Familia Profesional		Nombre del Ciclo Formativo		
Informática y Telecomunicaciones		Título de **Técnico Superior en Desarrollo de Aplicaciones Web**		
Centro Educativo		Módulo Profesional		Profesor
IES Campanillas (sede CITIC)		**Desarrollo Web en Entorno Servidor** Código: **0613** N.º de créditos ECTS: **12**		**Luis José Sánchez González**
Curso lectivo	Grupo	Tipo de documento	Trimestre	Fecha
2015 / 2016	**2º DAW**	**Examen**	**Segundo**	**18 de enero de 2016**

INSTRUCCIONES

➔ El alumno debe entregar una carpeta con las soluciones al examen cuyo nombre debe estar formado por el número de lista seguido de las iniciales. Por ejemplo, Facundo Romuedo Piladro que es el número 8 de la lista entregaría una carpeta con nombre **Ex08frp**.
➔ Los ficheros o carpetas correspondientes a las soluciones se deben nombrar igual que la carpeta junto con el número del ejercicio, por ejemplo **Ex08frp1.php, Ex08frp2.php, etc**. en caso de ser ficheros o **Ex08frp1, Ex08frp2, etc**. en caso de ser carpetas.
➔ En los comentarios de cada programa **se debe indicar el nombre completo**, la fecha y - si procede - el turno.
➔ Cuando el ejercicio haga uso de bases de datos, **se debe incluir el fichero .sql correspondiente**.
➔ Únicamente se necesita entregar el código fuente en php junto con las imágenes y las hojas de estilo en caso de que las haya. No se deben entregar los ficheros o carpetas con información del proyecto (por ej. la carpeta nbproject).

EJERCICIOS

1. Una empresa está desarrollando una aplicación para llevar la gestión integral de un hotel. El primer paso será definir la clase **Habitacion** con los atributos y métodos necesarios. Sobre cada habitación se necesita saber su **número**, el **tipo** – clásica, superior o suite – y su **estado** – libre, reservada u ocupada. La clase debe almacenar información sobre el **número total de habitaciones** y sobre el **número de habitaciones que quedan libres**. Siempre que se crea una habitación nueva, su estado es "libre". El método **reserva** debe poner el **estado** en "reservada" en caso de que esté libre (y debe decrementar el total de habitaciones libres); por el contrario, en caso de estar ya "reservada" u "ocupada", se debe mostrar el correspondiente mensaje de error. El método **ocupa** debe poner el estado en "ocupada", independientemente del estado que tuviera anteriormente. El siguiente código del programa principal debe dar la salida que se muestra:

```php
<?php
  include 'Habitacion.php';
  $h1 = new Habitacion(237, "clásica");
  $h2 = new Habitacion(418, "suite");
  $h3 = new Habitacion(217, "superior");
  $h2->reserva();
  $h3->ocupa();
  echo "Hay ".Habitacion::getTotalHabitaciones()." habitaciones.<br>";
  echo "Quedan ".Habitacion::getHabitacionesLibres()." libres.<br>";
  echo $h2."<br>";
  $h2->reserva();
  echo $h3."<br>";
  $h3->reserva();
?>
```

```
Hay 3 habitaciones.
Quedan 1 libres.
Habitación nº 418, suite, reservada
Esa habitación ya está reservada.
Habitación nº 217, superior, ocupada
Esa habitación ya está ocupada, no se puede reservar.
```

2. Realiza un programa que escoja al azar 10 cartas de la baraja española y que diga cuántos puntos suman según el juego de la brisca. La salida de este programa es análoga a la del ejercicio realizado en clase del capítulo sobre arrays. El programa se dividirá ahora en dos partes: la definición de la clase **Carta** y el **programa principal**. Emplea un array asociativo como atributo de clase para obtener los puntos a partir del nombre de la figura de la carta. Cada carta que se muestra debe ser un objeto de la clase Carta. No es necesario comprobar si se repiten o no las cartas echadas.

(continúa en la siguiente página)

3. Mejora el programa **Expocoches Campanillas** realizado en clase de tal forma que se puedan añadir nuevas zonas a la aplicación o borrar zonas ya creadas (no es necesario implementar la modificación de zonas).

4. Modifica el ejercicio **Expocoches Campanillas** realizado en clase teniendo en cuenta que ahora las entradas están numeradas y, además, se pueden reservar. Para simplificar el ejercicio, las entradas solo se pueden reservar o comprar de una en una y la entrada número cero se puede despreciar. Añade **"Reservar entrada"** a las opciones del menú. Para reservar una entrada el usuario dará su nombre y se le asignará la primera entrada que estuviera libre. La opción **"Vender entrada"** permitirá vender una única entrada y se indicará al usuario el número de la entrada vendida. Al elegir la opción **"Vender entrada"**, además de preguntar por la zona, se preguntará al usuario si tenía reserva, en caso afirmativo, se preguntará por el nombre y, tras comprobar la reserva, se le venderá la entrada. Si por el contrario, el usuario dice que no tenía reserva, se vende la entrada directamente. Crea las tres zonas con poco aforo (por ej. 10, 20 y 7) para que el programa sea "manejable". La manera más sencilla de implementar este sistema, es tener como atributo para cada zona un array de cadenas de caracteres con la información necesaria para cada elemento (cada entrada): "LIBRE", "VENDIDA" o el nombre de la persona que tiene la reserva. El método **__toString()** debe mostrar el estado de cada entrada dentro de la zona.
   Por ejemplo: 1:VENDIDA 2:VENDIDA 3:Pepe 4:Marta 5:VENDIDA 6:LIBRE 7:LIBRE

Nota: Los ejercicios 3 y 4 no se pueden entregar juntos, deben ir obligatoriamente por separado.