Universidad Tecnológica Metropolitana

DIVISIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIÓN

División Tecnologías de la Información y Comunicación

Coordinación de Servicios de Tecnologías de la Información y Comunicación

Universidad Tecnológica Metropolitana

4F

ACTIVIDAD # 5-7

Nombre alumno:
Fernando David Sanchez Sacnhez

Nombre del profesor(a): Ruth Dominguez

Fecha de entrega: Mérida, Yucatán a martes 12 de octubre de 2024

Practica 5.

Variable listasPorLetra:

- listasPorLetra es un objeto que actúa como un "contenedor" para las listas enlazadas, cada una correspondiente a una letra del alfabeto. Este objeto almacena las listas enlazadas en las que cada palabra está clasificada según la primera letra.

```
const listasPorLetra = {};
```

Función agregarPalabra:

- **Objetivo:** Esta función permite agregar una palabra introducida por el usuario, la clasifica por la primera letra y la almacena en la lista enlazada correspondiente.

Pasos:

- Obtener la palabra ingresada por el usuario del campo de texto.
- Validar que no esté vacía.
- Obtener la primera letra de la palabra, convertirla a minúscula y verificar si ya existe una lista enlazada para esa letra.
- Si no existe, crear una nueva lista enlazada para esa letra.
- Agregar la palabra a la lista correspondiente.
- Limpiar el campo de entrada y actualizar la interfaz de usuario.

```
// Función para agregar una palabra y clasificarla según su primera letra
function agregarPalabra() {
  const palabra = document.getElementById('palabraInput').value.trim();
  if (palabra === '') {
    alert('Por favor ingresa una palabra.');
    return;
  }

  const primeraLetra = palabra[0].toLowerCase();
  if (!listasPorLetra[primeraLetra]) {
    listasPorLetra[primeraLetra] = new ListaEnlazada(); // Crear una nueva lista para
  }

  // Agregar la palabra a la lista correspondiente
  listasPorLetra[primeraLetra].agregarPalabra(palabra);

  // Limpiar el campo de entrada
  document.getElementById('palabraInput').value = '';

  // Actualizar la interfaz de usuario
  actualizarUI();
}
```

Función actualizarUI:

- **Objetivo:** Esta función se encarga de actualizar la interfaz de usuario para mostrar las palabras clasificadas según su primera letra. Cada vez que se agrega una palabra nueva, la interfaz se actualiza para reflejar los cambios.

Pasos:

- Limpia el contenido anterior de la interfaz.
- Itera sobre el objeto listasPorLetra, que contiene las listas clasificadas por letra.
- Por cada letra que tiene palabras, crea un nuevo elemento visual que incluye un título con la letra y una lista de palabras.
- Agrega esos elementos al contenedor principal.

```
function actualizarUI() {
  const contenedorListas = document.getElementById('contenedorListas');
  contenedorListas.innerHTML = ''; // Limpiar contenido anterior

  for (const letra in listasPorLetra) {
    const lista = listasPorLetra[letra].obtenerPalabras();
    if (lista.length > 0) {
      const listaElem = document.createElement('div');
      listaElem.classList.add('lista');

      const titulo = document.createElement('h3');
      titulo.textContent = `Palabras que empiezan con "${letra.toUpperCase()}"`;

      const ul = document.createElement('ul');
      lista.forEach(palabra => {
        const li = document.createElement('li');
        li.textContent = palabra;
        ul.appendChild(li);
      });

      listaElem.appendChild(titulo);
      listaElem.appendChild(ul);
      contenedorListas.appendChild(listaElem);
    }
  }
}
```

Clase ListaEnlazada:

- **Objetivo:** La clase ListaEnlazada es la que gestiona la lista de nodos. Es responsable de agregar palabras, almacenarlas en la lista, y permitir que se puedan recuperar en orden.

Método agregarPalabra:

- **Propósito:** Agregar una palabra a la lista enlazada. Si la lista está vacía, el nodo se convierte en la cabeza de la lista. Si no está vacía, el método recorre la lista hasta encontrar el último nodo y agrega el nuevo nodo al final.

```
// Método para agregar palabras a la lista enlazada
agregarPalabra(palabra) {
  const nuevoNodo = new Nodo(palabra);
  if (!this.head) {
    this.head = nuevoNodo; // Si la lista está vacía, el nuevo nodo es la cabeza
  } else {
    let actual = this.head;
    while (actual.next) {
      actual = actual.next; // Recorrer la lista hasta el último nodo
    }
    actual.next = nuevoNodo; // Agregar el nuevo nodo al final
  }
}
```

Método obtenerPalabras:

- **Propósito:** Este método permite obtener todas las palabras almacenadas en la lista enlazada. Recorre la lista desde el principio hasta el final, agregando las palabras a un resultado.

```
// Método para obtener todas las palabras en la lista
obtenerPalabras() {
  let actual = this.head;
  const palabras = [];
  while (actual) {
    palabras.push(actual.data);
    actual = actual.next;
  }
  return palabras;
}
}
```

Practica 6.

Crear una instancia de la lista enlazada:

- Aquí se crea una nueva instancia de ListaEnlazada, la cual se utilizará para almacenar las letras de la palabra ingresada en orden inverso.

```
const lista = new ListaEnlazada();
```

Función invertirPalabra:

- **Objetivo:** Esta función invierte el orden de las letras de la palabra ingresada por el usuario y las almacena en una lista enlazada.

