Ejercicios Integradores

Javascript - React

1- Cree un contador utilizando HTML y Javascript que comience en 10 y cuente de 0 a 20. Incluir un botón para incrementar el contador de 1 en 1 y otro botón para decrementar el contador de 2 en 2. Los botones deben deshabilitarse automáticamente cuando una acción de incremento o decremento resulte en un valor fuera del rango especificado.

2- Realizar una aplicación en Vite con React que contenga el siguiente formulario:

- Un input de tipo texto con un label que diga "Nombre".
- Un input de tipo number con el label que diga "Edad".
- Un botón que contenga el texto "Agregar".

Dicho formulario debería permitir lo siguiente:

a) Al presionar el botón "Agregar", se debería cargar en una lista un Objeto que contenga el Nombre y la Edad ingresada.

b) Una vez agregado el objeto a la lista, se debería blanquear los campos de Nombre y Edad.

c) Mostrar dentro de un span si existe una persona que cumpla con la mayoría de edad mediante la frase “Existe una persona mayor de edad”.

d) Mostrar dentro de otro span el promedio de edad de las personas de la lista, mediante la frase "El promedio de edad de las personas de la lista es: "

Nota: Se tienen que usar Hooks para la resolución del ejercicio.

3- Desarrolle una aplicación utilizando Vite con React que incluya un formulario para ingresar una contraseña y su confirmación. Mientras el usuario modifica las contraseñas, el sistema debe verificar su validez según los siguientes requisitos:

- La contraseña debe tener al menos 8 caracteres.
- La contraseña debe incluir al menos un número.
- La contraseña debe incluir al menos un signo (@,!,$,%,&,#).
- La contraseña no debe contener espacios.
- Las contraseñas deben coincidir.

Debajo del formulario, muestre mensajes en elementos <p> (uno para cada requisito) que indiquen el cumplimiento o no de cada requisito. Los requisitos inválidos deben mostrarse en color rojo, mientras que los válidos deben aparecer en color verde. Por ultimo, agregar un boton que se habilita cuando todos los requisitos se cumplen y al hacer click avisa al usuario con un 'alert' que la contraseña es válida.

4- Desarrollar una aplicación empleando HTML + JS con el siguiente formulario:

- Un input de tipo texto con Label que diga "Animal".
- Un input de tipo numérico con Label que diga "Altura".
- Un botón que contenga el texto "Agregar".

Dicho formulario debería permitir lo siguiente:

a) Al presionar el botón "Agregar", se debería cargar dentro de un arreglo interno un nuevo objeto que contenga el nombre del animal y su altura. Verificar que el nombre del animal no este vacío y no se encuentre previamente cargado. También verificar que se ingresen alturas mayores a cero.

b) Una vez agregado el objeto al arreglo, blanquear los campos.

c) Mostrar dentro de una lista desordenada los nombres de los animales con sus respectivas alturasen el siguiente formato:

Jirafa (3m)

Elefante (2m)

Orangutan (1,3m)

d) Mostrar dentro de un span cuál es el animal más alto y cuál es el más bajo.

5- Desarrollar una aplicación utilizando React para convertir longitudes entre metros (M), pulgadas (P) y kilometros (KM). La interfaz de la aplicación debe incluir únicamente tres campos de entradas numéricas (<input type="number"/>) y etiqueta (<label>), uno para cada unidad de longitud. Los valores iniciales de estos campos deben ser 0.

Al modificar el valor de una de las entradas, las otras dos deben actualizarse automáticamente para reflejar la conversión correspondiente.

Ecuaciones de referencia:

- P = M * 39,37 (Metros a pulgadas)
- KM = M * 1000 (Metros a kilómetros)
- KM = P / 39370 (Pulgadas a kilómetros)

6- Realizar una aplicación en React que contenga el siguiente formulario:

- Un input de tipo texto con el label que diga "Ciudad".

- Un input de tipo number con el label que diga "Poblacion".

- Un botón que contenga el texto "Agregar".

Dicho formulario debería permitir lo siguiente:

a) Al presionar el botón "Agregar", se debería cargar en una lista un Objeto que contenga el Nombre de la Ciudad y su cantidad de población.

b) Una vez agregado el objeto a la lista, se debería blanquear los campos de Ciudad y Cantidad de Población.

c) Mostrar en un span la ciudad con la mayor población.

d) Mostrar dentro de un span cual es la población más próxima a la cantidad de pobladores de La Rioja (362605 pobladores).

Nota: Se tienen que usar Hooks para la resolución del ejercicio.

