Evaluación del módulo [Actividad Evaluada]

Evaluación del módulo

Objetivo general
Desarrollar un simulador interactivo por código JavaScript que permita simular una misión de exploración galáctica. El sistema debe administrar tripulantes, recursos, decisiones lógicas y registros de datos utilizando variables, estructuras de control, funciones, objetos y arreglos. El alumno debe entregar un archivo **.js** que contenga todo el código necesario para ejecutar esta simulación desde Node.js o desde un entorno online como repl.it o JSFiddle.

Descripción del reto
Construir un programa que simule una misión de exploración galáctica. El usuario podrá asignar datos de misión, registrar tripulantes, gestionar recursos, tomar decisiones, y consultar reportes mediante interacciones por consola (usando **console.log()** y **prompt()** si desea).

Fases de desarrollo

Fase 1: Configuración de misión

- Declarar variables para el nombre de la nave (tipo string), distancia estimada (number), estado de la misión (boolean) y recursos disponibles (objeto con propiedades como agua, comida y energía).
- Crear un objeto **nave** que contenga las propiedades generales de la misión: nombre, modelo, tripulación (como un arreglo vacío), y estado de la misión.
- Incluir métodos dentro del objeto como **mostrarEstado()** y **reportarRecursos()** para imprimir mensajes por consola.
- Utilizar el objeto **Math** para generar cantidades aleatorias de recursos iniciales.

Fase 2: Registro de tripulantes

- Crear una función para registrar tripulantes en un arreglo llamado tripulacion, cada uno representado por un objeto con nombre, rol y nivel de salud (de 0 a 100).
- Crear funciones auxiliares como **agregarTripulante(nombre, rol)** y **mostrarTripulacion()** que permitan administrar la tripulación.

Fase 3: Simulación de eventos

- Simular un menú de opciones con una variable opcion y estructuras condicionales if o switch para ejecutar distintas acciones:
  - Explorar (reduce recursos y puede afectar salud)
  - Comer (reduce comida, mejora salud)
  - Descansar (recupera salud, consume tiempo)
  - Reportar estado (muestra un resumen de variables)

- Usar ciclos como **while** o **for** para representar días o turnos de la misión. Cada turno debe modificar los recursos y aplicar lógica condicional.
- Implementar condiciones de borde como evitar que los recursos bajen de 0 o que la salud supere 100.

Fase 4: Reportes y lógica avanzada

- Crear funciones que muestren:
  - El promedio de salud de la tripulación
  - La cantidad de tripulantes con salud menor a 50
  - El estado de los recursos

- Usar ciclos, acumuladores y operadores para realizar estos cálculos.
- Emplear concatenación de strings y métodos del objeto **String** para mejorar los mensajes por consola.

Fase 5: Código limpio y buenas prácticas

- El código debe estar dividido en funciones con responsabilidades claras.
- Las variables deben tener nombres significativos.
- Se deben seguir las convenciones del lenguaje y evitar el uso innecesario de variables globales.
- Comentar el código donde sea necesario para describir partes importantes del programa.

Entregable

- Un archivo llamado **simuladorMision.js** que contenga todo el código comentado y organizado.
- El código debe poder ejecutarse sin errores desde un entorno de JavaScript puro, sin depender de HTML.

Estado de la entrega

Estado de la entrega	Todavía no se han realizado envíos
Estado de la calificación	Sin calificar
Última modificación	-

