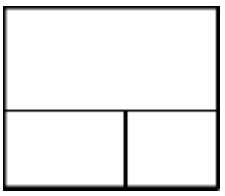
DWES Examen final Eval. Ordinaria 22-mar-2021



Se te habrá proporcionado un proyecto PhpStorm con este enunciado y una carpeta por ejercicio. Las carpetas son la base sobre la que se recomienda realizar los ejercicios. Tiempo para la realización: 1h30.

[1] Juego de memoria *(3,00p)*

(Solo PHP, sin DAO, sin Ajax)

Programa una aplicación web para ejercitar la memoria recordando uno o varios números. Se deberá poder elegir al inicio cuántos números será necesario recordar, por lo que esto es lo primero que debe preguntar la aplicación.

También existirá la opción de resetear la aplicación, es decir, eliminar toda la información de la aplicación (estado del juego en curso y mejor cantidad de respuestas correctas registrada). Esta opción estará disponible desde cualquier página.

A partir de ahí empezará el juego.

Por ejemplo, esto es el flujo de ejecución para recordar un numero (programar correctamente este caso de un número da 1,50 puntos).

- 1. Al abrirse, mostrará un número aleatorio entre 1 y 100 (denomínese número inicial) en forma no editable. Elige una manera adecuada para mostrarlo. Y un botón. El usuario deberá recordar el número.
- 2. Pulsara el botón y aparecerá otro número aleatorio, en la misma manera que el inicial, y un campo vacío para introducir otro número. El usuario deberá recordar el nuevo número, introducir el número inicial y pulsar el botón.
- 3. Mientras el usuario acierte, se repetirá sucesivamente el paso 2, debiendo introducir en cada caso el número aleatorio que mostraba la página anterior.
- 4. Cuando el usuario falle, la web informará del fin del juego e indicará cuántas respuestas correctas ha habido, así como cuál ha sido la mayor cantidad de respuestas correctas registrada. Habrá un botón para volver al inicio.

El flujo de ejecución para recordar cuatro números sería similar, pero en cada momento el usuario (y la aplicación) deberá mantener en memoria cuatro números en lugar de uno (programar correctamente el caso de N números a elegir da la puntuación completa del ejercicio). Así:

- La página inicial mostrará cuatro números aleatorios por ejemplo, 71, 44, 15 y 90. El usuario pulsará el botón.
- En la segunda página el usuario deberá introducir el primero de ellos, 71. Además, en esta página habrá otro número, por ejemplo 28, que se sumará a la lista, en último lugar, teniendo, de nuevo, cuatro números en memoria: 44, 15, 90 y 28.
- En la siguiente página deberá introducir el 44 y aparecerá otro número aleatorio más.
- Y así, sucesivamente. El juego terminará cuando haya un fallo e informará de la mayor cantidad de respuestas correctas registrada. Habrá un botón para volver al inicio.

Toda información de la aplicación, incluyendo el estado del juego y la mejor cantidad de respuestas correctas registrada, deberá permanecer aunque el usuario cierre el navegador, haciendo uso, para ello, de cookies.

[2] Restaurante DAO *(3,50p)*

(Sí se pide "DAO", no se pide inicio de sesión ni nada de eso, ni se pide AJAX.)

Importa la BD incluida en el ZIP. Se trata de la BD que dará soporte al TPV (Terminal Punto de Venta) de un restaurante. El restaurante vende menús o platos individuales. Las ventas se recogen en tickets, asociados a un

número de mesa. Los tickets se abren sin fecha de "pagado" (se les establece null) y, cuando los clientes los pagan, se cambia ese null por la fecha-hora en la que se produce el pago. Cada ticket lleva asociados de 0 a N menús y de 0 a N platos. Un menú se compone de platos y un plato se compone de componentes, de los que quedan determinadas unidades en stock. El modelo de datos proporcionado incluye la estructura para ello.

Programa lo necesario para:

- (1,00p) Gestionar componentes disponibles y su stock.
- (1,00p) Gestionar los platos y menús disponibles y su composición.
- (1,50p) Gestionar los tickets (abrirlos, añadir platos o menús respetando stock, marcarlos pagados).

Se pide preparar toda la infraestructura que necesitaremos para crear los scripts/páginas que se piden. Esto incluye la(s) clase(s), el "DAO" y demás.

Queremos utilizar el modelo DAO y queremos implementar las funciones de la manera más organizada posible, con el objetivo de crear una aplicación fácil de mantener y modificar, evitando copiapegas, etc.

Organiza el código lo mejor que puedas, dentro de estas premisas. Un objetivo en este ejercicio es demostrar que sabes organizar código. Por tanto, conseguir la funcionalidad indicada sin organización no es valorable.

[3] TPV AJAX *(3,50p)*

Se utilizará la BD del ejercicio anterior. Impórtala si no lo has hecho ya.

Se pide programar la misma funcionalidad que en el ejercicio anterior pero en modalidad AJAX, es decir:

- (1,00p) Gestionar componentes disponibles y su stock.
- (1,00p) Gestionar los platos y menús disponibles y su composición.
- (1,50p) Gestionar los tickets (abrirlos, añadir platos o menús respetando stock, marcarlos pagados).

Se pide implementar las funciones de forma lo más "Ajax" posible, es decir, evitando uso de la red y distribuyendo las responsabilidades entre los distintos agentes de la forma más eficiente.

En general, para todo el examen: decide tú las cuestiones que quedan abiertas. Si consideras que tu solución requiere hacer adiciones o modificaciones en la BD o en los ficheros que se proporcionan, hazlo. Realiza los cambios que necesites, siempre que estos contribuyan a una solución de mayor calidad (más clara, más flexible, con más genericidad que permita posteriores potenciales cambios). Sin perjuicio de lo anterior, en particular:

- Se pide realizar control de errores para detectar las situaciones habituales tales como introducción de texto en lugar de números o pulsado del botón sin haber introducido nada. En estos casos se deberá volver al punto en el que se estaba cuando se introdujo una entrada incorrecta mostrando un mensaje en la parte superior. En caso de no realizar este control de errores se podrá perder hasta ⅓ de la puntuación de los ejercicios.
- Se pide realizar una programación buscando la flexibilidad y la genericidad en el código, de manera que se facilite la futura introducción de cambios. En caso de no respetar estos principios se podrá perder hasta el 50% de la puntuación de los ejercicios.

Implementa primero lo básico y luego lo accesorio. Implementa primero los puntos que te resulten más rentables y luego los que menos. No te líes con detalles de control de errores si todavía no tienes la funcionalidad principal. Se pide explícitamente organizar el código según los modelos vistos en clase, u otros que sean objetivamente mejores en términos de calidad del código, simplicidad, genericidad y facilidad para su modificación. Dicho de otro modo, "que funcione" no es el único objetivo de los ejercicios: también es necesario realizar una programación de calidad según los criterios expuestos.

Puedes utilizar cualquier material de clase que exista y que esté cargado en tu ordenador previamente al inicio del examen. No puedes comunicarte con otras personas ni con otros ordenadores o servicios fuera de tu ordenador, excepto con el profesor y para la entrega del examen, que será por duplicado en una URL proporcionada por el profesor vía email al finalizar el tiempo de examen, así como adjunto por email a alain.fernandez@educa.madrid.org