Rial Leonardo. 1F

DNI 35994816

Recuperatorio Primer Parcial.

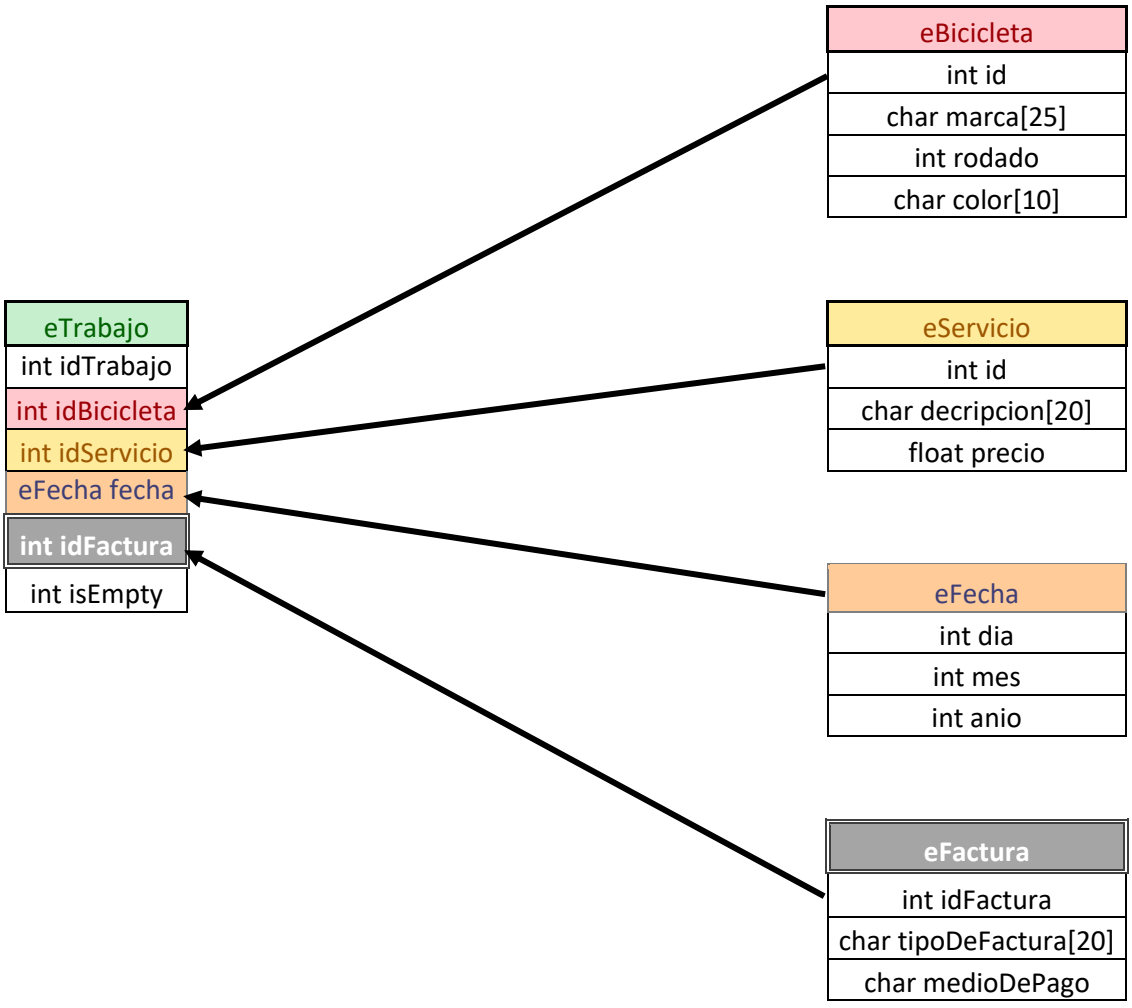
ESTRUCTURAS DEL PARCIAL:

```c
typedef struct
{
    int dia;
    int mes;
    int anio;
}eFecha;

typedef struct
{
    int idTrabajo;
    int idBicicleta;
    int idServicio;
    eFecha fecha;
    int idFactura;
    int isEmpty;
}eTrabajo;


typedef struct
{
    int id;
    char marca[25];
    int rodado;
    char color[10];
}eBicicleta;


typedef struct
{
    int id;
    char descripcion[20];
    float precio;
}eServicio;

typedef struct
{
    int id;
    char descripcion[20];
    float precio;
    int contadorTrabajos;
}eAuxServicio;
```