7- Realizar una aplicación en Vite con React que contenga el siguiente formulario:

- Un input de tipo texto con un label que diga "Producto".
- Un input de tipo number con el label que diga "Precio".
- Un botón que contenga el texto "Agregar".

Dicho formulario debería permitir lo siguiente:

a) Al presionar el botón "Agregar", se debería cargar en una lista un Objeto que contenga el Nombre del Producto y su Precio.

b) Una vez agregado el objeto a la lista, se debería blanquear los campos de Producto y Precio.

c) Mostrar en un span el nombre del producto mas caro.

d) Mostrar en un span el precio promedio de los productos en la lista.

Nota: Se tienen que usar Hooks para la resolución del ejercicio.

8- Desarrollar una aplicación utilizando React para convertir datos binarios entre bytes (B), mebibytes (MiB) y gibibytes (GiB). La interfaz de la aplicacion debe incluir únicamente tres campos de entradas numéricas (<input type="number"/>) y etiqueta (<label>), uno para cada unidad de datos.

Los valores iniciales de estos campos deben ser 0.

Al modificar el valor de una de las entradas, las otras dos deben actualizarse automáticamente para reflejar la conversión correspondiente.

Ecuaciones de referencia:

MiB = B * 1024 * 1024 (byte a mebibyte).

GiB = B * 1024 * 1024 * 1024 (byte a gibibyte).

GiB = MiB * 1024 (mebibyte a gibibyte).

9- Desarrollar una aplicación empleando HTML + JS con el siguiente formulario:

- Un input de tipo texto con Label que diga "Nombre".
- Un input de tipo numérico con Label que diga "Peso".
- Un botón que contenga el texto "Agregar".

Dicho formulario debería permitir lo siguiente:

a) Al presionar el botón "Agregar", se debería cargar dentro de un arreglo interno un nuevo objeto que contenga el nombre de una persona y su peso. Verificar que el nombre no este vacío y no se encuentre previamente cargado. También verificar que se ingresen alturas mayores a cero.

b) Una vez agregado el objeto al arreglo, blanquear los campos.

c) Mostrar dentro de una lista desordenada los nombres de las personas con sus respectivos pesos en el siguiente formato:

Pepe (82 kg)

Pedro (100 kg)

Maria (65 kg)

d) Mostrar dentro de un span cuál es la persona más liviana y cuál es el promedio de pesos.

10- Desarrolla una aplicación utilizando React con Vite para gestionar los productos de una ferretería. El programa debe cumplir con los siguientes requisitos:

- Listado de productos: Muestra una lista de productos con los siguientes datos: nombre, categoría y precio.
- Agregar nuevos productos:
  - Implementa un formulario para ingresar nuevos productos con las siguientes validaciones:
    - Nombre: Campo de texto que no puede estar vacio. No se pueden agregar productos con el mismo nombre en la misma categoria.
    - Categoría: Lista desplegable con tres opciones: "Electricidad", "Pintureria" y "Plomeria".
    - Precio: Campo numérico que debe ser mayor que cero.
  - Las validaciones se deben realizar al hacer click en "Agregar". Si algún criterio no se cumple, muestra una alerta al usuario. Después de agregar un producto, limpia los campos del formulario.
- Eliminar productos: cada producto en la lista debe tener un botón para eliminarlo. Antes de eliminar un producto, solicitar confirmación al usuario.
- Resumen por categoría: Debajo de la lista de productos, mostrar tres mensajes que indiquen la cantidad total de productos y el precio total por categoría.

11- Desarrolle una aplicación utilizando Vite con React que incluya un formulario para que el usuario ingrese los 3 lados de un triángulo. Permitir únicamente el ingreso de números positivos. Debajo del formulario indicar el perímetro del triángulo y si es equilátero, isósceles o escaleno.

12- Empleando HTML+JS, realice un programa para un viajero que permita agregar a un listado de tramos con nombre y distancia en kilómetros. Guardar el listado en un arreglo interno y mostrar el listado en pantalla dentro de un elemento <ul>. No permitir el ingreso de tramos con nombres vacíos o distancias negativas o cero. Luego de agregar un tramo limpiar las entradas. Al final del listado mostrar el promedio de distancias, el tramo más largo y el tramo más corto.

13- Desarrolle una aplicación de Vite con React de un contador que incremente o decremente un número del 0 al 10 cuando el usuario presione los botones de incrementar o decrementar. Empezar la cuenta en 5. Habilitar/Deshabilitar los botones para que la cuenta se mantenga dentro del rango. Agregar un botón para reiniciar la cuenta a 5.