Boletín de ejercicios Tema 5

1. **Uso de propiedades y métodos estáticos**

   Modifica la clase Fecha, para que la propiedad calendario no sea accesible desde lo exterior de la clase. Debes añadir los siguientes métodos estáticos:

   - getCalendario: que devolverá el valor de esta propiedad
   - getHora: que había devuelto la hora con el siguiente formato HH:MM:SS
   - getDataHora: que llamará a los métodos getData y getHora para mostrar tanto la fecha como la hora

   La salida que debe mostrar es la siguiente:

```
Usamos el calendario: Calendario gregoriano
Hoy es Jueves 21 de Enero del 2021 y son las 09:48:31
```

   La clase de la que partimos es la que se muestra a continuación, y deberemos hacer un ejemplo de su uso llamando a los métodos getCalendario y getDataHora.

```php
class Data {
    public static $calendario = "Calendario gregoriano";
    public static function getData(){
        $ano = date('Y');
        $mes = date('m');
        $dia = date('d');
        return $dia.'/'.$mes.'/'.$ano;
    }
}
```

2. Definición de constructores y métodos.

Crear una clase Persona con tres propiedades (nombre, fecha de nacimiento y sexo) que no deben ser accesibles directamente desde el exterior.

- Define un constructor que puede recibir como parámetro las 3 propiedades o únicamente las dos primeras. En cuyo caso el valor que debe tomar el atributo sexo es ' H'.
- Al mostrar el sexo debe devolver 'hombre' si el valor de sexo es ' H', 'mujer' si el valor de sexo es ' M', es desconocido para cualquier otro valor.
- La clase debe incluir un método de nombre diasVivo que devolverá información relativa a los días que pasaron desde la data de nacimiento de la persona, empleando la clase DateTime de PHP, de la forma: 5 años, 3 meses, 14 días, un total de 1932 días

  Un ejemplo de la salida podría ser:

```
Pedro tiene 5 años, 3 meses, 25 días, un total de 1943 días.
Su sexo es: Hombre
```

3. Uso de métodos mágicos

Crear una clase Articulo con dos propiedades (id y nombre) que no deben ser accesibles directamente desde el exterior.

- Define un constructor que establece el valor de las propiedades.
- Cuando se clone un objeto el id debe incrementarse en 1 con respeto al objeto original.
- Utiliza métodos mágicos para establecer y obtener los valores de los atributos de manera que sigan siendo válidos en caso de que añadiéramos más atributos a la clase.

- En caso de que intentemos mostrar el objeto con echo o print debe llamarse a un método de nombre mostrarArticulo que dé la información de los valores de los atributos de la forma:

  1 – linterna.

  2 – lámpara.

4. Implementación de herencia en PHP

Queremos gestionar una academia de baile. Para eso, tenemos que guardar información tanto de los alumnos como de los profesores que imparten clases en la academia. De ambos queremos saber su nombre, apellidos, móvil. De los profesores además queremos almacenar el NIF para lo cual llamarán al método constructor de Persona además de almacenar el NIF.

Tenemos que declarar las siguientes clases:

- La clase Persona debe tener un método *verInformación* que devuelve para la información con el siguiente formato: Uxia Loureiro Agra (699444999)
- La clase Alumno tiene dos métodos: *setNumClases* y *aPagar*, y debe emplear el método constructor de Persona.
  - El método *aPagar* devolverá el importe y pagan en función del número de actividades en las que se inscriben:
    - Por una actividad: 20 euros
    - Por dos actividades:32 euros
    - Por tres o más: 40 euros.
    - En caso de que no esté establecido el número de clases a las que asiste para ese alumno devolverá 'Debe indicar previamente el número de clases'.
  - La clase Profesor tiene un método *calcularSueldo* que calcula lo que cobran los profesores dependiendo del número de clases que imparten al mes. Recibe como parámetros el número de horas y

el importe de cada hora, que está establecido en 16 euros, pero podría variar.

- La clase Baile con dos atributos: nombre y edadMinima. La edad mínima será de 8 años salvo que se indique el contrario.
  El profesor tendrá 3 métodos para añadir los Bailes que imparte, eliminar un baile cuando deje de impartirlo y para devolver los bailes que imparte de la forma:

```
HIP HOP (edad min:8 años)
```

  Antes de añadir un baile debe comprobar si ya está dado de alta para ese profesor.

- La clase Academia: almacenará su nombre en una constante y debe permitir añadir Profesores y Alumnos. Para probarlo debes hacer el siguiente:
  - Añade a la academia un profesor que imparte 4 bailes (entre ellos AFRO, y uno de ellos duplicado) y 2 alumnos.
  - Muestra información del profesor (incluyendo el sueldo y los bailes que imparte) y de los alumnos incluyendo la cuota que deberá pagar.
  - El profesor deja de dar clase de AFRO. Actualiza la información de la academia y vuelve a mostrar la información del profesor.
  - Impide la herencia de las clases Alumno y Profesor.

