamTrabajo, eClientes cliente[], **int** tamCliente, eBicicletas bicicleta[], **int** tamBicicleta, eServicios servicio[], **int** tamServicio)

1. **BIBLIOTECA SERVICIOS.H**

/\*\*

\* @fn int ListarServicios(eServicios[], int)

\* @brief LISTA LA TOTALIDAD DE LOS SERVICIOS

\*

\* @param servicios

\* @param tam

\* @return 1 encontro servicio 0 tamaño de array vacio

\*/

**int** **ListarServicios**(eServicios servicios[], **int** tam);

/\*\*

\* @fn int MostrarServicio(eServicios)

\* @brief MUESTRA UN SOLO SERVICIO

\*

\* @param servicio

\* @return 0 servicio NULL o 1 exito

\*/

**void** **MostrarServicio**(eServicios servicio){

/\*\*

\* @fn int servicio\_buscarPorId(eServicios[], int, int)

\* @brief OBTIENE INDICE DEL SERVICIO BUSCADO POR ID

\*

\* @param servicio

\* @param tam

\* @param id

\* @return -1 Error, indice del servicio (OK)

\*/

**int** **servicio\_buscarPorId**(eServicios servicio[], **int** tamServicio, **int** id){

1. **BIBLIOTECA BICICLETAS.H**

/\*\*

\* @fn int BuscarBicicleta(eBicicletas[], int)

\* @brief OBTIENE INDICE DE LA BICICLETA BUSCADA POR ID

\*

\* @param bicicleta

\* @param tam

\* @return INDICE EXITO O -1 ERROR

\*/

**int** **BuscarBicicleta**(eBicicletas bicicleta[], **int** tam);

/\*\*

\* @fn void MostrarBicicleta(eBicicletas)

\* @brief MUESTRA UNA SOLA BICICLETA

\*

\* @param bicicleta

\*/

**void** **MostrarBicicleta**(eBicicletas bicicleta);

/\*\*

\* @fn int ListarBicicletas(eBicicletas[], int)

\* @brief MUESTRA LA TOTALIDAD DE LAS BICICLETAS

\*

\* @param bicicleta

\* @param tam

\* @return

\*/

**int** **ListarBicicletas**(eBicicletas bicicleta[], **int** tam)

/\*\*

\* @fn int bicicleta\_buscarPorId(eBicicletas[], int, int)

\* @brief BUSQUEDA POR ID DE LA BICICLETA

\*

\* @param bicicleta

\* @param tamBicicleta

\* @param id

\* @return INDICE O -1 ERROR

\*/

**int** **bicicleta\_buscarPorId**(eBicicletas bicicleta[], **int** tamBicicleta, **int** id);

1. **BIBLIOTECA CLIENTES.H**

/\*\*

\* @fn void MostrarCliente(eClientes)

\* @brief MUESTRA UN CLIENTE

\*

\* @param cliente

\*/

**void** **MostrarCliente**(eClientes cliente);

/\*\*

\* @fn int ListarClientes(eClientes[], int)

\* @brief LISTADO DE LA TOTALIDAD DE CLIENTES

\*

\* @param cliente

\* @param tam

\* @return 0 ERROR 1 EXITO

\*/

**int** **ListarClientes**(eClientes cliente[], **int** tam);

\*\*

\* @fn int cliente\_buscarPorId(eClientes[], int, int)

\* @brief BUSCA CLIENTE POR ID

\*

\* @param cliente

\* @param tam

\* @param id

\* @return INDICE DEL CLIENTE BUSCADO POR ID O -1 ERROR

\*/

**int** **cliente\_buscarPorId**(eClientes cliente[], **int** tam, **int** id)