Recuperatorio 1er parcial CAI

22 de Junio de 2021

# Enunciado

La plataforma E-Mobility quiere fortalecer su posición en nuestro país con su nuevo servicio de micro movilidad urbana a través de los distintos puntos de alquiler inteligentes que se encuentran distribuidos en toda la ciudad capital (inicialmente).

Para hacer esto, necesita trabajar en su sistema de alquiler de equipos eléctricos, que debe ser capaz de:

1) Contar con las entidades EntidadInteligente, PuntoAlquiler, Alquiler y EquipoMovil (que bien puede ser Bicicleta, Monopatín o Tabla (hoverboard)).

1. ToString se debe sobrescribir sólo en EntidadInteligente y en Alquiler.
2. Encapsular los atributos necesarios aplicando sólo los getters y setters (en los casos que corresponda)
3. Una EntidadInteligente es igual a otra si su NroSerie es el mismo
4. Se valorará el uso de constructores sobrecargados

2) Poder alquilar equipos por DNI del cliente (AlquilarEquipo)

*El cliente ingresa su DNI, se le muestra la lista de equipos. Selecciona un equipo de la lista, ingresa las horas y el punto de venta inteligente libera el equipo (que se encuentra conectado por un candado).*

1. En caso que el DNI ya tenga un alquiler sin devolver, se debe arrojar una excepción de tipo AlquilerEnCursoException.
2. Si el equipo está actualmente en alquiler se debe arrojar EquipoAlquiladoException.
3. Solo se pueden alquilar equipos que tengan más del 50% de la batería en caso que sea bicicleta o monopatín, 70% si es una tabla.
4. Se debe autogenerar el número de alquiler.

3) Poder listar los alquileres con el {FORMATO}. (ListarAlquileres)

1. En caso que no haya alquileres mostrar “No hay alquileres”.

4) Poder devolver un equipo alquilado por DNI. (DevolverEquipo)

*El cliente ingresa su DNI y con cuánta batería lo devuelve.*

Siempre debe validarse que exista un alquiler para ese DNI SIN devolver, sino, debe lanzarse la excepción AlquilerNoExistenteException.

1. -PUNTO OPCIONAL- Si la batería ingresada es mayor a la que el equipo disponía, se le resta 10% al precio del alquiler.

Formatos para ToString (display)

1. PuntoAlquiler “Punto {NroSerie}”
2. EquipoMovil “{Tipo} {NoSerie} ) $ {PrecioPorHora} ({Batería}%) - {Observación}”
3. Alquiler "{NroAlquiler} {EquipoMovil.ToString() } lo alquió {DNI} y [estó/NO está] devuelto - Importe “{MontoTotal}"

# Consideraciones generales

Se deben respetar las visibilidades del diagrama

* Público - Privado # Protegido = Internal

// En negrita son propiedades que encapsulan el atributo del mismo nombre (determinar cuando va solo get o get y set)

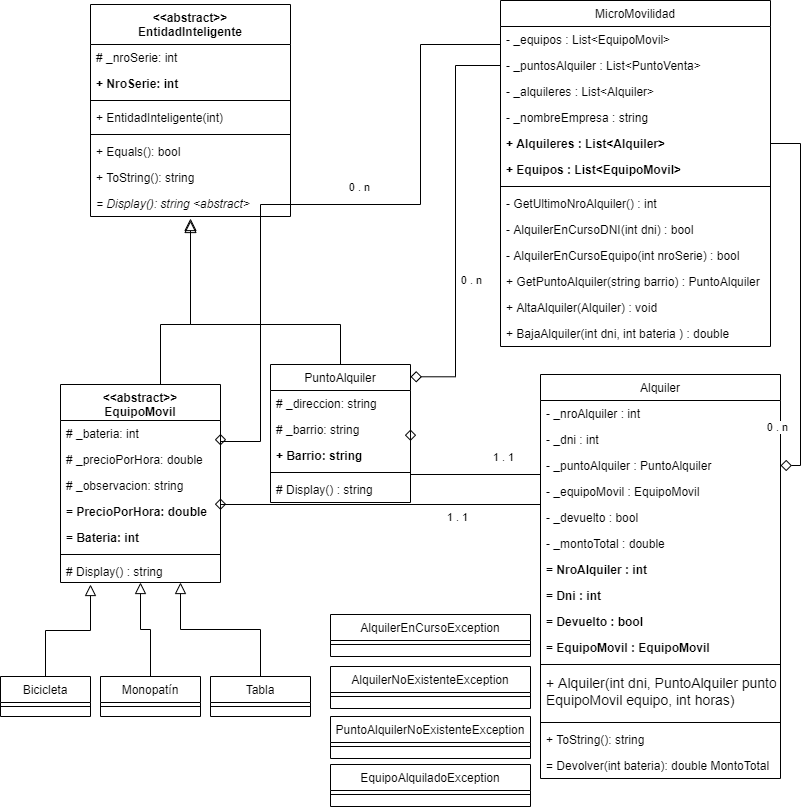
// Se puede saber el tipo de la instancia con .GetType().Name

