Ejercicios Avanzados PHP

1. Crear una página web que pida NIE. Después, crea otra página que lo valide de la siguiente forma:
   - El DNI tiene que cumplir las siguientes reglas de los DNI españoles: 8 números y una letra. Pero se puede especificar también un NIE en cuyo caso consta de una letra (solo puede ser X, Y o Z) y 7 números más una letra final. La letra final del DNI cumple esta fórmula: los números se dividen entre 23 y se toma el resto. El resto se sustituye por una letra siguiendo el siguiente patrón:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
T	R	W	A	G	M	Y	F	P	D	X	B	N	J	Z	S	Q	V	H	L	C	K	E

   - En el caso de los NIE para calcular la letra final se hace lo mismo, pero sustituyendo la letra inicial por 0 si es una X, por 1 si es una Y y por 2 si es una Z.
   - Deberá mostrarse si el NIE introducido es correcto o no.

   Nota: Investiga la función preg_match e intenta utilizar expresiones regulares para la validación

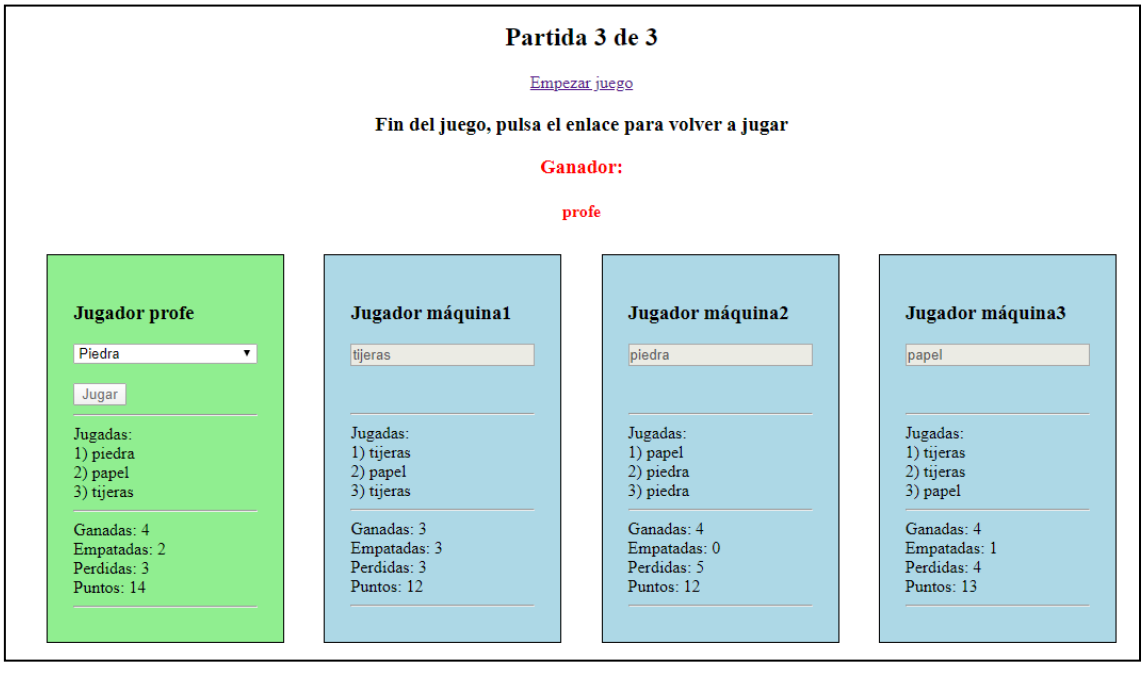
2. Escribe un programa PHP que permita al usuario introducir tres números
   - Dos de ellos serán los límites inferior y superior de un rango, el tercero será un número situado dentro de dicho rango
   - Cuando el programa reciba los datos generará un número aleatorio entre los límites inferior y superior y lo comparará con el tercer valor.
   - Finalmente se informará al usuario si ha acertado el número aleatorio o no y cuantos intentos lleva. En caso de fallo, la aplicación le permitirá realizar un nuevo intento con los límites introducidos al principio (es decir podrá intentar adivinar de nuevo el número aleatorio que se mantiene mientras no se introduzcan nuevos límites).
   - Debe haber un enlace para comenzar una nueva partida.

3. Partiendo del ejercicio de piedra, papel o tijeras que se entregó anteriormente, ampliarlo/modificarlo haciendo uso de sesiones y POO.
   Los requisitos de la aplicación para el juego son los siguientes:

   - Al inicio de una partida se pida mediante formulario al usuario:
     - Su nombre
     - Número de contrincantes (entre 1 y 5) : se refiere a ordenadores contra los que va a jugar.
     - Número de jugadas que va a tener una partida completa (entre 3 y 5)
   - Al pulsar el botón del formulario se accederá a la página del juego en cuya parte superior aparecerá el texto partida x de N siendo x el número de partida (o jugada) actual, y N el número total de partidas.
   - Debajo un enlace a la pantalla inicial del juego.

