CFGS DESARROLLO APLICACIONES WEB **CURSO 2023/24**
DESARROLLO WEB EN ENTORNO SERVIDOR **18/12/20223**

NOMBRE: _____

A partir de la estructura usada en clase, crear un proyecto para el examen. Se mantendrán las clases creadas anteriormente en la carpeta /scripts/clases y las clases que creemos en /scripts/examen. Se debe garantizar la autocarga en ambas carpetas. Se tendrá también la librería validaciones.php en su ubicación correcta.
Siempre que tengamos un objeto, función o método que nos haga algo se usará.
Se tendrán en cuenta todas las consideraciones que ya conocemos como modelo/vista/controlador, definir tipo de propiedades y parámetros, uso de funciones mb_, librería validaciones, etc

Desde el Ayuntamiento de Antequera nos piden un programa para controlar la campaña de "Bonos Navideños"
1.- (2.45 ptos) Lo primero es crear la clase **Beneficiario**, es decir, la persona que va a comprar el abono. La clase Beneficiario tiene las siguientes características:

- Constante privada **TIPOSREDUCCION**. Es un array con valores 1: Sin reducción, 2: Discapacidad, 3: Familia numerosa.
- Propiedades: **Nombre** (cadena 30 caracteres, obligatoria), **NIF** (cadena de 9 caracteres, obligatorio, formato de 99999999A o A9999999A –donde A es una letra y 9 es una cifra), **Reducción** (entero, un valor de entre los indicados en la constante TIPOSREDUCCION, por defecto 1), **Reduccion_texto** (cadena, se obtendrá a partir de reducción, se corresponde con la descripción de la reducción), **Fecha_nacimiento** (fecha en formato dd/mm/aaaa, no puede ser posterior a hoy, por defecto hace 10 años), **Mayor_edad** (booleano o entero "0-1", si tiene los 18 años o no, obtenido a partir de la diferencia en años entre la fecha_nacimiento y hoy, por defecto false), Estas propiedades no pueden ser accedidas directamente desde el exterior aunque si desde las clases descendientes.
- Se tendrá el constructor al que se le pasan parámetros para rellenar las propiedades (solo los no calculados; aquellos que tienen valores por defecto, pueden no indicarse en la llamada y se rellenarán con los valores por defecto). Se deben comprobar las restricciones y en caso de que no las cumplan se asigna el valor por defecto. Si no se cumplen las restricciones para dos o más propiedades, lanzar una excepción.
- Se tienen métodos getXXXX para todas las propiedades.
- Tenemos métodos setXXXX para las propiedades NOMBRE, NIF, REDUCCION y FECHA_NAC. Se deben comprobar las restricciones. Si no cumple las restricciones no se asignará el valor a la propiedad correspondiente. Devuelve true/false si se ha podido o no hacer la asignación.
- No se permite la sobrecarga dinámica.
- Cuando se convierte en cadena devuelve el texto "Beneficiario ***NOMBRE*** con nif **NIF** que nació el **FECHA_NAC**, que **es/no es** Mayor de edad y tiene reducción **REDUCCION_TEXTO**.

2.- (1.9 ptos) Se controlarán los bonos que tiene cada beneficiario para lo que creamos la clase **Bonos**. En esta clase se controlará para cada bono el numero y el importe del mismo usando propiedad dinámicas. La clase **Bonos** tendrá las siguientes características:

- Se tiene la propiedad privada de clase **IMPORTE_MAXIMO** con valor inicial de 100 (entero).
- Se tiene la propiedad privada **importe_total** que almacena el importe total de los bonos anotados (entero). No se tendrá ningún método get/set para la propiedad importe_total. Se podrá acceder a este importe a través de una propiedad dinámica llamada **importe** y de la que solo se podrá leer (acceder a su valor). La propiedad dinámica importe no debe estar guardada en ningún lugar. Por ejemplo, si importe_total vale 50, será válido $bon->importe que devolverá 50. Aunque escribamos en esta propiedad no se modificará importe_total.
- Los bonos se definirán como propiedades dinámicas que se guardarán en el array asociativo **MisBonos** de forma que la posición será el número de bono (entero, que se precede de B) y el valor es el importe (entero, excepción si no). Por ejemplo, si tenemos el bono con numero 235 e importe 10 se guardará en la posición asociativa B235 con el valor 10. Se accederá como $bon->235. Además, debe quedar actualizado correctamente el importe_total y no se permitirá asignar un bono si se supera el IMPORTE_MAXIMO (lanza una excepción). $bon->nav no se debe

permitir al no ser entero (excepción).

