Míni actividad formativa (no entrega puntos de RC). [Solución]

Revise el `ejemplo_5.html` y `programa_5.js`. Luego responda las siguientes preguntas:

a. ¿Qué está haciendo el código?

```
// Creamos una función que se encarga de actualizar el SVG según los datos que llegan.
function joinDeDatos(datos) {
  // Definimos el ancho y largo del SVG.
  svg.attr("width", 50 + datos.length * 100).attr("height", 500);

  // Vinculamos los datos con cada elemento rect
  const update = svg.selectAll("rect").data(datos);

  // Solicitamos la selección de rect que no tienen datos y los eliminamos.
  update.exit().remove();

  // Solicitamos la selección de datos que no tienen rect y les hacemos append de un rect
  const sinElementos = update.enter().append("rect");

  // Combinamos los rect existentes con los nuevos rect y los personalizamos según la
  // información de los datos.
  update
    .merge(sinElementos)
    .attr("width", 50)
    .attr("fill", "orange")
    .attr("height", (d) => d)
    .attr("x", (_, i) => 50 + i * 100);
}
```

Esta función se encarga de recibir una lista de números, actualiza el tamaño del SVG según cuantos datos llegan. Luego hace el *data join* con los elementos `rect` del SVG.

1. Aquellos elementos `rect` que se quedan sin datos asociados son eliminados.

2. Aquellos datos que quedan sin asignarse a un elemento, les creamos un rect (`append("rect")`).

3. A los datos que fueron correctamente vinculados (los update) y los nuevos datos (`sinElenentos`) los unificamos con merge y actualizamos la información de sus `rect` según el dato.

```
// Creamos una lista inicial de datos y llamamos a joinDeDatos
const datos = [10, 20, 30, 40];
joinDeDatos(datos);

// Buscamos el elemento HTML con id="boton"
const boton = document.getElementById("boton");

// Le agregamos un evento. Cada vez que le hagan click, se activa la función.
boton.addEventListener("click", () => {
  // Definimos un número aleatorio entre 1 y 11.
  const largo = Math.round(Math.random() * 10 + 1);

  // Agregamos números aleatorios entre 1 y 21 tantas veces según el
  // número definido anteriormente.
  const nuevos_datos = []
  for (let i = 0; i < largo; i++) {
    nuevos_datos.push(Math.round(Math.random() * 20 + 1))
  }

  // Actualizamos el SVG con esta nueva lista de datos.
  joinDeDatos(nuevos_datos);
});
```

Llamamos inicialmente a la función creada antes con los datos `[10, 20, 30, 40]`. Luego buscamos el botón en el DOM y vinculamos el evento "click" a una función.

Esta función se encarga de definir un número aleatorio entre 1 y 11 que corresponderá a cuantos datos inventar, luego llena una lista con datos aleatorios entre 1 y 21 hasta lograr la cantidad esperada. Finalmente se llama a la función que actualiza el SVG con la nueva lista de datos.

b. ¿Qué sucede si comentamos/eliminamos la línea 13?

La línea 13 corresponde a `update.exit().remove();`

Si esta línea es comentada, lo único que ocurre es que los elementos `rect` que quedaron sin datos asignados no serán eliminados del DOM.

Si uno inspecciona el SVG se podrá encontrar con más elementos `rect` qué datos usados en el *data joins*. Esos `rect` extras son los que debieron ser eliminados

c. ¿Qué sucede si comentamos/eliminamos las líneas 16 y 21?

La línea 16 y 21 son:

```
const sinElementos = update.enter().append("rect");

.merge(sinElementos)
```

Estas líneas se encargan de crear un `rect` para cada nuevo dato que no logró quedar asociados a un `rect` existente. Luego se encarga unificar el grupo `update` con el grupo `enter` para personalizar su información.

Si se comentan ambas líneas, nunca hacemos nada con el conjunto "`enter`" del *data join*. Por lo tanto, nunca se agregarían nuevos `rect` al SVG y solo se actualizarán los que ya existen.

Revise el `ejemplo_6.html` y `programa_6.js`. Luego responda las siguientes preguntas:

a. ¿Este código hace algo distinto al ejemplo 5?

No, hace exactamente lo mismo. LA diferencia es que ocupa el comando "join" en vez de `enter()` `exit()` y `merge()`

b. ¿Qué pasó con la parte de eliminar los elementos de la selección **`exit`**?

El comando join lo hace de forma automática, por lo cual no es necesario que nosotros accedamos a dicho grupo para eliminarlos.

c. ¿Por qué ya no hacemos **`merge`**?

El comando join se encarga de forma automática de retornar el merge `enter` el conjunto enter y update. Lo cual ya no es necesario que nosotros hagamos eso explícitamente.