* El proyecto principal deber ser de consola
* Se debe desarrollar utilizando al menos dos capas
* Deben realizarse validación de las entradas de usuario
* Manejo apropiado de excepciones
* La consola se debe mantener activa hasta que el usuario finalice el programa ingresando X
* Se debe resolver en menos de 2 horas
* Debe compilar sin errores
* No debe terminar por excepción no controlada
* El prefijo del nombre de los proyectos debe ser Recuperatorio1.APELLIDO.NombreProyecto
* Se entrega la solución entera (comprimida) en la dirección de drive previamente asignada
* Es válido agregar los constructores que sean necesarios

# 

# 

# Modelo



# Template de la clase Program (para completar)

public class Program

{

private static MicroMovilidad \_empresa;

static Program()

{

\_empresa = new MicroMovilidad(“E-Mobility”);

}

static void Main(string[] args)

{

PuntoAlquiler punto = \_empresa.GetPuntoAlquiler(“Belgrano”);

// sacar por pantalla >> Bienvenido [nombreEmpresa]

// sacar por pantalla >> Punto de venta del barrio [barrio]

DesplegarOpcionesMenu();

string opcionMenu = ""; // pedir el valor

switch (opcionMenu)

{

case "1":

ListarAlquileres();

break;

case "2":

AlquilarEquipo(punto);

break;

case "3":

DevolverEquipo();

break;

case "X":

// SALIR

break;

default:

break;

}

}

static void DesplegarOpcionesMenu()

{

Console.WriteLine("1) Listar Alquileres");

Console.WriteLine("2) Alquilar Equipo");

Console.WriteLine("3) Devolver Equipo");

Console.WriteLine("X: Terminar");

}

static void ListarAlquileres()

{

// mostrar alquileres

}

static void AlquilarEquipo(PuntoAlquiler punto)

{

// Pedir DNI cliente

// Listar equipos

// Pedir NroSerie equipo

// Pedir Horas

// Con el Nro de serie traer el equipo de la lista

// Generar Alquier con dni, punto, equipo, horas

// Alta Alquiler

}

static void DevolverEquipo()

{

// Ingreso DNI cliente

// Ingreso batería

// BajaAlquier

// Se le muestra el monto total a debitar en formato .ToString(“0.00”)

}

}

## Datos de Inicialización

// iniciar MicroMovilidadcon los siguientes datos

public MicroMovilidad()

{

// incializar listas

\_equipos.Add(new Bicicleta(5001, 90, 15, "Rodado 26"));

\_equipos.Add(new Bicicleta(5002, 25, 15, "Rodado 24"));

\_equipos.Add(new Monopatin(4003, 75, 18, "Mod. 2020"));

\_equipos.Add(new Monopatin(4004, 49, 12, "Batería dura poco"));

\_equipos.Add(new Monopatin(4040, 63, 15, "Sin guardabarro trasero"));

\_equipos.Add(new Tabla(3041, 79, 12, "Con luces"));

\_equipos.Add(new Tabla(3042, 65, 11, "Sin luces"));

\_puntosAlquiler.Add(new PuntoAlquiler(100013, "Roosevelt 1852","Belgrano"));

\_puntosAlquiler.Add(new PuntoAlquiler(100014, "Soler 3055", "Palermo"));

}