CHECKLIST
para corregir el proyecto Reservando

Para que un proyecto apruebe u obtenga categoría hacker, tiene que cumplir con todas las condiciones del checklist correspondiente.

Condiciones para entregar _

- Archivos mínimos entregados: En la carpeta entregada se incluyen los archivos disponibles en los recursos descargables, el archivo tests.js con lo tests y el archivo reserva.js.

- Funcionalidades desarrolladas: Revisar que el archivo el archivo tests.js y reserva.js no estén vacíos.

Condiciones para aprobar _

- Tests de la función reservarHorario(horario): Se testean al menos los 3 casos de prueba que se mencionan en la guía.

- Tests de la función obtenerPuntuacion(): Se testean al menos los 2 casos de prueba que se mencionan en la guía.

- Tests de la función calificar(): Se testea al menos 1 caso de prueba elegido por el/la estudiante.

- Tests de la función buscarRestaurantes(id): Se testea al menos los 1 caso de prueba elegido por el/la estudiante.

- Tests de la función obtenerRestaurantes(): Se testea al menos 1 caso de prueba elegido por el/la estudiante.

- Uso de filter en la función reservarHorario(): Se refactorizó correctamente la función aplicando filter().

- Modularización de la función obtenerPuntuacion(): Se crearon las funciones promedio() y sumatoria() para poder modularizar la función

y evitar que se encargue de sumar los elementos el arreglo de calificaciones y sacar el promedio.

- Variables y funciones declarativas: Se cambió el nombre de funciones que en los recursos se llamaban obtC(), obtR() y obtH() por uno que represente mejor su objetivo. Se hizo lo mismo con las variables utilizadas en esas funciones.

- No hay repetición de código en las funciones obtenerRubros(), obtenerHorarios() y obtenerCalificaciones(): Se creó una función que se encarga de eliminar los elementos repetidos de un arreglo. Las funciones que obtienen todos los rubros, horarios y calificaciones del listado la utilizan para evitar la repetición de código.

- Las funciones que obtenerRubros(), obtenerHorarios() y obtenerCalificaciones() utilizan la función map(): Estas funciones obtienen los atributos que necesitan de cada restaurante utilizando la función map().

- Las función obtenerRestaurant(id) utiliza la función find(): Esta función utiliza la función find() para encontrar un restaurante con un determinado id.

- Se implementa correctamente el objeto reserva: Se crea el objeto reserva con los atributos que corresponden y se implementan las funcionalidades de obtener el precio base y calcular el precio final.

- El cálculo del precio final de la reserva se encuentra modularizado: Se crean funciones que se encargan de calcular los descuentos y los adicionales y se las llama desde la función que calcula el precio final.

- Se crean tests que validan el funcionamiento de las funcionalidades del objeto reserva: Hay al menos un test de cada funcionalidad de la reserva que se utilizó para crear las funcionalidades usando TDD.

Condiciones para Categoría Hacker _

- Los tests tienen nombres que expresan claramente lo que se está testeando: Los tests creados tienen nombres que expresan bien que funcionalidad se está testeando y qué caso de prueba se está utilizando.

- Se identifica correctamente más de un caso de prueba en las funciones calificar(), buscarRestaurantes(id) y obtenerRestaurantes(): Se testea más de un caso de prueba para estas funciones.