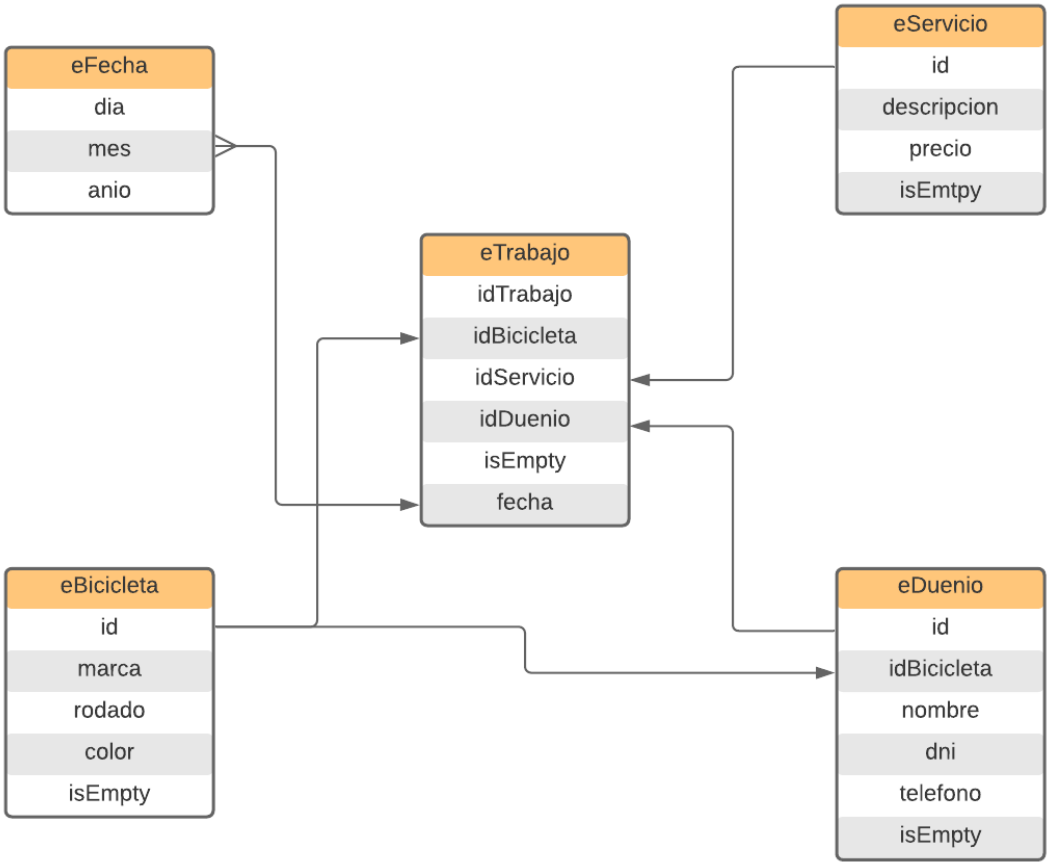
Recuperatorio Primer Parcial
Laboratorio 1

Nombre y Apellido: Mariano Forte
Curso: 1F
DNI: 36785993
N.º Legajo: 110518
E-mail: marianoforte92@gmail.com

Diagrama Entidad-Relación de las Estructuras:



Enunciado de los informes anexados:

K. Listar todos los dueños de las bicicletas.

Mostrar a todas las personas que tienen un trabajo registrado en el sistema.

L. Listar el monto a pagar del servicio por cada dueño.

Mostrar el monto en pesos a pagar por servicio por cada persona, detallando el servicio.
Estructuras relacionadas: eTrabajo, eDuenio, eServicio.

M. Listar todas las bicicletas con sus correspondientes dueños.

Mostrar todas las bicicletas registradas en el sistema y a quién pertenecen.
Estructuras relacionadas: eTrabajo, eDuenio, eBicicleta.

