Tarea 1

Esta tarera forma parte de un repositorio Github. El repositorio contiene un proyecto BlueJ en Java para la aplicación Buscaminas.

El Buscaminas es un juego solitario de ingenio, popularizado por los sistemas operativos Microsoft Windows. El juego presenta un tablero o campo minado, que consiste en una cuadrícula de celdas, donde las minas se colocan en ubicaciones o coordenadas aleatorias. Las dimensiones de la cuadrícula varían según los niveles del juego, siendo por ejemplo el nivel principiante, una cuadrícula de tamaño 9×9, con 10 minas en ubicaciones aleatorias.

Inicialmente, todas las celdas están *cerradas*, lo que significa que el jugador no sabe si una celda contiene una mina o no. El jugador *abre* una celda haciendo clic en ella: si una celda abierta contiene una mina, el jugador pierde. Si en cambio la celda no contiene una mina, mostrará un número que indica cuántas ubicaciones vecinas (adyacentes horizontal, vertical o diagonalmente) contienen minas. Además, si el número de ubicaciones minadas vecinas resulta ser cero, todas las celdas vecinas se abrirán automáticamente también, creando un efecto en cascada. La información sobre el número de ubicaciones minadas vecinas permite al jugador decidir, mediante razonamiento lógico, qué celdas contienen o no minas. El jugador gana cuando todas las celdas que no contienen minas están abiertas, es decir, las únicas celdas cerradas que quedan son las que contienen minas.

Además de abrir celdas, el jugador también puede interactuar con el juego *bloqueando* o *desbloqueando* celdas cerradas; una celda bloqueada es una celda que el jugador sospecha que contiene una mina; cuando está bloqueada, la celda no se puede abrir, evitando que se abra accidentalmente.

El proyecto *Buscaminas* es una implementación basada en texto del juego Buscaminas. Consta de cinco clases:

- `CeldaBuscaminas`: es la más simple de las clases. Modela una celda del buscaminas, con todos sus posibles estados: abierta o cerrada, con mina o sin mina, bloqueada o desbloqueada. Por supuesto, no todas las configuraciones son posibles (abierta y bloqueada no está permitido, por ejemplo), lo que debe garantizarse mediante los métodos públicos de la clase.
- `EstadoJuegoBuscaminas`: modela el tablero mediante un arreglo bidimensional de celdas, es decir de objetos CeldaBuscaminas. Contiene métodos para inicializar el juego, colocar minas en lugares aleatorios, y luego abrir celdas en ubicaciones específicas (con el correspondiente efecto en cascada), bloquearlas y desbloquearlas, etc.
- `LectorEntrada`: implementa la lectura de comandos de entrada y coordenadas del usuario, verificando que el comando y las coordenadas sean válidos (similar a otros proyectos BlueJ).
- `Buscaminas`: la clase que integra la implementación del juego. Su constructor crea el objeto lector de entrada y el estado del juego, y permite iniciar el juego llamando al método `startGame()`.
- `PrincipalBuscaminas`: Clase que contiene un método `main`, y permite crear el juego. Especialmente útil si se quiere compilar y ejecutar el juego desde línea de comandos (fuera de BlueJ).

La resolución de este trabajo práctico consiste en implementar varios métodos:

- De la clase `EstadoJuegoBuscaminas`:
  - `EstadoJuegoBuscaminas()` (el constructor). Está parcialmente implementado. Su tarea es completar la implementación llenando de celdas la matriz tablero.
  - `abrir(fila, col)`. Es una de las funcionalidades principales del juego. Abre una celda cerrada e implementa también el efecto en cascada. Debe verificar, después de abrir, si el juego ha terminado o no, por ejemplo, al abrir una celda con mina, o al abrir todas las celdas sin minas.

  - `contarMinasVecinas(fila, col)`. Método auxiliar que calcula el número de vecinos de una ubicación que contienen minas. Es útil para mostrar el número correspondiente cuando se abre una celda.
  - `abrirTodasLasCeldas()`. Abre todas las celdas del tablero. Se llama cuando termina el juego, para mostrar cómo estaba configurado el tablero.
  - `gcontarCeldasCerradas()`. Cuenta el número de celdas cerradas que quedan en el tablero. Puede ser útil para determinar si el juego ha terminado o no.

- De la clase `CeldaBuscaminas`:
  - `abrir()`. Abre una celda cerrada y desbloqueada.
  - `bloquear()`. Bloquea una celda cerrada y desbloqueada.
  - `desbloquear()`. Desbloquea una celda cerrada y bloqueada.

Su tarea es implementar todos los métodos mencionados. Pueden decidir agregar métodos auxiliares (privados), o implementar las funcionalidades requeridas dentro de los cuerpos de los métodos proporcionados. No está permitido cambiar las APIs de las clases, es decir, no se permite cambiar nombres de métodos, número de parámetros o sus tipos. No deben modificar otros métodos existentes en la implementación. Recomendamos estudiar primero el proyecto para entender cómo está organizado y cómo se distribuyen las funcionalidades en las cinco clases diferentes, antes de comenzar con la implementación.

Su solución debe contener implementaciones de todos los métodos. Deben enviar la solución a través de github, simplemente realizando *commit* y *push* de sus implementaciones en el repositorio correspondiente al grupo. Este trabajo práctico se resuelve de manera grupal, en grupos de hasta 5 personas. La corrección de la solución es importante, al igual que la calidad del código (claridad, legibilidad, estructura, etc.).