ACTIVIDAD PARA NAVIDADES – MÓDULO DE PROGRAMACIÓN

Juego del Laberinto

Desarrolla un programa en Java que simule un **juego de texto interactivo** en el que un jugador debe navegar por un laberinto compuesto por 4 salas conectadas entre sí. El objetivo del juego es llegar a la **Sala 3** con la mayor cantidad de energía posible antes de quedarse sin energía.

Reglas del Juego

1. **Estado Inicial del Jugador**:
    - El jugador comienza con **100 puntos de energía**.
    - La posición inicial es la **Sala 1**.
2. **Conexiones entre Salas**:
    - Las salas están conectadas de la siguiente manera:
        - **Sala 1 → Sala 2, Sala 3**
        - **Sala 2 → Sala 1, Sala 4**
        - **Sala 4 → Sala 1, Sala 2**
3. **Opciones del Jugador**: En cada turno, el jugador tiene las siguientes opciones:
    - **Moverse a otra sala**: Elige la sala a la que desea ir, siempre y cuando exista una conexión válida desde su posición actual.
    - **Inspeccionar la sala actual**: En esta acción, puede ocurrir un evento aleatorio:
        - Encontrar monedas (**+10 puntos de energía**).
        - Caer en una trampa (**-15 puntos de energía**).
        - No encontrar nada.

- **Salir del juego**: El jugador puede decidir abandonar el juego en cualquier momento.

Requisitos del Programa

1. **Estructura del Código**:
   - Usa un **bucle while** para mantener el juego activo hasta que el jugador gane, pierda o decida salir.
   - Usa **switch** para manejar las opciones del menú y los eventos aleatorios.
   - Usa **hasNextInt** para validar que las entradas del usuario sean correctas.
2. **Flujo del Juego**:
   - Muestra al jugador su **posición actual** y la **cantidad de energía restante** al inicio de cada turno.
   - Presenta un **menú interactivo** con las tres opciones (moverse, inspeccionar o salir).
   - Verifica que el movimiento entre salas sea válido según las conexiones.
3. **Mensajes en la Consola**:
   - Muestra mensajes claros que indiquen:
     - Los eventos que ocurren, por ejemplo,"¡Has encontrado monedas! Ganas 10 puntos de energía.".
     - Cambios en la energía del jugador.
     - Si el jugador ha ganado, perdido o salido del juego.
4. **Eventos Aleatorios**: Cada vez que el jugador inspecciona una sala, se genera un evento aleatorio. Los eventos posibles son:
   - **Evento 1**: El jugador encuentra monedas. Esto incrementa la energía en **10 puntos**.
   - **Evento 2**: El jugador cae en una trampa. Esto reduce la energía en **15 puntos**.
   - **Evento 3**: La sala está vacía y no sucede nada.

5. **Condiciones para Ganar o Perder**:
    - El jugador **gana** al llegar a la **Sala 3.**
    - El jugador **pierde** si su energía llega a **0 puntos** antes de llegar a la Sala 3.

Ampliación: Laberinto con energía variable

En esta ampliación, cada sala tiene un nivel de **energía variable** asignado aleatoriamente al inicio del juego. Los valores de energía de cada sala se almacenan en un **array fijo de tamaño 4** (porque hay 4 salas). Este array representa el costo o la ganancia de energía al entrar a cada sala.

Detalles del Ejercicio

1. **Asignación Inicial de Energía**:
    - Cada sala tiene un valor de energía aleatorio entre **-20** y **+20**.
    - Estos valores se generan al inicio del juego y se guardan en un array llamado energiaSalas.
2. **Efecto al Moverse**:
    - Cuando el jugador entra a una sala, su nivel de energía cambia según el valor correspondiente en el array energiaSalas.
    - Por ejemplo:
        - Si energiaSalas[1] = -15, el jugador pierde 15 puntos al entrar a la Sala 2.
        - Si energiaSalas[2] = +10, el jugador gana 10 puntos al entrar a la Sala 3.
3. **Visualización de la Energía de las Salas**:
    - El jugador no conoce los valores exactos de energía de cada sala al inicio del juego.
    - Puede optar por usar una acción especial llamada "inspeccionar salas", que revela los valores de energía del array.
4. **Objetivo del Juego**:
    - El jugador debe llegar a la **Sala 3** con al menos **30 puntos de energía** restantes para ganar.
    - Si su energía llega a 0 o menos, pierde.