- Debajo se generarán tantos “tableros” como participantes haya. Cada uno de ellos contendrá una cabecera “jugador: xx” siendo xx el nombre del jugador. En el caso de los ordenadores su nombre será máquina1, másquina2, etc.
- El color de fondo del jugador “persona” será diferente del de las máquinas.
- El jugador persona podrá elegir la ficha que desea jugar: piedra, papel o tijera, y mediante un botón (jugar) enviará la jugada al servidor.
- La “tirada” del jugador se comprobará con la de los ordenadores (que se han generado aleatoriamente).
- Los datos que se deben mostrar respecto a las partidas son:
    - La jugada que ha hecho cada uno (las de los ordenadores solo se mostrarán cuando el jugador haya pulsado el botón de jugar)
    - Nº de jugadas ganadas, perdidas y empatadas , para ello cada jugada se compara con el resto de los oponentes, si por ejemplo el jugador “persona” ha seleccionado la ficha “piedra”, y la máquina 1 ha jugado tijeras, y la máquina 2 “papel”, persona suma 1 a ganadas y 1 a perdidas (gana a máquina 1 y pierde con máquina 2). Máquina 1 pierde con persona y gana a máquina 2 y máquina 2 gana a persona y pierde con máquina 1.
    - También deben mostrarse los puntos totales de la partida que se calculan de la siguiente manera:
        - Cada jugada ganada cuenta 3 puntos.
        - Cada jugadas empatada cuenta 1 punto.
        - Las jugadas perdidas no suman ni restan.
- Al finalizar las partidas se mostrará en la parte superior de la pantalla el mensaje “Fin del juego, pulsa el enlace para volver a jugar” y en color resaltado el nombre del ganador (el participante con más puntos). Si hay varios jugadores con el mismo número de puntos se mostrarán los nombres de todos ellos.

Ejemplo de visualización del juego:



4. Cookies: preferencias de usuario
    - El usuario debe seleccionar en un formulario sus preferencias en cuanto a color de fondo, color de letra, y tipografía (entre tres opciones). Además debe introducir el nombre y los apellidos.
    - Al enviar el formulario se redirige a una página en la que se le saluda por sus nombre y que está configurada con los colores y tipo de letra elegido. Las opciones seleccionadas se guardarán en cookies, de forma que si se vuelve a ejecutar el script principal, si se detecta que hay cookies, se redirige automáticamente a esta segunda página, configurada con las selecciones realizadas.
    - La segunda página contiene un enlace que al pulsarlo redirige a la primera página, borrando además las cookies con las opciones elegidas.

5) Trabajar con clases: Protectora de animales POO, BBDD

Se proporcionará una base de datos en el aula virtual que os podéis descargar.

Crear las siguientes clases:

- **Clase conexión.php:** se encargará de realizar la conexión con la BD.
    - Tiene como atributos privados los parámetros de la conexión a la BD.
    - Tienen un único método protegido que realiza la conexión y devuelve el manejador de la conexión o si hay error muestra un mensaje y finaliza el programa.
- **Clase abstracta Crud.php:** va a contener métodos que realizarán funciones básicas con la BD (Create, Update, Read y Delete)
    - Hereda de la clase Conexión
    - Tiene dos atributos privados, tabla (contendrá el nombre de la tabla de la base de datos con la que se va a operar, y conexión contendrá el manejador de la conexión a la BD.
    - Métodos públicos
        - Constructor: le llega como parámetro el nombre de la tabla que se asignará a la propiedad privada tabla y asignará a la propiedad conexión el resultado de invocar al método de la clase padre que realiza la conexión a la base de datos.
        - obtieneTodos: hace una consulta preparada en la que devuelve todos los registros de la tabla que está asignada a la propiedad tabla, como un objeto (usar FETCH_OBJ).
        - obtieneDeID: le llega como parámetro un id. Devuelve el resultado de una consulta preparada de la tabla que figura en la propiedad tabla que coincida con el id que le ha llegado como parámetro.
        - borrar: recibe como parámetro una variable “id”. Este método borra el registro cuyo id de la tabla asignada en la propiedad tabla coincida con el id recibido.
    - Métodos abstractos que se implementarán en otras clases:
        - crear()
        - actualizar()