ESTRUCTURA AGREGADA:

```
typedef struct
 {
      int idFactura;
      char tipoFactura[20];
      char medioDePago[20];
 }eFactura;
```

DIAGRAMA ENTIDAD-RELACION:



ESTRUCTURA DE MENU:

1. ALTA
2. MODIFICACION
3. BAJA
4. LISTAR TRABAJOS POR AÑOS (Y MARCA)
5. LISTAR SERVICIOS
6. TOTAL EN PESOS POR SERVICIOS
7. LISTAR TRABAJOS POR MARCA DE BICICLETA
8. EL/LOS SERVICIOS CON MAS TRABAJOS REALIZADOS
9. LISTAR SERVICIOS CON LOS DATOS DE LAS BICICLETAS QUE SE LO REALIZARON
10. CANTIDAD DE BICICLETAS ROJAS POR SERVICIO
11. LISTAR TRABAJOS ORDENADOS POR RODADO DE BICICLETA (AGREGADO)
12. CANTIDAD DE PARCHES ABONADOS CON TARJETA (AGREGADO)
13. LISTAR TRABAJOS CUYO SERVICIO FUE ABONADO EN EFECTIVO (AGREGADO)
0. SALIR

PROTOTIPO DE FUNCIONES:

Función para imprimir por pantalla la lista de trabajos

```
/**
 * @fn int controller_listarTrabajos(eTrabajo*, int, eBicicleta*, int,
eServicio*, int, eFactura*, int)
 * @brief imprime en pantalla el array de eTrabajo tomando los datos de los otros
Arrays que recibe.
 *
 * @param listaTrabajos puntero a array de eTrabajo
 * @param size1 tamaño de array
 * @param listaBicicletas puntero a array de bicicletas
 * @param size2 tamaño de array
 * @param listaServicios puntero a array de servicios
 * @param size3 tamaño de array
 * @param listaFacturas puntero a array de facturas
 * @param size4 tamaño de array
 * @return [1]luego de imprimir al menos 1 linea [0]si no imprime nada
 */
int controller_listarTrabajos(eTrabajo* listaTrabajos, int size1, eBicicleta*
listaBicicletas, int size2, eServicio* listaServicios, int size3, eFactura*
listaFacturas, int size4);

/**
 * @fn int controller_listarUnTrabajo(eTrabajo*, int, eBicicleta*, int,
eServicio*, int, eFactura*, int, int)
 * @brief imprime un elemento del array eTrabajo segun el index indicado y
tomando los datos de los otros Arrays que recibe.
 *
 * @param listaTrabajos puntero a array de eTrabajo
 * @param size1 tamaño de array
```

```
 * @param listaBicicletas puntero a array de bicicletas
 * @param size2 tamaño de array
 * @param listaServicios puntero a array de servicios
 * @param size3 tamaño de array
 * @param listaFacturas puntero a array de facturas
 * @param size4 tamaño de array
 * @param index ubicacion en array del elemento a imprimir
 * @return [1]si imprime correctamente [0]si no imprime nada
 */
int controller_listarUnTrabajo (eTrabajo* listaTrabajos, int size1, eBicicleta*
listaBicicletas, int size2, eServicio* listaServicios, int size3, eFactura*
listaFacturas, int size4, int index);


1.  ALTA
/**
 * @fn int controller_agregarUnTrabajo(eTrabajo*, int, eBicicleta*, int,
eServicio*, int, eFactura*, int, int, int*)
 * @brief agrega un nuevo trabajo utilizando un auxilia y solicitando ingresar
los datos al usuario. al finalizar pide confirmacion.
 *
 * @param listaTrabajos puntero a array de eTrabajo
 * @param size1 tamaño de array
 * @param listaBicicletas puntero a array de bicicletas
 * @param size2 tamaño de array
 * @param listaServicios puntero a array de servicios
 * @param size3 tamaño de array
 * @param listaFacturas puntero a array de facturas
 * @param size4 tamaño de array
 * @param index ubicacion libre para agregar elemento
 * @param lastId puntero a ultima id utilizada
 * @return[1]si se agrega nuevo trabajo [0]si falla o se cancela el agregado
 */
int controller_agregarUnTrabajo (eTrabajo* listaTrabajos, int size1, eBicicleta*
listaBicicletas, int size2, eServicio* listaServicios, int size3, eFactura*
listaFacturas, int size4, int index, int* lastId);
```

