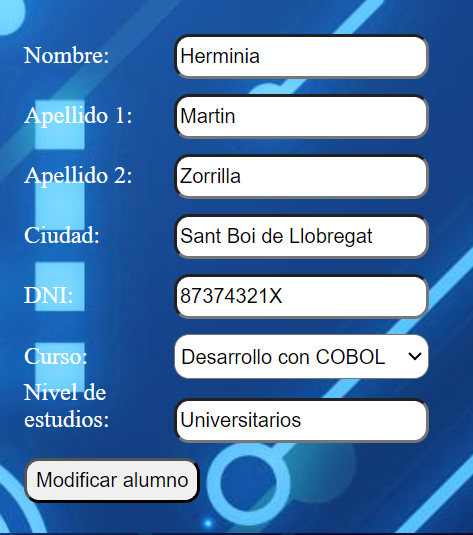
**Enlace a la página :** <https://nascor04.md360.es/CRUDGestion_academicaV2.0/controladores/leerAlumnosControlador.php>

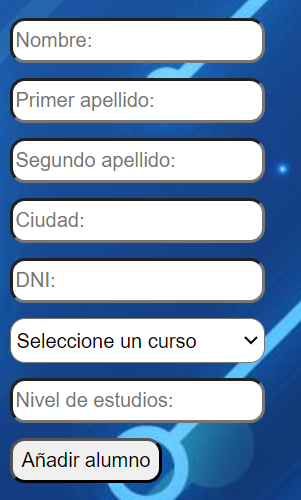
**Explicación funcional:**

Desde el inicio entraremos a la Lista de alumnos como página principal donde podremos Borrar/Modificar/Añadir alumnos y en cuanto lo hagamos veremos como la lista se actualiza y se cumple lo que hagamos cumpliendo con esta manera el modelo CRUD.

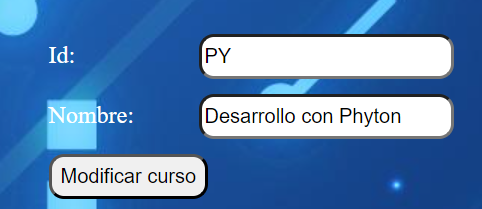


Al **BORRAR** elimina al alumno sin más.

Al **MODIFICAR** sale un formulario los datos actuales a modificar, una vez le damos a modificar, se cambian los datos automáticamente en la lista de alumnos:

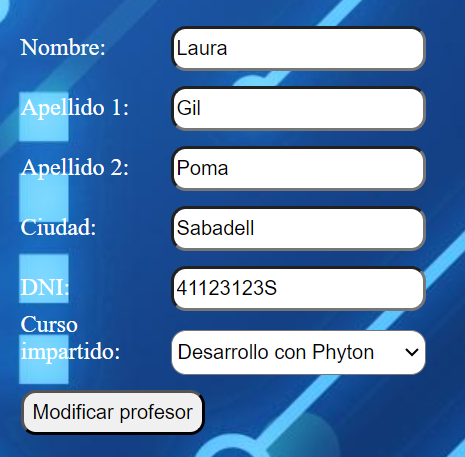


Al **AÑADIR** tendremos el formulario vacío para rellenar con los datos que queremos introducir, al darle a enviar crearemos el alumno que aparecerá en la lista de alumnos:

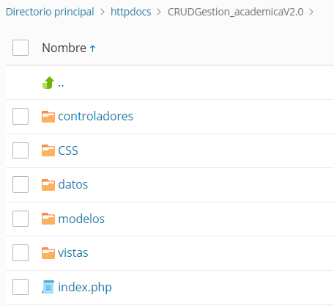
Tenemos en la zona superior un botón con listado de cursos: 

Aquí de igual forma podremos modificar añadir o borrar como en el caso de los alumnos saliendo el correspondiente formulario:

En el último botón superior tendremos el listado de profesores donde igual que antes podremos añadir borrar o modificar el profesor:



Añadido a la modificación, adición y borrado, en la lista de alumnos el curso será un enlace el cual al “clickar” nos llevará a un nuevo archivo donde se nos mostrará el curso y los **alumnos apuntados a ese curso:**

**PROGRAMACIÓN DE LA PÁGINA:**

Iniciaremos con las carpetas del host que contendrán nuestra página cumpliendo el CRUD, una carpeta de Controladores que unirán los archivos de las carpetas de los Modelos y las Vistas, una carpeta con los Datos donde guardamos los objetos y otra con el CSS para el estilo de la página. Además, claro el index.php.