- **Clase Usuario.php hereda de Crud.php:** contiene datos de las personas que han adoptado algún animal:
    - Atributos privados:
        - Id, nombre, apellido, sexo, dirección, teléfono, edad, conexión.
    - Contante: TABLA que contendrá el nombre de la tabla que tiene los datos de los usuarios.
    - Métodos públicos:
        - constructor: invoca al constructor padre enviándole el nombre de la tabla (definido en la constante de la clase). Guarda en la propiedad conexión el manejador de la conexión que le devuelve la llamada a la función conexión de la clase padre.
        - Métodos mágicos __set y __get.
        - Método crear (implementación del método abstracto de la clase padre): inserta mediante una consulta preparada en la tabla (definida en la constante de la clase los datos de un usuario. Los valores de los datos se encuentran almacenados en las propiedades del objeto.
        - Método actualizar (implementación del método abstracto de la clase padre): hace una consulta preparada de tipo UPDATE en la tabla. Los datos se encuentran en las propiedades del objeto.
- **Clase Animal.php, hereda de Crud.php:** contiene los datos de los animales que hay en la protectora.
    - Atributos privados:
        - Id, nombre, especie, raza, género, color edad, conexión.
    - Contante: TABLA que contendrá el nombre de la tabla que tiene los datos de los animales.
    - Métodos públicos:
        - constructor: invoca al constructor padre enviándole el nombre de la tabla (definido en la constante de la clase). Guarda en la propiedad conexión el manejador de la conexión que le devuelve la llamada a la función conexión de la clase padre.
        - Métodos mágicos __set y __get.
        - Método crear (implementación del método abstracto de la clase padre): inserta mediante una consulta preparada en la tabla (definida en la constante de la clase los datos de un animal. Los valores de los datos se encuentran almacenados en las propiedades del objeto.
        - Método actualizar (implementación del método abstracto de la clase padre): hace una consulta preparada de tipo UPDATE en la tabla. Los datos se encuentran en las propiedades del objeto.
- **Clase Adopción.php, hereda de Crud.php:** contiene los datos de las adopciones que se han realizado en la protectora.
    - Atributos privados:
        - Id, idAnimal, idUsuario, fecha, razón, conexion.
    - Contante: TABLA que contendrá el nombre de la tabla que tiene los datos de las adopciones realizadas.
    - Métodos públicos:
        - constructor: invoca al constructor padre enviándole el nombre de la tabla (definido en la constante de la clase). Guarda en la propiedad conexión el

manejador de la conexión que le devuelve la llamada a la función conexión de la clase padre.
        - Métodos mágicos __set y __get.
        - Método crear (implementación del método abstracto de la clase padre): inserta mediante una consulta preparada en la tabla (definida en la constante de la clase los datos de una adopción. Los valores de los datos se encuentran almacenados en las propiedades del objeto.
        - Método actualizar (implementación del método abstracto de la clase padre): hace una consulta preparada de tipo UPDATE en la tabla. Los datos se encuentran en las propiedades del objeto.

Crear un script index.php en el que utilizando los métodos correspondientes:

- Se crean 3 animales, se borre uno, y se actualice otro cambiando alguna de las propiedades.
- Se crean 3 usuarios, se borre uno y se actualice otro cambiando alguna de las propiedades.
- Se crean 3 adopciones, se borre una y se actualice otra cambiando alguna de las propiedades.

6) Agenda electrónica

- Adapta el ejercicio de la agenda electrónica para que los datos se guarden en una base de datos que debe tener los siguientes campos y con ese nombre **(no usar tildes ni caracteres extraños en los nombres de columnas)**:
    - **codigo**: numérico de 6 números, clave primaria, autoincremental.
    - **nombre:** VARCHAR (50).
    - **telefono:** VARCHAR (12)
    - **correo:** VARCHAR (25)
    - **fechaNac:** DATE.
- Crea en PhpMyAdmin una cuenta de usuario con el usuario **"agenda"** y la contraseña **"2DAWdwes"**.
- Puedes crear un script para pasar los datos que tengas en el fichero de un ejercicio anterior a la BD. En la BD debe haber en total 20 contactos.
- Debes añadir una pantalla de login para poder acceder. El usuario será **"amiguis"** y la contraseña **"PHP4ever"**. No se puede acceder a la aplicación web si no se identifica correctamente. **El usuario y contraseña estarán guardadas en una tabla "usuarios" en la BD con los siguientes campos:**
    - usuario: VARCHAR(10)
    - password: VARCHAR(255)
- Si se identifica correctamente, se mostrará un listado paginado con los datos de los contactos. Se mostrarán 6 contactos por página.
- En todas las páginas de la aplicación web (excepto el login) debe haber un menú con las siguientes opciones:
    - **Añadir contacto:** formulario para añadir un nuevo contacto.
    - **Borrar contacto:** listado paginado, con 6 elementos por página con checkbox para poder borrar más de un contacto.
    - **Modificar contacto:** listado paginado, con 6 elementos por página con elementos de tipo input para poder cambiar más de un contacto (El código no debe poder modificarse).

    - **Buscar:** listado de los contactos que contengan en el nombre (solo nombre) el texto introducido en un formulario. Si no hay ninguno se mostrará un mensaje indicándolo.
    - **Listar:** listado paginado. Es la misma página que se muestra al entrar en la aplicación web después de identificarse.
    - **Salir:** Va a la página del login.
- Puedes crear la BD puedes crearla mediante PHPMyAdmin o por código, como prefieras.
- Hay que hacer todas las comprobaciones necesarias antes de operar con la BD, para evitar datos duplicados o eliminaciones .
- Puedes hacer los scripts que consideres necesarios.
- Usar PDO.