Prototipos de funciones del núcleo del programa:

```
/**
 * @fn int eTrabajo_Alta(eTrabajo*, int, eBicicleta*, int, eServicio*, int, eDuenio*, int, int)
 * @brief Da la alta de un trabajo
 *
 * @param arrayT Array donde lee los datos
 * @param TAM Tamaño del array recibido
 * @param arrayB Array donde lee los datos
 * @param TAMB Tamaño del array recibido
 * @param arrayS Array donde lee los datos
 * @param TAMS Tamaño del array recibido
 * @param arrayD Array donde lee los datos
 * @param TAMD Tamaño del array recibido
 * @param lastId Ultima ID en el sistema
 * @return 0 si fue exitoso, 1 si falló
 */
int eTrabajo_Alta(eTrabajo* array, int TAM, eBicicleta* arrayB, int TAMB, eServicio* arrayS,
int TAMS, eDuenio* arrayD, int TAMD, int lastID);
```

```
/**
 * @fn eTrabajo eTrabajo_CargarDatos(eServicio*, int, eBicicleta*, int, eDuenio*, int)
 * @brief Solicita los datos a agregar en el sistema
 *
 * @param arrayS Array donde lee los datos
 * @param TAMS Tamaño del array recibido
 * @param arrayB Array donde lee los datos
 * @param TAMB Tamaño del array recibido
 * @param arrayD Array donde lee los datos
 * @param TAMD Tamaño del array recibido
 * @return El trabajo a dar de alta
 */
eTrabajo eTrabajo_CargarDatos(eServicio* arrayS, int TAMS, eBicicleta* ArrayB, int
TAMB, eDuenio* arrayD, int TAMD);
```

```
/**
 * @fn int eTrabajo_Modificacion(eTrabajo*, int, eBicicleta*, int, eServicio*, int, eDuenio*,
int)
 * @brief Modifica un trabajo
 *
 * @param arrayT Array donde lee los datos
 * @param TAM Tamaño del array recibido
 * @param arrayB Array donde lee los datos
 * @param TAMB Tamaño del array recibido
 * @param arrayS Array donde lee los datos
 * @param TAMS Tamaño del array recibido
 * @param arrayD Array donde lee los datos
```

```
 * @param TAMD Tamaño del array recibido
 * @return 0 si fue exitoso, 1 si falló
 */
int eTrabajo_Modificacion(eTrabajo* array, int TAM, eBicicleta* bicicleta, int TAMB,
eServicio* arrayS, int TAMS, eDuenio* arrayD, int TAMD);

-----------------------------------------------------------------------------------------------------------------------

/**
 * @fn eTrabajo eTrabajo_CargarDatos(eTrabajo, eServicio*, int, eBicicleta*, int, eDuenio*,
int)
 * @brief Solicita los datos a agregar en el sistema
 *
 * @param trabajo Trabajo al que modificar
 * @param arrayS Array donde lee los datos
 * @param TAMS Tamaño del array recibido
 * @param arrayB Array donde lee los datos
 * @param TAMB Tamaño del array recibido
 * @param arrayD Array donde lee los datos
 * @param TAMD Tamaño del array recibido
 * @return El trabajo a modificar
 */
eTrabajo eTrabajo_ModificarUno(eTrabajo trabajo, eServicio* arrayS, int TAMS,
eBicicleta* arrayB, int TAMB, eDuenio* arrayD, int TAMD);

-----------------------------------------------------------------------------------------------------------------------

/**
 * @fn int eTrabajo_Baja(eTrabajo*, int, eBicicleta*, int, eServicio*, int, eDuenio*, int)
 * @brief Da la baja de un trabajo
 *
 * @param arrayT Array donde lee los datos
 * @param TAM Tamaño del array recibido
 * @param arrayB Array donde lee los datos
 * @param TAMB Tamaño del array recibido
 * @param arrayS Array donde lee los datos
 * @param TAMS Tamaño del array recibido
 * @param arrayD Array donde lee los datos
 * @param TAMD Tamaño del array recibido
 * @return 0 si fue exitoso, 1 si falló
 */
int eTrabajo_Baja(eTrabajo* array, int TAM, eBicicleta* bicicleta, int TAMB, eServicio*
servicio, int TAMS, eDuenio* arrayD, int TAMD);

-----------------------------------------------------------------------------------------------------------------------

/**
 * @fn int indicarMaximoServicios(int*, int)