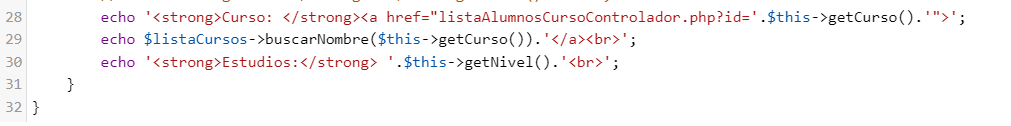
**Comenzaremos por la creación de clases:**

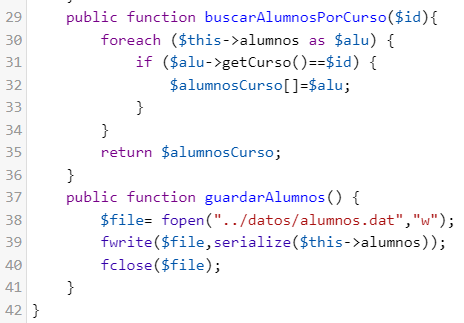


Crearemos **la clase principal de la aplicación: Persona**, donde le asignaremos unas propiedades de nombre, apellidos, DNI y ciudad y sus respectivos “setters” y “getters” para poder tanto introducir datos como extraerlos desde fuera de la clase. Finalmente tendremos una función mostrar datos mara mostrar todo desde fuera llamando a una simple función.

**La clase alumno:**

Añadiremos la clase persona y conjuntoCursos.En este caso, la clase alumno va a ser una hija de la clase persona que cogerá sus propiedades y además le añadira una propiedad curso y una propiedad nivel (académico). Haremos sus setters y getters y haremos también la función mostrar datos que dentro tendrá la función mostrarDatos() del padre para mostrar las propiedades de persona y además la propiedad curso y nivel. Si nos fijamos en la propiedad Curso: mostraremos el curso con un link que irá a listaAlumnosCursoControlador.php que después veremos para qué sirve.

****  
Las dos ultimas propiedades que mostraremos es el curso (para el que cogimos al inciar la clase la otra clase conjuntoCursos y el nivel.

**La clase conjuntoAlumnos:**

****

En esta clase obtendremos todo el conjunto de alumnos introducido y por lo tanto hay que llamar a la clase alumnos al iniciarla. Crearemos una variable alumnos que los contendrá.

Utilizaremos una función constructora con los datos que hay en el archivo alumnos, donde se guarda cada alumno que introducimos.

En esta clase conjunta haremos varias funciones:

**Borrar alumno**, donde hacemos un unset para borrar los datos del alumno en relación a $pos que será su posición del array. Después usamos la función guardar datos para que se actualicen.

**Añadir alumno,**  donde crearemos el array alumno con cada persona introducida y guardaremos los datos.

Otra función para **modificar** los datos del alumno. En este caso lo que haremos es que alumnos[$pos] que en este caso es el alumno con la posición que cogemos (alumno antiguo) sea igual a $per, el nuevo que introducimos. Es decir serian dos objetos distintos.

**Ademas tendremos la función de buscarAlumnosPorCurso** que nos servirá para buscar por cada alumno el curso que tiene asignado y lo mete en un array y nos devolverá este array con los datos.

Finalmente **una función guardar datos** para guardar los datos en un archivo alumnos.dat

**La clase curso:**

****

Creamos la clase curso donde le daremos una propiedad que será el identificador del curso y el nombre. Crearemos una función constructora con este ID y este Nombre y sus setters y getters.

**Clase conjuntoCursos:**

****

Donde lo primero que tenemos que hacer es llamar a la clase curso. Haremos una variable $cursos que luego utilizaremos como array.

Creamos una función constructora para descargar los datos de los cursos del archivo cursos.dat

Creamos una función de borrado, añadido y modificación para poder aplicar cada función en cada curso.

Tendremos la función buscarNombre que servirá para buscar el nombre del curso que se le asigna a un identificador.

Finalmente tendremos una función de guardado para actualizar nuestros datos cuando queramos.

**La clase profesor:**

La clase profesor al igual que alumno va a ser una extensión de la clase Persona. Creamos la propiedad curso que será la que tendrá además de las de persona y que será el curso que imparte. Crearemos sus setters y sus getters y la función mostrar datos que utilizará la del padre (persona) para mostrar nombre, apellidos, dni, ciudad y además le añadiremos un get curso para ver el curso que imparte.