DESARROLLO EVALUACIÓN DEL MÓDULO
Simular una misión de exploración galáctica

DESARROLLO CODIGO:

```
//  OBJETIVO RECIBIR VALORES DE VARIABLES POR INTERACCIÓN DE CONSOLA */
import * as readline from 'node:readline/promises';
import { stdin as input, stdout as output } from 'node:process';

async function leerDelUsuario(pregunta){
    const rl = readline.createInterface({ input, output });
    const respuesta = await rl.question(pregunta);
    rl.close();
    return respuesta;
}

// ------------------------------------
// Fase 1: Configuración de misión
// ------------------------------------
// Variables básicas
let nombreNave = await leerDelUsuario("Ingrese el nombre de la nave: ");
let distanciaEstimada = await leerDelUsuario("Ingrese la distancia estimada: ");
let estadoMision = true;

// Recursos iniciales con valores aleatorios
let recursosDisponibles = {
     agua: Math.floor(Math.random() * 100) + 50,
     comida: Math.floor(Math.random() * 100) + 50,
     energia: Math.floor(Math.random() * 100) + 50
};

// Objeto nave con propiedades y métodos
let nave = {
     nombre: nombreNave,
     modelo: "X-Galaxy",
     tripulacion: [],
     estado: estadoMision,
     mostrarEstado: function () {
         console.log(`Estado de la nave: ${this.nombre} (${this.modelo})`);
         console.log(`Misión           : ${this.estado}`);
         console.log(`Tripulantes      : ${this.tripulacion.length}`);
     },
```

```
    reportarRecursos: function () {
        console.log("Recursos Disponibles:");
        console.log(`Agua   : ${recursosDisponibles.agua}`);
        console.log(`Comida : ${recursosDisponibles.comida}`);
        console.log(`Energía: ${recursosDisponibles.energia}`);
    }
};

// ------------------------------------------
// Fase 2: Registro de tripulantes
// ------------------------------------------
// Crear una función para registrar tripulantes
function agregarTripulante(nombre, rol) {
     let tripulante = {
         nombre: nombre,
         rol: rol,
         salud: Math.floor(Math.random() * 101),
     };
     nave.tripulacion.push(tripulante);
     console.log(`Tripulante agregado: ${nombre} (${rol})`);
}

// Muestra la tripulación
function mostrarTripulacion() {
     console.log("---------------------------------------------");
     console.log("Mostrar Tripulación:");
     console.log("---------------------------------------------");
     if (nave.tripulacion.length === 0) {
          console.log("No hay tripulantes.");
     } else {
       nave.tripulacion.forEach((t, index) => {
          console.log(`${index + 1}. ${t.nombre} - ${t.rol} - Salud: ${t.salud}`);
       });
     }
     console.log("-------------------------------------------------------------");
}
```

```
// ------------------------------------------
// Fase 3: Simulación de eventos
// ------------------------------------------
// Simular un menú de opciones con una variable
async function iniciarMision() {
     let turno = 1;
     while (nave.estado) {
         console.log(`Turno ${turno}: Seleccione una opción:`);
         console.log("1. Explorar");
         console.log("2. Comer");
         console.log("3. Descansar");
         console.log("4. Reportar estado");
         console.log("5. Finalizar misión");
         let opcion = await leerDelUsuario("Ingresa una opción (1-5): ");
         switch (opcion) {
            case "1":
                explorar();
                break;
            case "2":
                comer();
                break;
            case "3":
                descansar();
                break;
            case "4":
                nave.mostrarEstado();
                nave.reportarRecursos();
                mostrarTripulacion();
                break;
            case "5":
                console.log("Finalizando la misión...");
                nave.estado = false;
                break;
            default:
                console.log("Opción no válida. Intenta nuevamente.");
         }
         // Verificar condiciones críticas
         if (recursosDisponibles.agua <= 0 || recursosDisponibles.comida <= 0 ||
recursosDisponibles.energia <= 0) {
            console.log("La nave se ha quedado sin recursos. La misión ha fallado.");
            nave.estado = false;
         }
         turno++;
     }
}
```

```
// Explorar (reduce recursos y puede afectar salud)
function explorar() {
     console.log("-----------------------------------------------");
     console.log("OPCIÓN 1: Explorando el espacio...");
     console.log("-----------------------------------------------");
     recursosDisponibles.energia -= 15;
     recursosDisponibles.agua -= 5;
     recursosDisponibles.comida -= 5;

     nave.tripulacion.forEach(t => {
         t.salud -= Math.floor(Math.random() * 10);
         if (t.salud < 0) t.salud = 0;
    });
    console.log("Exploración completada. Recursos y salud afectados.");
    console.log("-------------------------------------------------");
}

// Comer (reduce comida, mejora salud)
function comer() {
     if (recursosDisponibles.comida >= nave.tripulacion.length * 5) {
         recursosDisponibles.comida -= nave.tripulacion.length * 5;
         nave.tripulacion.forEach(t => {
             t.salud += 10;
             if (t.salud > 100) t.salud = 100;
         });
         console.log("-----------------------------------------------");
         console.log("OPCIÓN 2: Reduce comida y recuperando salud.");
         console.log("-----------------------------------------------");
     } else {
         console.log("-----------------------------------------------");
         console.log("OPCIÓN 2: No hay suficiente comida.");
         console.log("-----------------------------------------------");
     }
}

// Descansar (recupera salud, consume tiempo)
function descansar() {
     console.log("-----------------------------------------------");
     console.log("OPCIÓN 3: La tripulación ha descansado. Salud recuperada.");
     console.log("-----------------------------------------------");
     recursosDisponibles.energia -= 10;
     nave.tripulacion.forEach(t => {
         t.salud += 5;
         if (t.salud > 100) t.salud = 100;
     });
}
```

```
// ------------------------------------
// Fase 4: Reportes y lógica avanzada
// ------------------------------------
async function InformesFinales() {
    let continuar = true;

    while (continuar) {
        console.log("-----------------------------------------------");
        console.log("Reportes y lógica avanzada de la Misión:");
        console.log("-----------------------------------------------");
        console.log("1. Promedio de salud de la tripulación");
        console.log("2. Ver tripulantes con salud menor a 50");
        console.log("3. Mostrar estado final de recursos");
        console.log("4. Finalizar");

        let opcion = await leerDelUsuario("Seleccione una opción (1-4): ");
        switch (opcion.trim()) {
            case "1":
                let promedio = promedioSalud();
                console.log("-----------------------------------------------");
                console.log(`OPCIÓN 1: Promedio de salud tripulación: ${promedio}`);
                console.log("-----------------------------------------------");
                break;
            case "2":
                let criticos = tripulantesConSaludMenor();
                console.log("-----------------------------------------------");
                console.log(`OPCIÓN 1: Tripulantes con salud menor a 50: ${criticos}`);
                console.log("-----------------------------------------------");
                break;
            case "3":
                estadoFinalMision();
                break;
            case "4":
                console.log("Informes finalizados.");
                continuar = false;
                break;
            default:
                console.log("Opción no válida. Intenta nuevamente.");
        }
    }
}

// El promedio de salud de la tripulación
function promedioSalud() {
     if (nave.tripulacion.length === 0) return 0;
     let totalSalud = 0;
     for (let trip of nave.tripulacion) {
         totalSalud += trip.salud;
     }
     return (totalSalud / nave.tripulacion.length).toFixed(2);
}
```

```
// La cantidad de tripulantes con salud menor a 50
function tripulantesConSaludMenor() {
     let criticos = nave.tripulacion.filter(trip => trip.salud < 50);
     return criticos.length;
}

// El estado de los recursos|
function estadoFinalMision() {
     console.log("-----------------------------------------------");
     console.log("Estado final de recursos:");
     nave.reportarRecursos();
     console.log("-----------------------------------------------");
}

// ----------------------------------------
// Fase 5: Código limpio y buenas prácticas
// ----------------------------------------
// Agregar algunos tripulantes de prueba
console.log("-----------------------------------------------");
console.log("AGREGANDO TRIPULANTES");
console.log("-----------------------------------------------");
agregarTripulante("Rodrigo", "Comandante");
agregarTripulante("Lisandro", "Ingeniero");
agregarTripulante("Victor", "Médico");


// Iniciar misión
console.log("-----------------------------------------------");
console.log("INICIO MISIÓN");
console.log("-----------------------------------------------");
await iniciarMision();

// Reportes finales
console.log("-----------------------------------------------");
console.log("REPORTES FINALES");
console.log("-----------------------------------------------");
await InformesFinales();
```