- Debe habilitarse el recorrido de la propiedad importe y de los bonos que hayamos indicado (foreach), teniendo en cuenta que importe debe ser el primero que aparezca.

3.- (0.9 ptos) Añade a la clase Beneficiario soporte para los bonos. Para ello:

- Se tendrá la propiedad privada **bonos** que será un objeto de la clase **Bonos**.
- Tenemos el método **aniadeBonos** con al menos 3 parámetros: nbonos (entero, se devolverá el número de bonos que se han podido asignar), bono (string, numero de bono a asignar) y valor (string valor del bono a asignar). En el momento de llamar a este método se pueden indicar mas bonos-valores a incluir en el beneficiario.

  Este método asigna los bonos que se indican como parámetros (no se verifican directamente el importe sino que se añadirá y se controlará con un bloque try/catch). En el parámetro nbonos se devolverá el número de bonos que se han podido asignar. Este método devuelve true si ha asignado al menos un bono y false si no se ha podido asignar ningún bono.
- Método **getImporteBonos**. Este método devuelve el importe de los bonos anotados.
- Método **getListaBonos**: Devolverá un array con la lista de bonos dados de alta para el beneficiario, donde cada elemento será de la forma numero_bono => importe_bono

4.- (0.75 ptos) Para mantener los beneficiarios tendremos el array $BENEFI cuyos elementos serán del tipo Beneficiario (se definirá en un solo punto). Se debe garantizar que podamos guardar nuevos beneficiarios y tenerlos actualizados entre diferentes llamadas a páginas. Inicialmente el array tiene que tener 2 beneficiarios con al menos 2 bonos cada uno (insertados con el método aniadeBonos.

5.- (0.75 ptos) Crear la función cargarBeneficiariosDesdeFichero que recibe como parámetros el nombre del fichero y un array donde guardar los beneficiarios que se encargará de leer y analizar el fichero para crear tantos beneficiarios (se debe controlar posibles excepciones al crear bonos) como indique el fichero. Os adjunto el fichero a_incorporar.txt con datos que se deben poder incorporar.

6.- (3.25 ptos) Interfaz de la aplicación.La página index.php tendrá:

- **Botón loguearse**: Se tendrá una cookie con valor inicial a 1 y que se increntará cada vez que se pulse este botón. Si el valor es par se registrará al usuario PAR, con permiso 1 y si es impar al usuario IMPAR con el permiso 2. Junto al botón aparecerá si hay o no usuario registrado y el nombre del usuario y si esta logueado un botón para cerrar sesión
- Párrafo (o textarea) **Mostrar Beneficiarios** en el que se mostrarán todos los beneficiarios que haya actualmente (información del beneficiario y de los bonos que tiene incluyendo el importe -se usará foreach-). Esta opción aparecerá si hay usuario logueado y tiene el permiso 2. Si no hay usuario o no tiene el permiso 2 aparecerá el texto “sin permiso para consultar los datos”
- Botón **Cargar Fichero**. Cuando se pulse este botón usará la función cargarBeneficiariosDesde Fichero() para recoger los beneficiarios del fichero “a_incorporar.txt” que se adjunta y que situaremos en la carpeta /imágenes.
- Botón **Nuevo beneficiario**. Al pulsarlo nos llevará a la pagina /aplicacion/beneficiarios/nuevo.php. Esta página tendrá en formulario nuevo beneficiario con cajas para el nombre, fecha y un select para reducción. Cuando se envíen debe validarse. En caso de que no cumplan algún requisito, se mostrará de nuevo el formulario con el mensaje de error apropiado y los valores anteriores. Si todo es correcto, crea el beneficiario y lo añade al array $BENEFI (garantizando que esté disponible el nuevo beneficiario para otras ventanas) y vuelve a index.php
- Formulario **Exportar**. Tendremos un combo para mostrar los beneficiarios disponibles (debe tenerse una línea más con un beneficiario que no exista) y un botón exportar. Cada opción del combo tendrá como valor el índice en $BENEF y como texto “nombre - fecha”. Cuando se pulse el botón se llamará a /aplicación/beneficiarios/descarga.php que debe descargar un fichero de texto cuyo contenido será los datos-bonos del beneficiario seleccionado.