5. Definición y uso de métodos y clases abstractas

Define una clase abstracta de nombre Calculo que tenga como atributos $operando1, $operando2 y $resultado y que defina los métodos setOperando1, setOperando2, getResultado y un método abstracto calcular. A continuación,

define tres subclases de esta clase que tienen como objetivo realizar las operaciones de suma, resta y multiplicación.

- Antes de realizar la operación debes comprobar que los operandos tienen algún valor.
- Las clases y subclases que crees deben estar en una carpeta de nombre clases. En el script donde compruebes el funcionamiento de estos cálculos debes hacer que se carguen automáticamente todas las clases que se encuentren en esa carpeta.

6. **Crea una clase Empleado con:**
    - Atributos: su nombre y sueldo.
    - Definir un método de inicialización al que lleguen como dato el nombre y sueldo.
    - Plantear un segundo método que imprima el nombre y un mensaje si debe o no pagar impuestos (si el sueldo supera a 3000 paga impuestos)
    - **Ponlo en funcionamiento**: Crea dos empleados, uno que paque impuestos y otro que no.

7. **Crea una clase Menu con:**
    - Atributos: arrays con los enlaces web y sus títulos. Ejemplo: www.marca.com y Marca
    - Métodos para cargar los enlaces, y para mostrar los menús horizontal o verticalmente.
    - **Ponlo en funcionamiento**: Crea un Menu con tres enlaces, y muéstralos de las dos formas (horizontal y vertical).

8. **Crea una clase Racional que:**
    - Podamos inicializar con un string del tipo racional. Por ejemplo "8/5"

- **Ponlo en funcionamiento**: Crea dos valores racionales y muéstralos por pantalla.

9. Crea una clase Factura:

- Constantes: IVA
- Atributos: Importe Base, fecha, impuestos, Importe bruto, estado (pagada o pendiente)
- Métodos: imprime
- **Ponlo en funcionamiento**: Crea varias facturas, de diferentes fechas, con diferentes impuestos e importes brutos y estados.

10. Sobrecargar el constructor de Racional

- Siguiendo el ejercicio 8 establecido anteriormente realiza un constructor que permita instanciar un objeto de la clase racional de la siguiente manera

11. Crea una clase Persona con:

- Un atributo donde se almacene su nombre.
- Dos métodos, uno que cargue el nombre y otro que lo imprima.

12. Crea una clase CabeceraPagina que permita:

- Mostrar un título, indicarle si queremos que aparezca centrado, a derecha o izquierda.

13. Crea una clase CabeceraPagina que permita:

- Mostrar un título, indicarle si queremos que aparezca centrado, a derecha o izquierda.
- Utilizar un constructor para inicializar los dos atributos (titulo y posición)

14. Crea una clase Tabla que permita:

- Indicarle en el constructor la cantidad de filas y columnas.
- Definir un método que nos permita cargar un dato en una determinada fila y columna.
- Mostrar los datos en una tabla HTML

15. Crea una clase Tabla que permita:

- Indicarle en el constructor la cantidad de filas y columnas.
- Definir un método que nos permita cargar un dato en una determinada fila y columna.
- Mostrar los datos en una tabla HTML
- Definir los modificadores de acceso (public y private) para atributos y métodos

16. Crea una clase Pagina que:

- Contenga como atributos objetos de las clases Cabecera, Cuerpo y Pie.
- La clase Cabecera y Pie deben tener un atributo donde almacenar el texto a mostrar.
- La clase Cuerpo debe tener un atributo de tipo vector donde se almacenen todos los párrafos.

17. Crea una clase Opcion y una clase Menu:

- La clase Opcion definirá como atributos el titulo, enlace y color de fondo, los métodos a implementar serán el constructor y el graficar.
- Por otro lado, la clase Menú administrará un array de objetos de la clase Opcion e implementará un método para insertar objetos de la clase Menu y otro para graficar. Al constructor de la clase Menu indicarle si queremos el menú en forma 'horizontal' o 'vertical'.

18. Crea una clase CabeceraPagina que permita:

- Mostrar un título alineado con un determinado color de fuente y fondo.
- Definir en el constructor parámetros opcionales para los colores de fuente, fondo y el alineado del título.

19. Crea una clase Operacion que permita:

- Definir como atributos $valor1, $valor2, $resultado
- Definir como métodos cargar1 (inicializa el atributo $valor1), cargar2 (inicializa el atributo $valor2) y por último un método que muestre el contenido de $resultado.
- Definir dos subclases de la clase Operacion.
    - Suma: que tiene por objetivo la carga de dos valores, sumarlos y mostrar el resultado.
    - Resta: que tiene por objetivo la carga de dos valores, restarlos y mostrar el resultado de la diferencia.

20. Crea una clase Operacion que permita:

- Definir como atributos $valor1, $valor2, $resultado
- Definir como métodos cargar1 (inicializa el atributo $valor1), cargar2 (inicializa el atributo $valor2) y por último un método que muestre el contenido de $resultado.
- Definir la subclase Suma, que tiene por objetivo la carga de dos valores, sumarlos y mostrar el resultado.
- Tratar de asignarle el valor 10 al atributo $valor1 en donde definimos un objeto.