**La clase conjuntoProfesores:**

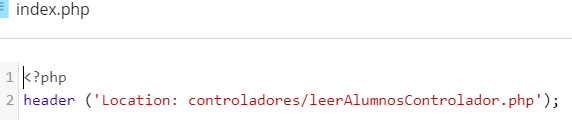
****Como es primero cogemos la **clase profesor**.

**Creamos la clase conjuntoProfesores** y creamos la variable profesores que será luego un array donde irá cada profe.

Creamos la **función constructora** como siempre para descargar los datos del archivo profesores.dat.

Creamos las **funciones/métodos de borrado añadido y modificado como siempre** cada una **con su respectiva función de guardarProfesores().**

**Finalmente una función de guardado.**

**Desde index:**

Accederemos directamente al archivo leer alumnos controlador donde haremos lo siguiente:

**LeerAlumnosControlador:**

cogeremos los datos de la clase de conjuntoAlumnos y haremos la función constructora en la variable $alumnos y llamaremos a la vista de leerAlumnosVista.php

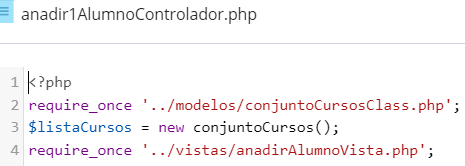
**LeerAlumnosVista:**

****

Aquí será la parte de **html y css** (que será la que veremos por pantalla) le aplicamos el CSS con la cabecera y el pie y tres botones que llevaran a añadir alumno a consultar cursos y a gestionar profesores.

A continuación mostrará el listado de alumnos mediante un **foreach** donde le decimos que por **cada alumno muestre sus datos junto a dos botones, uno de modificado y uno de borrado.**

Donde le pasaremos el ‘’parametro’’ $clave que será el número de la posición del array del objeto que estemos eliminando para luego cogerlo con un $get en su pertinente archivo.

**añadir1AlumnoControlador:**

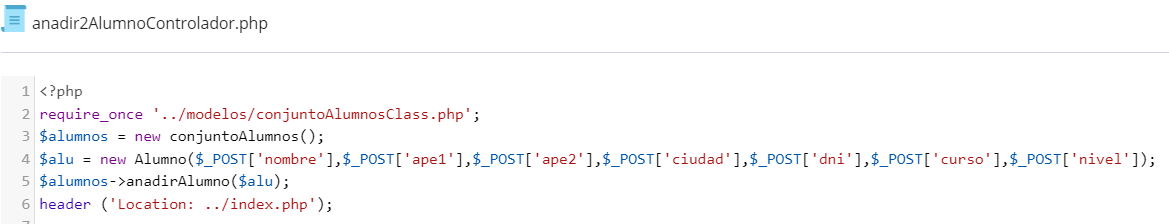
Llamamos al conjunto de cursos y hacemos una variable con los datos de conjuntoCursos para que en el desplegable salgan los cursos que hay al añadir un alumno. Llamamos a añadirAlumnoVista…

**añadirAlumnoVista:**

En este tendremos los archivos CSS y HTML como en todas las vistas. Tendremos un formulario donde introduciremos los datos del alumno y un foreach para mostrar los cursos (motivo por el que antes tuvimos que coger el conjuntoCursosClass y hacer una variable con la lista de cursos que tenemos).

Mediante el submit de este formulario accederemos al añadir2AlumnoControlador.php…

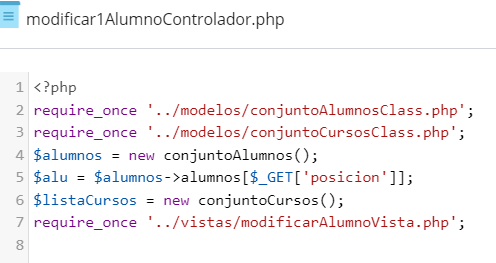
**añadir2AlumnoControlador:**



Aquí cogeremos la clase conjuntoAlumnos cogeremos la lista de alumnos y crearemos el $alu que será el nuevo alumno con los datos del formulario.

Dentro de $alumnos (lista de alumnos) haremos un añadir alumno donde le pasamos de parámetro el nuevo alumno y lo añade al array alumnos mediante la función. Finalmente vuelve a index.php (que mandará a leerAlumnos) y podremos ver el nuevo alumno en la página.

**Modificar1AlumnoControlador:**

****

Cogemos las clases conjuntoAlumnos y el conjuntoCursos.