2. MODIFICACION

```
/**
 * @fn int controller_opcionesAModificar(eTrabajo*, int, eBicicleta*, int,
eServicio*, int, eFactura*, int, int)
 * @brief submenu de modificacion del trabajo seleccionado
 *
 * @param listaTrabajos puntero a array de eTrabajo
 * @param size1 tamaño de array
 * @param listaBicicletas puntero a array de bicicletas
 * @param size2 tamaño de array
 * @param listaServicios puntero a array de servicios
 * @param size3 tamaño de array
 * @param listaFacturas puntero a array de facturas
 * @param size4 tamaño de array
 * @param id id del trabajo a modificar
 * @return [1]si se agrega nuevo trabajo [0]si falla o se cancela el agregado
 */
int controller_opcionesAModificar(eTrabajo* listaTrabajos, int size1, eBicicleta*
listaBicicletas, int size2, eServicio* listaServicios, int size3, eFactura*
listaFacturas, int size4, int id);

/**
 * @fn int controller_modificarTrabajo(eTrabajo*, int, eBicicleta*, int,
eServicio*, int, eFactura*, int)
 * @brief muestra array de eTrabajo y modifica el trabajo seleccionado. Solicita
confirmación del cambio.
 *
 * @param listaTrabajos puntero a array de eTrabajo
 * @param size1 tamaño de array
 * @param listaBicicletas puntero a array de bicicletas
 * @param size2 tamaño de array
 * @param listaServicios puntero a array de servicios
 * @param size3 tamaño de array
 * @param listaFacturas puntero a array de facturas
 * @param size4 tamaño de array
 * @return [1]si se agrega nuevo trabajo [0]si falla o se cancela el agregado
 */
int controller_modificarTrabajo(eTrabajo* listaTrabajos, int size1, eBicicleta*
listaBicicletas, int size2, eServicio* listaServicios, int size3, eFactura*
listaFacturas, int size4);
```

3. BAJA

```
/**
 * @fn int controller_bajaDeTrabajo(eTrabajo*, int, eBicicleta*, int, eServicio*,
int, eFactura*, int)
 * @brief muestra array de eTrabajo y da de baja al eTrabajo indicado
 *
 * @param listaTrabajos puntero a array de eTrabajo
 * @param size1 tamaño de array
 * @param listaBicicletas puntero a array de bicicletas
 * @param size2 tamaño de array
 * @param listaServicios puntero a array de servicios
 * @param size3 tamaño de array
 * @param listaFacturas puntero a array de facturas
 * @param size4 tamaño de array
 * @return [1]si se da de baja trabajo [0]si falla o se cancela la baja
 */
int controller_bajaDeTrabajo(eTrabajo* listaTrabajos, int size1, eBicicleta*
listaBicicletas, int size2, eServicio* listaServicios, int size3, eFactura*
listaFacturas, int size4);
```

4. LISTAR TRABAJOS POR AÑOS (Y MARCA)

```
/**
 * @fn void controller_ordenamientoPorMarca(eTrabajo*, int, eBicicleta*, int,
int, int)
 * @brief ordena los trabajos recibidos por marca
 *
 * @param listaTrabajos puntero a array de eTrabajo
 * @param size1 tamaño de array
 * @param listaBicicletas puntero a array de bicicletas
 * @param size2 tamaño de array
 * @param i variable de control
 * @param j variable de control
 */
void controller_ordenamientoPorMarca(eTrabajo* listaTrabajos, int size,
eBicicleta* listaBicicletas, int size2, int i, int j);

/**
 * @fn void controller_ordenamientoPorAnioYMarca(eTrabajo*, int, eBicicleta*,
int, int)
 * @brief ordena los trabajos recibidos por año y en caso de ser del mismo año,
por marca
 *
 * @param listaTrabajos puntero a array de eTrabajo
 * @param size1 tamaño de array
 * @param listaBicicletas puntero a array de bicicletas
 * @param size2 tamaño de array
 * @param i variable de control
 */
void controller_ordenamientoPorAnioYMarca(eTrabajo* listaTrabajos, int size1,
eBicicleta* listaBicicletas, int size2, int i);
```

5. LISTAR SERVICIOS

```
/**
 * @fn void servicio_listarServicios(eServicio*, int)
 * @brief imprime en pantalla la lista de servicios
 *
 * @param listaServicios puntero a array de lista de servicios
 * @param size tamaño del array
 */
void servicio_listarServicios(eServicio* listaServicios, int size);
```