Pasos:

- Obtener la palabra ingresada por el usuario.
- Validar que no esté vacía.
- Limpiar la lista enlazada antes de procesar la nueva palabra.
- Recorrer cada letra de la palabra y agregarla al inicio de la lista enlazada para invertir el orden.
- Mostrar la palabra invertida en el elemento correspondiente de la interfaz.

```
// Función para invertir la palabra ingresada
function invertirPalabra() {
    const palabra = document.getElementById('palabra').value.trim();

    if (palabra === '') {
        alert('Por favor ingresa una palabra válida.');
        return;
    }

    // Limpiar la lista antes de procesar una nueva palabra
    lista.limpiar();

    // Agregar cada letra de la palabra a la lista enlazada en orden inverso
    for (let i = 0; i < palabra.length; i++) {
        lista.agregarAlInicio(palabra[i]);
    }

    // Mostrar la palabra invertida
    const resultadoElem = document.getElementById('resultado');
    resultadoElem.textContent = `Palabra invertida: ${lista.obtenerPalabraInvertida()}`;
}
```

Clase ListaEnlazada:

- **Objetivo:** La clase ListaEnlazada es responsable de gestionar los nodos de la lista. Se utiliza para agregar letras en orden inverso, recorrer la lista y construir la palabra invertida.

Método agregarAlInicio:

- **Propósito:** Este método agrega una letra al inicio de la lista enlazada. Esto permite que las letras se agreguen en orden inverso al ingresarlas una por una.

```
// Método para agregar un nodo al inicio de l
agregarAlInicio(data) {
    const nuevoNodo = new Nodo(data);
    if (!this.head) {
        this.head = nuevoNodo;
    } else {
        nuevoNodo.next = this.head;
        this.head = nuevoNodo;
    }
}
```

Método obtenerPalabraInvertida:

- **Propósito:** Este método recorre la lista enlazada, construyendo una cadena que contiene las letras en orden inverso.

```
// Método para obtener la palabra invertida como una cadena de texto
obtenerPalabraInvertida() {
    let actual = this.head;
    let palabraInvertida = '';
    while (actual) {
        palabraInvertida += actual.data; // Concatenar las letras en orden
        actual = actual.next;
    }
    return palabraInvertida;
}
```

Practica 7.

Importar la clase ListaEnlazada:

- Se importa la clase ListaEnlazada que será usada para almacenar las letras de la palabra y realizar las operaciones de verificación de palíndromo.

```
// Importar la clase ListaEnlazada
import { ListaEnlazada } from './lista.js';
```

Función verificarPalindromo:

- **Objetivo:** Esta función verifica si la palabra ingresada por el usuario es un palíndromo (se lee igual hacia adelante que hacia atrás).

Pasos:

- Obtener la palabra ingresada y convertirla a minúsculas para asegurar que la comparación no sea sensible a mayúsculas/minúsculas.
- Crear una instancia de ListaEnlazada para almacenar las letras de la palabra.
- Recorrer la palabra original, agregando cada letra a la lista enlazada.
- Obtener la palabra original directamente desde la lista enlazada.
- Invertir el orden de la lista y obtener la palabra invertida.
- Comparar la palabra directa con la palabra invertida.
- Mostrar el resultado en el DOM, indicando si es o no un palíndromo.

```
function verificarPalindromo() {
    const palabraOriginal = document.getElementById('palabra').value.trim().toLowerCase();
    const lista = new ListaEnlazada();

    // Agregar cada letra de la palabra a la lista enlazada
    for (let i = 0; i < palabraOriginal.length; i++) {
        lista.agregarNodo(palabraOriginal[i]);
    }

    // Obtener la palabra normal
    const palabraDirecta = lista.obtenerPalabra();

    // Invertir la lista
    lista.invertir();

    // Obtener la palabra invertida
    const palabraInvertida = lista.obtenerPalabra();

    // Comparar la palabra original con la palabra invertida
    const resultado = document.getElementById('resultado');
    if (palabraDirecta === palabraInvertida) {
        resultado.textContent = `"${palabraOriginal}" es un palíndromo.`;
        resultado.style.color = 'green';
    } else {
        resultado.textContent = `"${palabraOriginal}" no es un palíndromo.`;
        resultado.style.color = 'red';
    }
}
```

Clase ListaEnlazada:

- **Objetivo:** La clase ListaEnlazada es responsable de gestionar los nodos que contienen las letras de la palabra. Se encarga de agregar nodos, invertir la lista y devolver la palabra para verificar si es un palíndromo.

Método agregarNodo:

- **Propósito:** Este método agrega un nodo con la letra de la palabra al final de la lista enlazada.

```
// Método para agregar caracteres a la lista
agregarNodo(data) {
    const nuevoNodo = new Nodo(data);
    if (!this.head) {
        this.head = nuevoNodo; // Si la lista está vacía, el nuevo nodo es la
    } else {
        let actual = this.head;
        while (actual.next) {
            actual = actual.next; // Recorrer la lista hasta el último nodo
        }
        actual.next = nuevoNodo; // Agregar el nuevo nodo al final
    }
}
```

Método obtenerPalabra:

- **Propósito:** Este método recorre la lista enlazada para construir una palabra a partir de los nodos que contienen las letras. Se utiliza para obtener tanto la palabra original como la invertida.

```
obtenerPalabra() {
    let actual = this.head;
    let palabra = '';
    while (actual) {
        palabra += actual.data; // Construir la palabra concatenando
        actual = actual.next;
    }
    return palabra;
}
```

Método invertir:

- **Propósito:** Este método invierte el orden de los nodos en la lista enlazada, cambiando el puntero siguiente de cada nodo para invertir la secuencia de letras.

```
    invertir() {
        let anterior = null;
        let actual = this.head;
        let siguiente = null;

        while (actual) {
            siguiente = actual.next;
            actual.next = anterior; // Revertir el puntero
            anterior = actual;
            actual = siguiente;
        }
        this.head = anterior; // Actualizar la cabeza de la lista
    }
}
```