Cogemos la lista de alumnos con newConjuntoAlumnos() y cogemos la posición que queremos modificar.

Tambien cogemos la lista de cursos y llamamos a modificarAlumnoVista.

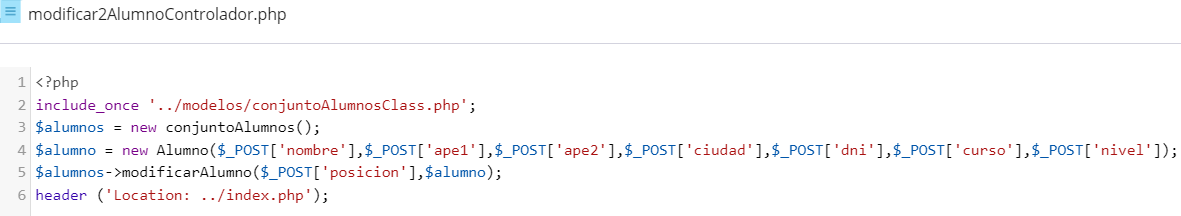
**modificarAlumnoVista:**



HTML+CSS.

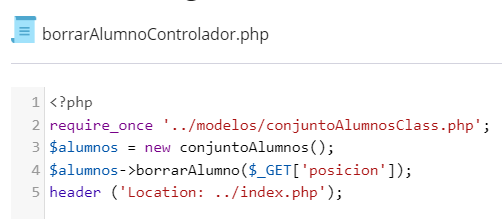
Formulario con los datos donde con el $alu anterior conseguimos introducir los datos del alumno que queremos modificar en el input mediante el value de cada uno. En el foreach, le pasaremos un if para ver cual es la opción que el usuario tiene seleccionada y finalmente antes del submit un input type hidden con la posición como valor. Con el submit iremos a modificar2AlumnoControlador.php….

**modificar2AlumnoControlador:**



Cogemos el conjuntoAlumnos y la lista de alumnos que tenemos, creamos un nuevo alumno en $alumno y aplicamos dentro de la lista de alumnos el método modificarAlumno donde como primer parámetro le pasamos la posición del alumno a modificar y las nuevas propiedades. Es decir estamos creando un nuevo alumno en realidad. Llamamos al index.

**borrarAlumnoControlador:**

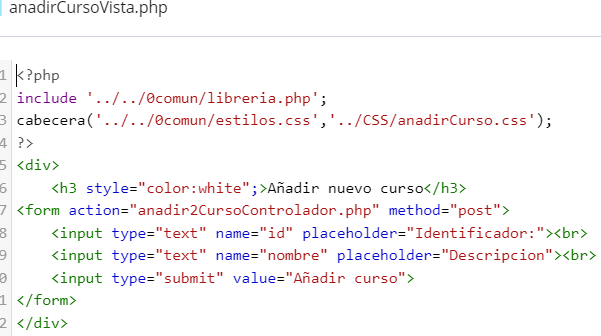
En este caso solo tendremos un archivo de borrador que ira al index.php ya que no necesitamos ver formularios ni nada similar, simplemente que al darle a botón de borrar, borre.

Cogeremos la clase conjuntoAlumnos, la lista de alumnos y **aplicaremos el método borrarAlumno con la posición del alumno a borrar (que se introduce mediante el botón con la URL)** y volvemos al index.php.

**añadir1CursoControlador:**

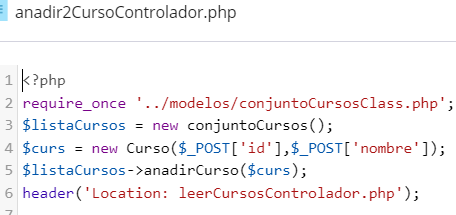
****

Vamos a la vista… **añadirCursoVista:**

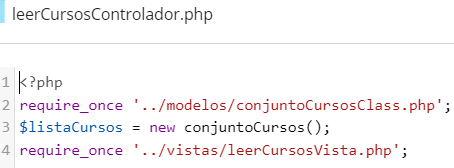


HTML+CSS.

Formulario básicopara añadir un nuevo curso y que llamará a añadir2cursoControlador. **añadir2cursoControlador:**

****Cogemos la clase conjuntoCursos, con la lista de cursos y creamos uno nuevo con los datos del formulario, utilizamos la función añadirCurso con el curso nuevo y llamamos a **leerCursosControlador.php**.