21. Herencia. Crearemos una clase Admin, que es una clase secundaria de la clase Usuario.

- Agrega a la clase una propiedad privada $username
- Crea un método para establecer el valor del nombre de usuario
- Crea una clase Admin que herede la clase Usuario
- Agrega a la clase Admin un método público con el nombre de muestraTuRol() y haga que devuelva la cadena con el nombre de la clase: "Admin" .
- Agrega a la clase Admin otro método público, saluda() , que devuelve la cadena "Hola Admin, XXX" con el username en lugar de XXX.
- Cree un objeto Admin $admin1, con el username "Ramon" y salude al usuario. ¿Ves algún problema?
- Cambia el código para solucionar el problema.
- Escribe la solución con un método getter dentro del padre que se pueda usar desde la clase secundaria.

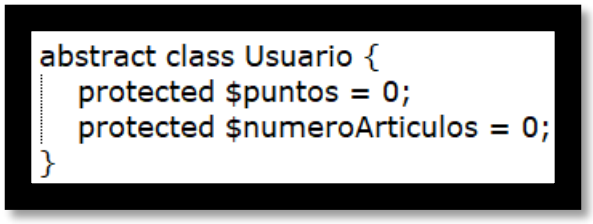
22. **Clases abstractas. Crearemos una clase Usuario abstracta y dos clases secundarias (clases Admin y Viewer) que heredan de la clase abstracta:**
    - Crea una clase abstracta llamada Usuario, que tiene un método abstracto con el nombre de estableceRol() .
    - Agrega a la clase una variable protected $username y los métodos setter y getter públicos para establecer y obtener el nombre de usuario.
    - Crea una clase *Admin* que herede la clase *Usuario* abstracta.
    - Defina el método estableceRol() en la clase secundaria y deje que devuelva la cadena "Admin" ;
    - Cree otra clase, *Viewer*, que herede la clase abstracta *Usuario*. Defina el método que debe definirse en cada clase secundaria de la clase *Usuario*.
    - Cree un objeto desde la clase Admin, establezca el nombre de usuario en "Ramon" y haga que devuelva la cadena "Admin" .

23. **Interfaces. Permitiremos que la misma clase secundaria herede tanto de una clase primaria como de dos interfaces.**

- Crea una clase Usuario con una propiedad protegida $username y métodos que puedan establecer y obtener el $username .
- Crea una interfaz Autor con los siguientes métodos abstractos que pueden proporcionar al usuario una serie de privilegios de autoría:
    - setPrivilegiosAutor() , obtiene un parámetro de $array
    - getPrivilegiosAutor()
- Crea una interfaz Editor con métodos para establecer y obtener los privilegios del editor.
- Crea una clase AutorEditor que amplíe la clase Usuario e implemente las interfaces Autor y Editor.
- Crea en la clase AutorEditor los métodos que debe implementar y las propiedades que estos métodos nos obligan a agregar a la clase. Por ejemplo, para implementar el método público setPrivilegiosAutor() , debemos agregar a nuestra clase una propiedad que contenga la matriz de privilegios de autoría y denominarla $privilegiosAutor en consecuencia.
- Ahora, creemos un objeto con el nombre $user1 de la clase AutorEditor , y establezcamos su nombre de usuario en "Ramon" .
- Establece en el objeto $user1 una serie de privilegios de autoría, con los siguientes privilegios: "escribir texto", "agregar puntuación" .
- Establece en el objeto $user1 una matriz con los siguientes privilegios editoriales: "editar texto", "editar puntuación".
- Muestra el nombre y los privilegios de $user1

24. Polimorfismo. Crear una clase de usuario abstracta que cuente las clases que heredan de ella para calcular la cantidad de puntos que tiene un usuario en función de la cantidad de artículos que ha creado o editado. Sobre la base de la clase *Usuario*, vamos a crear las clases *Autor* y *Editor*, y ambas calcularán el número de puntos con el método calculaPuntos().

- La clase abstracta *Usuario* tendrá este esqueleto:



```
abstract class Usuario {
    protected $puntos = 0;
    protected $numeroArticulos = 0;
}
```

- Agrega a la clase Usuario métodos concretos para establecer y obtener el número de artículos
- Agrega a la clase el método abstracto: calculaPuntos(), que realiza los cálculos de puntos por separado para cada clase.
- Crea una clase de autor que herede de la clase de usuario. En el Autor, cree un método concreto calculaPuntos() que devuelva el número de puntajes del siguiente cálculo: **numeroArticulos * 10 + 20**
- También crea una clase Editor que herede de la clase Usuario. En el Editor, cree un método concreto calculaPuntos() que devuelva el número de puntajes del siguiente cálculo: **numeroArticulos * 5 + 10**
- Crea un objeto, $autor1, de la clase Autor, establece el número de artículos en 5 y muestra los puntos que obtuvo el autor.
- Cree otro objeto, $editor1, de la clase Editor, establece el número de artículos en 10 y muestra los puntos que obtuvo el editor.