Universidad Mariano Gálvez

Armando José De León Ovando 9959-24-155

Ing. Esduardo Del Aguila

Programación I

celda.cpp

```
#include "Celda.h" // Incluye el encabezado de la clase Celda
#include <iostream> // Biblioteca para operaciones de entrada/salida
using namespace std;

// Constructor por defecto de la clase Celda
Celda::Celda()
{
}

// Constructor parametrizado que inicializa las coordenadas y el estado de la mina
Celda::Celda(int coordenadaX, int coordenadaY, bool estadoMina)
{
   this->coordenadaX = coordenadaX; // Inicializa la coordenada X
   this->coordenadaY = coordenadaY; // Inicializa la coordenada Y
   this->mina = estadoMina; // Define si la celda contiene una mina
   this->minaDescubierta = false; // Por defecto, la mina no está descubierta
}

// Método para establecer la coordenada X
int Celda::setCoordenadaX(int coordenadaX)
{
```

```
    this->coordenadaX = coordenadaX;
}

// Método para obtener la coordenada X
int Celda::getCoordenadaX()
{
    return this->coordenadaX;
}

// Método para establecer la coordenada Y
int Celda::setCoordenadaY(int coordenadaY)
{
    this->coordenadaY = coordenadaY;
}

// Método para obtener la coordenada Y
int Celda::getCoordenadaY()
{
    return this->coordenadaY;
}

// Método para establecer si la celda contiene una mina
// Si ya hay una mina, devuelve false; de lo contrario, establece el estado y devuelve true
bool Celda::setMina(bool estadoMina)
{
    if (this->getMina())
    {
        return false; // No se puede cambiar el estado si ya hay una mina
    }
```

```
    else
    {
        this->mina = estadoMina; // Actualiza el estado de la mina
        return true; // Confirmación de que el cambio fue exitoso
    }
}

// Método para obtener el estado de la mina (true si hay una mina, false si no)
bool Celda::getMina()
{
    return this->mina;
}

// Método para establecer si la mina ha sido descubierta
bool Celda::setMinaDescubierta(bool minaDescubierta)
{
    this->minaDescubierta = minaDescubierta;
}

// Método para verificar si la mina ha sido descubierta
bool Celda::getMinaDescubierta()
{
    return this->minaDescubierta;
}

// Método para imprimir información sobre la celda
void Celda::imprimirCelda()
{
    cout << "Celda en " << this->coordenadaX << ", " << this->coordenadaY
```

```
        << " con mina? " << (this->mina ? "Sí" : "No") << "\n"; // Imprime si la celda tiene una mina
}
```

Config.cpp

```
#include <iostream> // Biblioteca para entrada y salida estándar
#include <unistd.h> // Biblioteca para funciones del sistema (posiblemente usada en Linux)
#include "Config.h" // Inclusión del archivo de encabezado de la clase Config

using namespace std;

// Constructor de la clase Config
Config::Config(int filasTablero, int columnasTablero, int minasTablero, bool
modoDesarrolladorTablero, int vidasTablero)
{
    // Inicializa los atributos con los valores recibidos
    this->filasTablero = filasTablero;
    this->columnasTablero = columnasTablero;
    this->minasTablero = minasTablero;
    this->modoDesarrolladorTablero = modoDesarrolladorTablero;
    this->vidasTablero = vidasTablero;
}

// Método para mostrar y modificar la configuración del juego
void Config::menuConfiguracion()
```

```
{
    int opciones; // Variable para almacenar la opción seleccionada por el usuario
    int valorIngresado; // Variable para almacenar el nuevo valor ingresado
    bool repetir = true; // Controla el bucle del menú

    do
    {
        system("cls"); // Limpia la pantalla (en Windows, usar "clear" en Linux)

        // Muestra el menú de configuración actual
        cout << "\n\n\t\tCONFIGURACION ACTUAL - Menu -" << endl;
        cout << "\t\t----------------------------" << endl;
        cout << "\t\t1. Filas del Tablero ----> " << this->getfilasTablero() << endl;
        cout << "\t\t2. Columnas del Tablero -> " << this->getcolumnasTablero() << endl;
        cout << "\t\t3. Minas del Tablero ----> " << this->getminasTablero() << endl;
        cout << "\t\t4. Modo del Juego -------> " << this->getmodoDesarrolladorTablero() << endl;
        cout << "\t\t5. Vidas del Jugador ----> " << this->getvidasTablero() << endl;
        cout << "\t\t6. Regresar al menu general" << endl;
        cout << "\n\t\tIngrese una opcion: ";
        cin >> opciones;

        // Si el usuario no eligió salir, pide un nuevo valor para actualizar la opción seleccionada
        if (opciones != 6)
        {
            cout << "\n\tIngrese el valor que desea cambiar: ";
            cin >> valorIngresado;
        }

        // Estructura switch para modificar la configuración en función de la opción seleccionada
```

```
switch (opciones)
{
   case 1:
      this->setfilasTablero(valorIngresado);
      cout << "Filas del Tablero actualizadas" << endl;
      break;

   case 2:
      this->setcolumnasTablero(valorIngresado);
      cout << "Columnas del Tablero actualizadas" << endl;
      break;

   case 3:
      this->setminasTablero(valorIngresado);
      cout << "Minas del Tablero actualizadas" << endl;
      break;

   case 4:
      this->setmodoDesarrolladorTablero(valorIngresado);
      cout << "Modo del Juego actualizado" << endl;
      break;

   case 5:
      this->setvidasTablero(valorIngresado);
      cout << "Vidas del Juego actualizadas" << endl;
      break;

   case 6:
      repetir = false; // Sale del bucle y regresa al menú principal
```

```
            break;
        }

        system("pause"); // Pausa la ejecución para que el usuario vea los cambios antes de continuar
    } while (repetir);
}

// Métodos de acceso y modificación para las filas del tablero
int Config::getfilasTablero()
{
    return this->filasTablero;
}

int Config::setfilasTablero(int filasTablero)
{
    this->filasTablero = filasTablero;
}

// Métodos de acceso y modificación para las columnas del tablero
int Config::getcolumnasTablero()
{
    return this->columnasTablero;
}

int Config::setcolumnasTablero(int columnasTablero)
{
    this->columnasTablero = columnasTablero;
}
```

```
// Métodos de acceso y modificación para la cantidad de minas en el tablero
int Config::getminasTablero()
{
   return this->minasTablero;
}

int Config::setminasTablero(int minasTablero)
{
   this->minasTablero = minasTablero;
}

// Métodos de acceso y modificación para el modo desarrollador (true/false)
bool Config::getmodoDesarrolladorTablero()
{
   return this->modoDesarrolladorTablero;
}

bool Config::setmodoDesarrolladorTablero(bool modoDesarrolladorTablero)
{
   this->modoDesarrolladorTablero = modoDesarrolladorTablero;
}

// Métodos de acceso y modificación para la cantidad de vidas del jugador
int Config::getvidasTablero()
{
   return this->vidasTablero;
}

int Config::setvidasTablero(int vidasTablero)
```

```
{
   this->vidasTablero = vidasTablero;
}
```

