**Ejercicios Objetos**[**¶**](https://aitor-medrano.github.io/dwes2122/03phpoo.html#objetos)

**Realiza los ejercicios y pon tu solución en el foro de la unidad. Elegiremos la mejor solución de entre las 3 primeras**

**Ejercicio 1** **Empleado.php**

Crea una clase **Empleado** con su nombre, apellidos, sueldo y teléfono (almacena un array de números de teléfonos) .

Encapsula las propiedades mediante *getters/setters* para sueldo y teléfono.

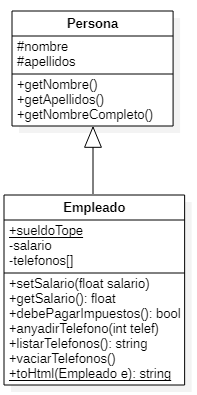
Crea constructor que asignará valor a nombre y apellidos, si recibe un tercer parámetro, será el sueldo del Empleado. Si no, se le asignará 1000€ como sueldo inicial.

Añade métodos para:

* + Obtener su nombre completo → getNombreCompleto(): string
  + Que devuelva un booleano indicando si debe o no pagar impuestos (se pagan cuando el sueldo es superior a 3333€) → debePagarImpuestos(): bool
  + Añade un teléfono al array → public function anyadirTelefono(int $telefono) : void
  + Muestra los teléfonos separados por comas → public function listarTelefonos(): string
  + Elimina todos los teléfonos → public function vaciarTelefonos(): void

Añade una variable estática sueldoTope, de manera que mediante *getter/setter* puedas modificar su valor.

**Ejercicio 1\_b Persona.php EmpleadoHijo.php**



Crea una clase Persona que sea padre de Empleado, de manera que Persona contenga el nombre y los apellidos, y en Empleado quede el salario y los teléfonos.

Añade en Persona un atributo edad.

A la hora de saber si un empleado debe pagar impuestos, lo hará siempre y cuando tenga más de 21 años y dependa del valor de su sueldo. Modifica todo el código necesario para mostrar y/o editar la edad cuando sea necesario.

Copia la clase del ejercicio anterior y modifícala para adecuarse  
  
**Ejercicio2**

Desarrolla una clase segura para la gestión de sesiones en PHP, integrando buenas prácticas de seguridad web y considerando aspectos clave de protección contra ataques comunes.

Puedes incluir más requisitos para aumentar la funcionalidad o seguridad.

Propiedades:

* *$session\_name (permite darle un nombre a la cookie de sesión diferente al estándar).* Le daremos un valor por defecto.
* *$tiempo\_inactividad* Le daremos un valor por defecto

Requisitos propuestos:

1. **Constructor:**
   * Inicia sesión.
   * Puede recibir tiempo de inactividad y nombre de sesión.
   * Comprueba inactividad y time en caso de ser necesario (cuando sea usuario logueado).
2. **Cierre de Sesión Seguros:**
   * Al cerrar sesión, asegurarse de eliminar adecuadamente todas las variables de sesión.
3. **Inicio sesión usuarios logueados:**
   * Al iniciar sesión, regenerar identificador de sesión.
   * Incluir información relevante como la dirección IP del usuario y la fecha y hora de la actividad.
   * Almacenar información relevante en $\_SESSION. Se recibirá como array asociativo (userName, nivelUsuario, ImagenPerfil,…)
4. **Control inactividad:**
   * En cada interacción del usuario logueado con la aplicación.
   * Implementar una función para controlar tiempo de inactividad y renovar el tiempo de vida de la sesión.
5. **Control de IP:**
   * Realizar una comprobación de la dirección IP del usuario en cada solicitud para asegurar que la sesión esté asociada con la misma IP que la sesión original.
6. **Modificary leer Array $\_SESSION:**
   * Implementar métodos adicionales para modificar y leer el array $\_SESSION