6. TOTAL EN PESOS POR SERVICIOS

```
/**
 * @fn float controller_sumadorPesos(eTrabajo*, int, eServicio*, int, int)
 * @brief recibe un elemento del array de eTrabajo y busca el precio de su
servicio
 *
 * @param listaTrabajos puntero a array de eTrabajo
 * @param size1 tamaño de array
 * @param listaServicios puntero a array de servicios
 * @param size3 tamaño de array
 * @param i variable de control
 * @return retorna el precio del servicio relizado
 */
float controller_sumadorPesos(eTrabajo* listaTrabajos, int size1, eServicio*
listaServicios, int size3, int i);
```

7. LISTAR TRABAJOS POR MARCA DE BICICLETA

```
/**
 * @fn void controller_listarTrabajosPorMarca(eTrabajo*, int, eBicicleta*, int,
eServicio*, int, eFactura*, int)
 * @brief mustra por pantalla un array auxiliar de eTrabajo ordenado por marca de
bicileta
 *
 * @param listaTrabajos puntero a array de eTrabajo
 * @param size1 tamaño de array
 * @param listaBicicletas puntero a array de bicicletas
 * @param size2 tamaño de array
 * @param listaServicios puntero a array de servicios
 * @param size3 tamaño de array
 * @param listaFacturas puntero a array de facturas
 * @param size4 tamaño de array
 */
void controller_listarTrabajosPorMarca(eTrabajo* listaTrabajos, int size1,
eBicicleta* listaBicicletas, int size2, eServicio* listaServicios, int size3,
eFactura* listaFacturas, int size4);
```

8. EL/LOS SERVICIOS CON MAS TRABAJOS REALIZADOS

```
/**
 * @fn void controller_contadorDeServicios(eTrabajo*, int, eAuxServicio*, int,
int)
 * @brief suma 1 al contador de trabajo del servicio realizado
 *
 * @param listaTrabajos puntero a array de eTrabajo
 * @param size1 tamaño de array
 * @param listaServicios puntero a array de servicios
 * @param size3 tamaño de array
 * @param i variable de control
 */
void controller_contadorDeServicios(eTrabajo* listaTrabajos, int size1,
eAuxServicio* auxListaServicios, int size3, int i);

/**
 * @fn void controller_contadorDeTrabajosRealizadosPorServicio(eTrabajo*, int,
eAuxServicio*, int)
 * @brief recorre el array de eTrabajo sumando de a 1 al contador de trabajos en
array de servicios segun cada servicio realizado
 *
 * @param listaTrabajos puntero a array de eTrabajo
 * @param size1 tamaño de array
 * @param listaServicios puntero a array de servicios
 * @param size3 tamaño de array
 */
void controller_contadorDeTrabajosRealizadosPorServicio(eTrabajo* listaTrabajos,
int size1, eAuxServicio* auxListaServicios, int size3);
```

9. LISTAR SERVICIOS CON LOS DATOS DE LAS BICICLETAS QUE SE LO REALIZARON

```
/**
 * @fn void controller_mostrarBicicletaSegunServicio(eTrabajo*, int, eBicicleta*,
int, eServicio*, int, int)
 * @brief muestra una bicicleta segun el servicio indicado
 *
 * @param listaTrabajos puntero a array de eTrabajo
 * @param size1 tamaño de array
 * @param listaBicicletas puntero a array de bicicletas
 * @param size2 tamaño de array
 * @param listaServicios puntero a array de servicios
 * @param size3 tamaño de array
 * @param i variable de control
 */
void controller_mostrarBicicletaSegunServicio(eTrabajo* listaTrabajos, int size1,
eBicicleta* listaBicicletas, int size2, eServicio* listaServicios, int size3, int
i);
```

10. CANTIDAD DE BICICLETAS ROJAS POR SERVICIO

```
/**
 * @fn int controller_contarBiciRoja(eTrabajo*, int, eBicicleta*, int, int)
 * @brief recibe un eTrabajo segun el servicio indicado y cuenta si es una
bicicleta roja
 *
 * @param listaTrabajos puntero a array de eTrabajo
 * @param size1 tamaño de array
 * @param listaBicicletas puntero a array de bicicletas
 * @param size2 tamaño de array
 * @param i variable de control
 * @return retorna resultado del contador;
 */
int controller_contarBiciRoja(eTrabajo* listaTrabajos, int size1, eBicicleta*
listaBicicletas, int size2, int i);

/**
 * @fn int controller_contadorBicicletasRojasSegunServicio(eTrabajo*, int,
eBicicleta*, int, eServicio*, int)
 * @brief permite elegir un servicio y cuenta cuantas bicicletas rojas se lo
realizaron imprimiendo el resultado por pantalla
 *
 * @param listaTrabajos puntero a array de eTrabajo
 * @param size1 tamaño de array
 * @param listaBicicletas puntero a array de bicicletas
 * @param size2 tamaño de array
 * @param listaServicios puntero a array de servicios
 * @param size3 tamaño de array
 * @return [1]si conto e imprimio correctamente [0]si fallo el conteo
 */
int controller_contadorBicicletasRojasSegunServicio(eTrabajo* listaTrabajos, int
size1, eBicicleta* listaBicicletas, int size2, eServicio* listaServicios, int
size3);
```