**leerCursosControlador:**

Cogemos la clase conjuntoCursos y la lista de cursos y vamos a **leerCursosVista**…

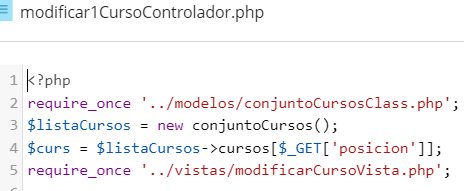
**leerCursosVista:**

****

HTML+CSS, tendremos los botones para Añadir un nuevo curso, para ir a la gestión de alumnos y a la gestión de profesores.

Mostraremos con un foreach cada curso con botones de borrado y modificación con las url con la posición correspondiente, su identificador y su nombre.

**modificar1CursoControlador:**

****

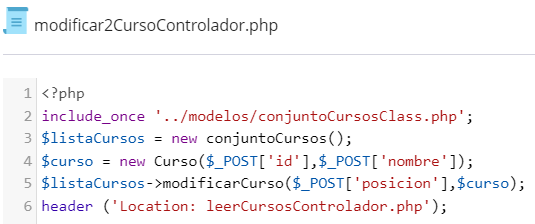
Cogemos la lista de cursos, borramos de la lista de cursos el curso con la posición introducida por la url en el botón ($curs) y vamos a **modificarCursoVista**…

**modificarCursoVista:**



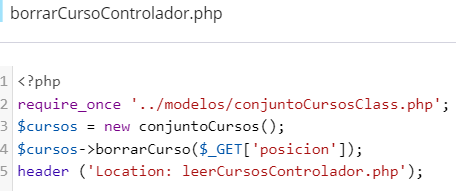
HTML + CSS + FORMULARIO. Tendremos los datos que estaban introducidos en el curso que estamos modificando (mediante el valor de $curs).Antes del submit tenemos un input type hidden donde le pasamos el valor del numero de la posición del objeto que vamos a cambiar. Mediante el submit vamos a **modificar2CursoControlador**.php

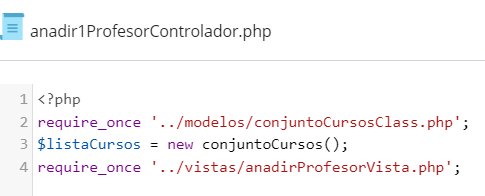
**modificar2CursoControlador:**



Cogemos el conjunto de cursos mediante la clase, creamos el nuevo concurso y usamos la función modificarCursos donde el primer parámetro es la posición del objeto a modificar y el segundo es el nuevo valor que le damos a ese objeto (que realmente es un objeto nuevo). Volvemos a **leerCursosControlador para que coja de nuevo la lista de cursos y vaya a la vista a mostrarlos**.

**borrarCursoControlador:**

Cojemos la lista de cursos que tenemos, borramos el curso que queremos con el parámetro con la posición que tenemos pasada en la url del botón y volvemos a leerCursosControlador para coger la nueva lista con este curso borrado.

**añadir1ProfesorControlador:**

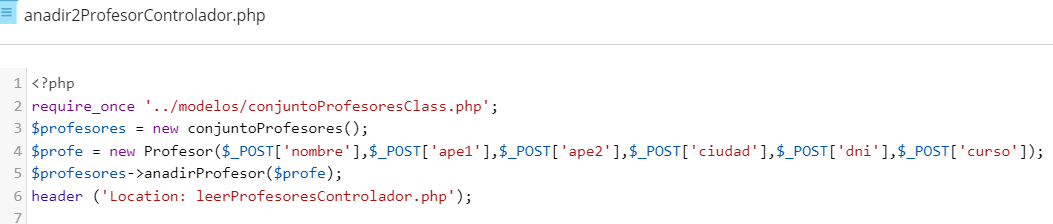
Cogemos el listado de cursos que tenemos y la clase conjuntoCursos (que le tendremos que pasar al profesor) e ir al añadirProfesorVista con el formulario a rellenar…

**añadirProfesorVista:**

****

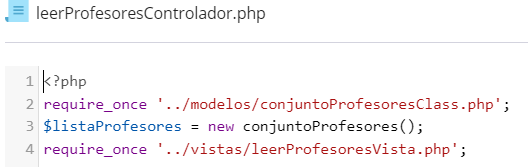
HTML CSS Y FORMULARIO. En el formulario añadiremos las propiedades de persona, y el curso que va a impartir con un foreach de listaCursos. Al hacer submit iremos a **añadir2ProfesorControlador.**

**añadir2ProfesorControlador:**



Cogemos la lista con el conjunto de profesores que tenemos, creamos un nuevo objeto profesor con las propiedades recogidas del formulario ($profe) , hacemos la función añadirProfesor con $profe como parámetro para que se añada este objeto con los datos recogidos al array de profesores. Vamos a leerProfesoresControlador.

