```
#ifndef TABLERO_H
#define TABLERO_H

#include <iostream>
#include <vector>
#include "Celda.h"

using namespace std;

/**
 * Clase que representa un tablero de juego.
 * Se encarga de gestionar la estructura del tablero, el modo desarrollador y las minas.
 */
class Tablero
{
   public:
      /**
       * Constructor por defecto de la clase Tablero.
       */
      Tablero();

      /**
       * Constructor con parámetros.
       * @param alturaTablero Altura del tablero.
       * @param anchoTablero Ancho del tablero.
       * @param modoDesarrollador Indica si está activado el modo desarrollador.
       */
      Tablero(int alturaTablero, int anchoTablero, bool modoDesarrollador);
```

```
/**
 * Obtiene la altura del tablero.
 * @return Altura del tablero.
 */
int getAlturaTablero();

/**
 * Establece la altura del tablero.
 * @param alturaTablero Nueva altura del tablero.
 * @return Altura actualizada del tablero.
 */
int setAlturaTablero(int alturaTablero);

/**
 * Obtiene el ancho del tablero.
 * @return Ancho del tablero.
 */
int getAnchoTablero();

/**
 * Establece el ancho del tablero.
 * @param anchoTablero Nuevo ancho del tablero.
 * @return Ancho actualizado del tablero.
 */
int setAnchoTablero(int anchoTablero);

/**
 * Obtiene el estado del modo desarrollador.
 * @return true si el modo desarrollador está activado, false en caso contrario.
```

```
 */
bool getModoDesarrollador();

/**
 * Establece el modo desarrollador.
 * @param modoDesarrollador Nuevo estado del modo desarrollador.
 * @return true si el modo desarrollador fue activado, false si fue desactivado.
 */
bool setModoDesarrollador(bool modoDesarrollador);

/**
 * Imprime el separador de encabezado del tablero.
 */
void imprimirSeparadorEncabezado();

/**
 * Imprime el separador de las filas del tablero.
 */
void imprimirSeparadorFilas();

/**
 * Imprime el encabezado del tablero.
 */
void imprimirEncabezado();

/**
 * Imprime el tablero en consola.
 */
void imprimir();
```

```
    /**
     * Coloca una mina en una celda específica.
     * @param x Coordenada en el eje X.
     * @param y Coordenada en el eje Y.
     * @return true si la mina fue colocada correctamente, false si la celda ya tenía una mina.
     */
    bool colocarMina(int x, int y);

    /**
     * Descubre si hay una mina en una celda específica.
     * @param x Coordenada en el eje X.
     * @param y Coordenada en el eje Y.
     * @return true si se descubre una mina, false si no hay mina en la celda.
     */
    bool descubrirMina(int x, int y);

    /**
     * Cuenta cuántas celdas no tienen mina y aún no han sido descubiertas.
     * @return Número de celdas sin mina y sin descubrir.
     */
    int contarCeldasSinMinasYSinDescubrir();

protected:

private:
    int alturaTablero, anchoTablero; ///< Dimensiones del tablero.
    bool modoDesarrollador; ///< Indica si el modo desarrollador está activado.
    vector<vector<Celda>> contenidoTablero; ///< Matriz de celdas que conforman el tablero.
```

```
    /**
     * Obtiene la representación de la mina en una celda específica.
     * @param x Coordenada en el eje X.
     * @param y Coordenada en el eje Y.
     * @return Representación de la mina (según el estado del juego).
     */
    string getRepresentacionMina(int x, int y);

    /**
     * Calcula el número de minas cercanas a una celda específica.
     * @param fila Índice de la fila de la celda.
     * @param columna Índice de la columna de la celda.
     * @return Número de minas cercanas a la celda dada.
     */
    int minasCercanas(int fila, int columna);
};

#endif // TABLERO_H
```

```
#ifndef JUEGO_H
#define JUEGO_H

#include "Tablero.h"

/**
 * Clase que representa el juego principal.
 * Se encarga de manejar la lógica del juego, la colocación de minas y la interacción con el usuario.
 */
class Juego
{
private:
    Tablero tablero; ///< Tablero del juego.
    int cantidadMinas; ///< Cantidad de minas en el juego.

    /**
     * Genera un número aleatorio dentro de un rango dado.
     * @param minimo Valor mínimo del rango.
     * @param maximo Valor máximo del rango.
     * @return Número aleatorio dentro del rango especificado.
     */
    int aleatorio_en_rango(int minimo, int maximo);

    /**
     * Obtiene una fila aleatoria dentro del tablero.
     * @return Índice de una fila aleatoria.
     */
    int filaAleatoria();
```

```
    /**
     * Obtiene una columna aleatoria dentro del tablero.
     * @return Índice de una columna aleatoria.
     */
    int columnaAleatoria();

public:
    /**
     * Constructor de la clase Juego.
     * @param tablero Objeto Tablero donde se desarrollará el juego.
     * @param cantidadMinas Número de minas a colocar en el tablero.
     */
    Juego(Tablero tablero, int cantidadMinas);

    /**
     * Coloca minas aleatoriamente en el tablero.
     */
    void colocarMinasAleatoriamente();

    /**
     * Solicita al usuario ingresar una fila.
     * @return Índice de la fila seleccionada por el usuario.
     */
    int solicitarFilaUsuario();

    /**
     * Solicita al usuario ingresar una columna.
     * @return Índice de la columna seleccionada por el usuario.
     */
```

```
    int solicitarColumnaUsuario();

    /**
     * Verifica si el jugador ha ganado la partida.
     * @return true si el jugador gana, false en caso contrario.
     */
    bool jugadorGana();

    /**
     * Inicia la partida del juego, manejando el flujo de juego e interacción con el usuario.
     */
    void iniciar();

    /**
     * Dibuja la portada del juego a partir de un archivo de texto.
     * @param nombreArchivo Nombre del archivo que contiene la portada del juego.
     */
    void dibujarPortada(string nombreArchivo);
};

#endif // JUEGO_H
```