11. LISTAR TRABAJOS ORDENADOS POR RODADO DE BICICLETA (AGREGADO)

```
/**
 * @fn void controller_ordenarTrabajoPorRodado(eTrabajo*, int, eBicicleta*, int,
int)
 * @brief ordena un array recibido por tamaño de rodado
 *
 * @param listaTrabajos puntero a array de eTrabajo
 * @param size1 tamaño de array
 * @param listaBicicletas puntero a array de bicicletas
 * @param size2 tamaño de array
 * @param i variable de control
 */
void controller_ordenarTrabajoPorRodado(eTrabajo* listaTrabajos, int size1,
eBicicleta* listaBicicletas, int size2, int i);
```

12. CANTIDAD DE PARCHES ABONADOS CON TARJETA (AGREGADO)

```
/**
 * @fn void controller_contarParchesConTarjeta(eTrabajo*, int, eBicicleta*, int,
eServicio*, int, eFactura*, int, int, int*)
 * @brief  recorre el array de eTrabajo y cuenta cuantos parches fueros abonados
con tarjeta.
 *
 * @param listaTrabajos puntero a array de eTrabajo
 * @param size1 tamaño de array
 * @param listaBicicletas puntero a array de bicicletas
 * @param size2 tamaño de array
 * @param listaServicios puntero a array de servicios
 * @param size3 tamaño de array
 * @param listaFacturas puntero a array de facturas
 * @param size4 tamaño de array
 * @param i variable de control
 * @param contador puntero a contador de parches
 */
void controller_contarParchesConTarjeta(eTrabajo* listaTrabajos, int size1,
eBicicleta* listaBicicletas, int size2, eServicio* listaServicios, int size3,
eFactura* listaFacturas, int size4, int i, int* contador);

/**
 * @fn void controller_contarParches(eTrabajo*, int, eBicicleta*, int,
eServicio*, int, eFactura*, int, int, int*)
 * @brief recibe un array de eTrabajo y suma uno al contador si se realizo un
parche y fue abonado con tarjeta
 *
 * @param listaTrabajos puntero a array de eTrabajo
 * @param size1 tamaño de array
 * @param listaBicicletas puntero a array de bicicletas
 * @param size2 tamaño de array
 * @param listaServicios puntero a array de servicios
 * @param size3 tamaño de array
 * @param listaFacturas puntero a array de facturas
 * @param size4 tamaño de array
 * @param i variable de control
 * @param contador puntero a contador de parches
 */
void controller_contarParches(eTrabajo* listaTrabajos, int size1, eBicicleta*
listaBicicletas, int size2, eServicio* listaServicios, int size3, eFactura*
listaFacturas, int size4, int i, int* contador);
```

13. LISTAR TRABAJOS CUYO SERVICIO FUE ABONADO EN EFECTIVO (AGREGADO)

```
/**
 * @fn void controller_listarUnTrabajoEnEfectivo(eTrabajo*, int, eBicicleta*,
int, eServicio*, int, eFactura*, int, int)
 * @brief imprime en pantalla un trabajo del array recibido si este fue abonado
en efectivo
 *
 * @param listaTrabajos puntero a array de eTrabajo
 * @param size1 tamaño de array
 * @param listaBicicletas puntero a array de bicicletas
 * @param size2 tamaño de array
 * @param listaServicios puntero a array de servicios
 * @param size3 tamaño de array
 * @param listaFacturas puntero a array de facturas
 * @param size4 tamaño de array
 * @param i variable de control
 */
void controller_listarUnTrabajoEnEfectivo(eTrabajo* listaTrabajos, int size1,
eBicicleta* listaBicicletas, int size2, eServicio* listaServicios, int size3,
eFactura* listaFacturas, int size4, int i);
```

ENLACE DEL VIDEO EN DRIVE:

https://drive.google.com/file/d/1GfrsrU-HSXw71zc1P22mS_IxoLzIIpe0/view?usp=sharing