

Profesores: *Neiner Maximiliano, Villegas Octavio*

Parte 3 - Ejercicios con Funciones

Aplicación N° 12 (Invertir palabra)

Realizar el desarrollo de una función que reciba un Array de caracteres y que invierta el orden de las letras del Array.

Ejemplo: Se recibe la palabra "HOLA" y luego queda "ALOH".

Aplicación N° 13 (Invertir palabra)

Crear una función que reciba como parámetro un string (**\$palabra**) y un entero (**\$max**). La función validará que la cantidad de caracteres que tiene **\$palabra** no supere a **\$max** y además deberá determinar si ese valor se encuentra dentro del siguiente listado de palabras válidas: "Recuperatorio", "Parcial" y "Programacion". Los valores de retorno serán:

1 si la palabra pertenece a algún elemento del listado.

0 en caso contrario.

Parte 4 - Ejercicios con POO

Aplicación N° 17 (Auto)

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 como parámetros:

- i. La marca y el color.
- ii. La marca, color y el precio.
- iii. La marca, color, precio y fecha.

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)

Aplicación Nº 18 (Auto - Garage)

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.