RESULTADOS:

PS C:\TD_JS0081\modulo_3\evaluacion_modulo> node simuladorMision.js

Ingrese el nombre de la nave: **GALAXY**
Ingrese la distancia estimada: **122**

AGREGANDO TRIPULANTES

Tripulante agregado: **Rodrigo (Comandante)**
Tripulante agregado: **Lisandro (Ingeniero)**
Tripulante agregado: **Víctor (Médico)**

INICIO MISIÓN

Turno 1: Seleccione una opción:
1. Explorar
2. Comer
3. Descansar
4. Reportar estado
5. Finalizar misión
Ingresa una opción (1-5): 1

OPCIÓN 1: Explorando el espacio...

Exploración completada. Recursos y salud afectados.

Turno 2: Seleccione una opción:
1. Explorar
2. Comer
3. Descansar
4. Reportar estado
5. Finalizar misión
Ingresa una opción (1-5): 2

OPCIÓN 2: Reduce comida y recuperando salud.

Turno 3: Seleccione una opción:
1. Explorar
2. Comer
3. Descansar
4. Reportar estado
5. Finalizar misión
Ingresa una opción (1-5): 3

OPCIÓN 3: La tripulación ha descansado. Salud recuperada.

Turno 4: Seleccione una opción:
1. Explorar
2. Comer
3. Descansar
4. Reportar estado
5. Finalizar misión
Ingresa una opción (1-5): 4

OPCIÓN 4:
Estado de la nave: GALAXY (X-Galaxy)
Misión : true
Tripulantes : 3

Recursos Disponibles:
Agua : 74
Comida: 117
Energía: 108

Mostrar Tripulación:
---
1. Rodrigo - Comandante - Salud: 71
2. Lisandro - Ingeniero - Salud: 86
3. Víctor - Médico - Salud: 48

Turno 5: Seleccione una opción:
1. Explorar
2. Comer
3. Descansar
4. Reportar estado
5. Finalizar misión
Ingresa una opción (1-5): 5
Finalizando la misión...

REPORTES FINALES

Reportes y lógica avanzada de la Misión:
---
1. Promedio de salud de la tripulación
2. Ver tripulantes con salud menor a 50
3. Mostrar estado final de recursos
4. Finalizar
Seleccione una opción (1-4): 1

OPCIÓN 1: Promedio de salud tripulación: 68.33

Reportes y lógica avanzada de la Misión:
---
1. Promedio de salud de la tripulación
2. Ver tripulantes con salud menor a 50
3. Mostrar estado final de recursos
4. Finalizar
Seleccione una opción (1-4): 2

OPCIÓN 2: Tripulantes con salud menor a 50: 1

Reportes y lógica avanzada de la Misión:
---
1. Promedio de salud de la tripulación
2. Ver tripulantes con salud menor a 50
3. Mostrar estado final de recursos
4. Finalizar
Seleccione una opción (1-4): 3

OPCIÓN 3: Estado final de recursos:

Recursos Disponibles:
Agua : 74
Comida: 117
Energía: 108

Reportes y lógica avanzada de la Misión:
---
1. Promedio de salud de la tripulación
2. Ver tripulantes con salud menor a 50
3. Mostrar estado final de recursos
4. Finalizar
Seleccione una opción (1-4): 4

Informes finalizados.

PS C:\TD_JS0081\modulo_3\evaluacion_modulo>