Juego.cpp

```
#include "Juego.h" // Incluye el encabezado de la clase Juego
#include <fstream> // Biblioteca para manejo de archivos (aunque no se usa en este fragmento)
#include <unistd.h> // Biblioteca para funciones del sistema (en Linux o macOS)

using namespace std;

// Genera un número aleatorio dentro de un rango (mínimo, máximo)
int Juego::aleatorio_en_rango(int minimo, int maximo)
{
   return minimo + rand() / (RAND_MAX / (maximo - minimo + 1) + 1);
}

// Devuelve una fila aleatoria dentro del rango del tablero
int Juego::filaAleatoria()
{
   return this->aleatorio_en_rango(0, this->tablero.getAlturaTablero() - 1);
}

// Devuelve una columna aleatoria dentro del rango del tablero
```

```
int Juego::columnaAleatoria()
{
   return this->aleatorio_en_rango(0, this->tablero.getAnchoTablero() - 1);
}

// Constructor de la clase Juego
Juego::Juego(Tablero tablero, int cantidadMinas)
{
   this->tablero = tablero; // Inicializa el tablero
   this->cantidadMinas = cantidadMinas; // Inicializa la cantidad de minas
   this->colocarMinasAleatoriamente(); // Coloca minas de manera aleatoria en el tablero
}

// Coloca las minas aleatoriamente en el tablero hasta alcanzar la cantidad especificada
void Juego::colocarMinasAleatoriamente()
{
   int x, y, minasColocadas = 0;

   while (minasColocadas < this->cantidadMinas) // Bucle hasta colocar todas las minas
   {
      x = this->columnaAleatoria(); // Genera una columna aleatoria
      y = this->filaAleatoria(); // Genera una fila aleatoria
      if (this->tablero.colocarMina(x, y)) // Intenta colocar una mina en la posición generada
      {
         minasColocadas++; // Incrementa el contador si la mina fue colocada exitosamente
      }
   }
}
```

```
// Solicita al usuario la fila donde desea jugar
int Juego::solicitarFilaUsuario()
{
   int fila = 0;
   cout << "Ingresa la FILA en la que desea jugar: ";
   cin >> fila;
   return fila - 1; // Ajusta el índice para que sea compatible con el tablero
}

// Solicita al usuario la columna donde desea jugar
int Juego::solicitarColumnaUsuario()
{
   int columna = 0;
   cout << "Ingresa la COLUMNA en la que desea jugar: ";
   cin >> columna;
   return columna - 1; // Ajusta el índice para que sea compatible con el tablero
}

// Verifica si el jugador ha ganado (todas las celdas sin minas están descubiertas)
bool Juego::jugadorGana()
{
   int conteo = this->tablero.contarCeldasSinMinasYSinDescubrir(); // Cuenta las celdas sin minas sin descubrir
   return conteo == 0; // Si no quedan celdas por descubrir, el jugador gana
}

// Método principal para iniciar el juego
void Juego::iniciar()
{
```

```
    int fila, columna;

    // Bucle principal del juego
    while (true)
    {
        this->tablero.imprimir(); // Imprime el estado actual del tablero
        fila = this->solicitarFilaUsuario(); // Solicita la fila al usuario
        columna = this->solicitarColumnaUsuario(); // Solicita la columna al usuario

        // Intenta descubrir la celda en la posición indicada
        bool respuestaAUsuario = this->tablero.descubrirMina(columna, fila);
        if (!respuestaAUsuario) // Si se descubre una mina, el jugador pierde
        {
            cout << "Perdiste el Juego\n";
            this->tablero.setModoDesarrollador(true); // Activa el modo desarrollador para mostrar
todas las minas
            this->tablero.imprimir(); // Muestra el tablero completo
            break;
        }

        if (this->jugadorGana()) // Si el jugador gana, muestra un mensaje y termina el juego
        {
            cout << "Ganaste el Juego\n";
            this->tablero.setModoDesarrollador(true); // Activa el modo desarrollador para mostrar
todas las minas
            this->tablero.imprimir(); // Muestra el tablero completo
            break;
        }
    }
}
```