**leerProfesoresControlador**:



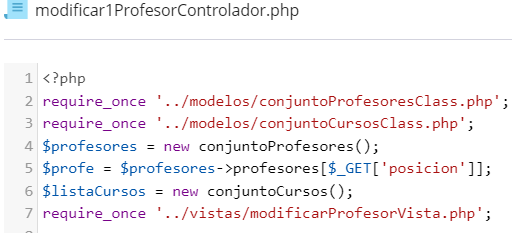
Cogemos la lista del conjuntodeProfesores y la llevamos a la vista donde la mostraremos con un foreach.

**leerProfesoresVista**:

HTML + CSS . Como siempre tendremos botones para ir a gestionar alumnos y consultar cursos y en este caso añadir profesor.

Mostraremos el listado de profesores con un foreach con un botón para cada uno de borrar y modificar con las url correspondientes con su posición y la función mostrarDatos para mostrar los datos de cada profesor.

**modificar1ProfesorControlador**:



Cogemos las clases de conjuntoProfesores y conjuntoCursos, cojemos el listado de profesores y cogemos la posición a modificar pasada por el botón modificar. Cogemos la lista de cursos para que nos salgan en el desplegable a la hora de cambiar el curso que imparte (si quisiéramos cambiarlo). Finalmente vamos a la vista de modificarProfesor.

**modificarProfesorVista**:



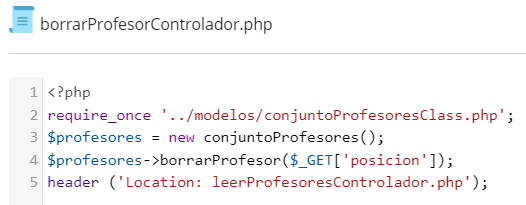
HTML, CSS Y FORMULARIO. En el formulario tendremos los datos del profesor que hemos elegido modificar y la lista de cursos con un foreach para poder cambiar el curso que imparte si fuese necesario. Tendremos luego un input type hidden para pasar en el value la posición a modificar. Al hacer submit vamos a modificar2ProfesorControlador.

**modificar2ProfesorControlador:**

****

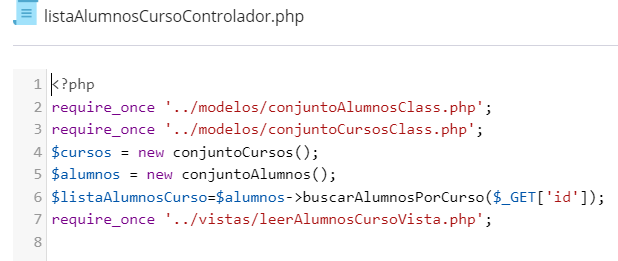
Cogemos la lista de profesores, ponemos los datos introducidos en el formulario en $profesor y mediante la función modificarProfesor cambiaremos el objeto con la posición pasada por el formulario por el nuevo profesor ($profesor). Realmente no modificamos, creamos un nuevo objeto. Volvemos a leerProfesoresControlador para actualizar la lista e ir a la vista luego.

**borrarProfesorControlador**:

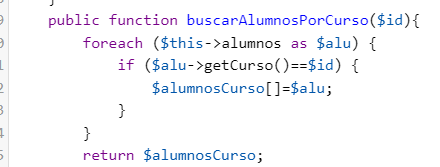


Cogemos la lista de profesores hacemos la función de borrarProfesor con el valor introducido en la url del botón y volvemos a leerProfesores controlador para actualizar la lista y volver luego a la vista automáticamente con el profesor borrado.

**listaAlumnosCursoControlador**:



Cogemos el listado de alumnos y de cursos usamos la función de buscarAlumnosPorCurso como parámetro la pasaremos el identificador.

Recordar que la función hacia que en cada alumno cogíamos el curso y lo comparábamos con el parámetro y si eran iguales lo metíamos al array alumnosCurso[] para que metiese a todos los alumnos que pertenecen al mismo curso en el mismo array.

y vamos a **leerAlumnosCursoVista**.



Donde veremos además del CSS y el HTML con sus botones correspondientes de las otras secciones, un titulo donde saldrá esta vez el nombre del curso y a continuación con un foreach cada alumno que esta en ese curso.