public class Maze

{

private char[,] layout;

public int Width { get; }

public int Height { get; }

public Maze(string[] map)

{

Height = map.Length;

Width = map[0].Length;

layout = new char[Height, Width];

for (int i = 0; i < Height; i++)

{

for (int j = 0; j < Width; j++)

{

layout[i, j] = map[i][j];

}

}

}

public char GetCell(int x, int y) => layout[y, x];

public void SetCell(int x, int y, char value) => layout[y, x] = value;

public bool IsWall(int x, int y) => layout[y, x] == '#';

}

public class Player

{

public int X { get; private set; }

public int Y { get; private set; }

public int Lives { get; set; }

// Конструктор

public Player(int startX, int startY)

{

X = startX;

Y = startY;

Lives = 3; // начальное количество жизней

}

public void Move(int newX, int newY, Maze maze)

{

if (!maze.IsWall(newX, newY))

{

X = newX;

Y = newY;

}

}

}

public class Coin

{

public int X { get; }

public int Y { get; }

public Coin(int x, int y)

{

X = x;

Y = y;

}

}

public abstract class Enemy

{

public int X { get; protected set; }

public int Y { get; protected set; }

public abstract void Move(Maze maze);

}

public class RandomEnemy : Enemy

{

public override void Move(Maze maze)

{

// Логика случайного движения

Random rnd = new Random();

int newX = X + rnd.Next(-1, 2);

int newY = Y + rnd.Next(-1, 2);

if (!maze.IsWall(newX, newY))

{

X = newX;

Y = newY;

}

}

}

public class Game

{

private Maze maze;

private Player player;

private List<Coin> coins;

private List<Enemy> enemies;

public Game()

{

string[] map = {

"#########",

"# #",

"# # ### #",

"# # #",

"#########"

};

maze = new Maze(map);

player = new Player(1, 1);

coins = new List<Coin> { new Coin(3, 2), new Coin(1, 3) };

enemies = new List<Enemy> { new RandomEnemy() { X = 5, Y = 1 } };

// Основной игровой цикл здесь

}

public void Run()

{

while (true)

{

// Обработка ввода игрока

// Движение врагов

// Проверка на столкновение

// Обновление состояния игры

// Отрисовка

}

}

}