 * @brief Busca el maximo de los servicios
 *
 * @param array Array donde lee los datos
 * @param TAM Tamaño del array recibido
```

```
 * @return 0 si fue exitoso, 1 si falló
 */
int indicarMaximoServicios(int* array, int TAM);

-----------------------------------------------------------------------------------------------------------------

/**
 * @fn int eServicio_ContarServiciosConMasTrabajos(eTrabajo*, int, eServicio*, int)
 * @brief Cuenta los servicios que tienen mas trabajos a realizar
 *
 * @param arrayT Array donde lee los datos
 * @param TAM Tamaño del array recibido
 * @param arrayS Array donde lee los datos
 * @param TAMS Tamaño del array recibido
 * @return 0 si fue exitoso, 1 si falló
 */
int eServicio_ContarServiciosConMasTrabajos(eTrabajo* arrayT, int TAMT, eServicio*
arrayS, int TAMS);

-----------------------------------------------------------------------------------------------------------------

/**
 * @fn int eTrabajo_CalcularGananciaTotal(eTrabajo*, int, eServicio*, int)
 * @brief Calcula las ganacias totales
 *
 * @param array Array que ordenar
 * @param TAM Tamaño del array recibido
 * @param arrayS Array que ordenar
 * @param TAMB Tamaño del array recibido
 * @return 1 por ordenamiento exitoso, 0 por error
 */
int eTrabajo_CalcularGananciaTotal(eTrabajo* arrayT, int TAMT, eServicio* arrayS, int
TAMS);

-----------------------------------------------------------------------------------------------------------------

/**
 * @fn int eTrabajo_CantidadBicicletasRojo(eTrabajo*, int, eBicicleta*, int)
 * @brief Cuenta la cantidad de bicicletas rojas que hay en el sistema
 *
 * @param arrayT Array donde lee los datos
 * @param TAM Tamaño del array recibido
 * @param arrayB Array donde lee los datos
 * @param TAMB Tamaño del array recibido
 * @return 0 si fue exitoso, 1 si falló
 */
int eTrabajo_CantidadBicicletasRojo(eTrabajo* arrayT, int TAMT, eBicicleta* arrayB, int
TAMB);

-----------------------------------------------------------------------------------------------------------------
```

```
/**
 * @fn int eTrabajo_SortByYearAndBrand(eTrabajo*, int, int, eBicicleta*, int)
 * @brief Ordena el array de datos por ascendente y descendente
 *
 * @param arrayT Array que ordenar
 * @param TAM Tamaño del array recibido
 * @param criterio 1 para ascendente, -1 para descendente
 * @param arrayB Array que ordenar
 * @param TAMB Tamaño del array recibido
 * @return 1 por ordenamiento exitoso, 0 por error
 */
int eTrabajo_SortByYearAndBrand(eTrabajo* arrayT, int TAM, int criterio, eBicicleta*
arrayB, int TAMB);
```

```
/**
 * @fn int eTrabajo_SortByBrand(eTrabajo*, int, int, eBicicleta*, int)
 * @brief Ordena el array de datos por ascendente y descendente
 *
 * @param array Array que ordenar
 * @param TAM Tamaño del array recibido
 * @param criterio 1 para ascendente, -1 para descendente
 * @param arrayB Array que ordenar
 * @param TAMB Tamaño del array recibido
 * @return 1 por ordenamiento exitoso, 0 por error
 */
int eTrabajo_SortByBrand(eTrabajo* arrayT, int TAM, int criterio, eBicicleta* arrayB, int
TAMB);
```

```
/**
 * @fn int eTrabajo_SortByOwnerId(eTrabajo*, int, int, eDuenio*, int)
 * @brief Ordena el array de datos por ascendente y descendente
 *
 * @param arrayT Array que ordenar
 * @param TAM Tamaño del array recibido
 * @param criterio 1 para ascendente, -1 para descendente
 * @param arrayD Array que ordenar
 * @param TAMD Tamaño del array recibido
 * @return 1 por ordenamiento exitoso, 0 por error
 */
int eTrabajo_SortByOwnerId(eTrabajo* arrayT, int TAM, int criterio, eDuenio* arrayD, int
TAMD);
```

Enlace a Google Drive con el video explicativo:

https://drive.google.com/file/d/1EPtdL2UWye9PXnKDNWu1u-_O3JTyKcvl/view?usp=sharing