Profesores: *Villegas Octavio*

Parte POO

Aplicación Nº 1 (PRODUCTO-CONTEINER)

los conteiner pueden ser (chico, mediano o grande) con una capacidad de 1000kg o 2500kg o 9000 kg

los productos tiene un identificador único de producto, el nombre, el importador , el país de origen y los kilos.

si el producto ya existe se suma los kilos.

Aplicación Nº 2 (Auto-estacionamiento)

. Realizar una clase llamada **"Auto"** que posea los siguientes atributos **privados**:
_color (String)
_precio (Double)
_marca (String).
_fecha (DateTime)
Realizar un constructor capaz de poder instanciar objetos pasándole todos los parámetros:
Realizar un método de **instancia** llamado "**AgregarImpuestos",** que recibirá un doble por parámetro y que se sumará al precio del objeto.
Realizar un método de **clase** llamado **"MostrarAuto",** que recibirá un objeto de tipo **"Auto"** por parámetro y que mostrará todos los atributos de dicho objeto.
Crear el método de instancia **"Equals"** que permita comparar dos objetos de tipo **"Auto"**. Sólo devolverá *TRUE* si ambos **"Autos"** son de la misma marca.
Crear un método de clase, llamado **"Add"** que permita sumar dos objetos **"Auto"** (sólo si son de la misma marca, y del mismo color, de lo contrario informarlo) y que retorne un **Double** con la suma de los precios o cero si no se pudo realizar la operación.
Ejemplo: $importeDouble = Auto::Add($autoUno, $autoDos);
En *testAuto.php*:

- Crear **dos** objetos **"Auto"** de la misma marca y distinto color.
- Crear **dos** objetos **"Auto"** de la misma marca, mismo color y distinto precio.
- Crear **un** objeto **"Auto"** utilizando la sobrecarga restante.
- Utilizar el método **"AgregarImpuesto"** en los últimos tres objetos, agregando $ 1500 al atributo precio.

- Obtener el importe sumado del primer objeto **"Auto"** más el segundo y mostrar el resultado obtenido.
- Comparar el primer **"Auto"** con el segundo y quinto objeto e informar si son iguales o no.
- Utilizar el método de clase "**MostrarAuto**" para mostrar cada los objetos impares (1, 3, 5)

Crear la clase **Garage** que posea como atributos privados:
_razonSocial (String)
_precioPorHora (Double)
_autos (Autos[], reutilizar la clase Auto del ejercicio anterior)
Realizar un constructor capaz de poder instanciar objetos pasándole como parámetros:
i. La razón social.
ii. La razón social, y el precio por hora.
Realizar un método de **instancia** llamado **"MostrarGarage",** que no recibirá parámetros y que mostrará todos los atributos del objeto.
Crear el método de instancia **"Equals"** que permita comparar al objeto de tipo **Garaje** con un objeto de tipo **Auto.** Sólo devolverá *TRUE* si el auto está en el garaje.
Crear el método de instancia **"Add"** para que permita sumar un objeto **"Auto"** al **"Garage"** (sólo si el auto **no** está en el garaje, de lo contrario informarlo).
Ejemplo: $miGarage->Add($autoUno);
Crear el método de instancia **"Remove"** para que permita quitar un objeto **"Auto"** del **"Garage"** (sólo si el auto **está** en el garaje, de lo contrario informarlo).
Ejemplo: $miGarage->Remove($autoUno);
En *testGarage.php*, crear autos y un garage. Probar el buen funcionamiento de todos los métodos.

Aplicación Nº 3 (pasajero-vuelo)

Dadas las siguientes clases:
Pasajero

Atributos **privados**: _apellido (string), _nombre (string), _dni (string), _esPlus (boolean)
Crear un constructor capaz de recibir los cuatro parámetros.
Crear el método de instancia **"Equals"** que permita comparar dos objetos Pasajero. Retornará *TRUE* cuando los _dni sean iguales.
Agregar un método getter llamado *GetInfoPasajero*, que retornará una cadena de caracteres con los atributos concatenados del objeto.
Agregar un método de clase llamado *MostrarPasajero* que mostrará los atributos en la página.
Vuelo

Atributos **privados**: _fecha (DateTime), _empresa (string) _precio (double), _listaDePasajeros (array de tipo Pasajero), _cantMaxima (int; con su getter). Tanto _listaDePasajero como _cantMaxima sólo se inicializarán en el constructor.
Crear el constructor capaz de que de poder instanciar objetos pasándole como parámetros:
i. La empresa y el precio.
ii. La empresa, el precio y la cantidad máxima de pasajeros.

Agregar un método getter, que devuelva en una cadena de caracteres toda la información de un vuelo: fecha, empresa, precio, cantidad máxima de pasajeros, y toda la información de **todos** los pasajeros.
Crear un método de **instancia** llamado AgregarPasajero, en el caso que no exista en la lista, se agregará (utilizar Equals). Además tener en cuenta la capacidad del vuelo. El valor de retorno de este método indicará si se agregó o no.
Agregar un método de instancia llamado MostrarVuelo, que mostrará la información de un vuelo.
Crear el método de clase **"Add"** para que permita sumar dos vuelos. El valor devuelto deberá ser de tipo numérico, y representará el valor recaudado por los vuelos. Tener en cuenta que si un pasajero es **Plus**, se le hará un descuento del 20% en el precio del vuelo.
Crear el método de clase **"Remove"**, que permite quitar un pasajero de un vuelo, siempre y cuando el pasajero esté en dicho vuelo, caso contrario, informarlo. El método retornará un objeto de tipo Vuelo.