using System;

using System.Collections.Generic;

using System.Text;

using System.Threading;

namespace TheSnakeProject

{

public enum Movements

{

Moveright,

Moveleft,

Movetop,

Movebottom

}

class Player

{

public Movements someMovement { get; set; }

List<string[]> field = new List<string[]>();

public bool snakeflag = true;

static GameObj objplayer = new GameObj(GameObj.Entity.Player);

string player = objplayer.ToString();

string emptycell;

public Player(Field \_field)

{

\_field.FieldGeneration();

this.field = \_field.field;

this.emptycell = \_field.emptycell;

PlayerPlacement();

PlayerMove();

}

public void PlayerPlacement()

{

Random rdm = new Random();

while(true)

{

int x = rdm.Next(1, field.Count);

int y = rdm.Next(1, field[x].Length-1);

if (field[x][y] == emptycell)

{

field[x][y] = player;

render();

break;

}

}

}

public void render()

{

Console.Clear();

for (int i = 0; i < field.Count; i++)

{

Console.WriteLine(String.Join("", field[i]));

}

}

public void CheckMovement(ConsoleKeyInfo consolekey)

{

switch (consolekey.Key)

{

case ConsoleKey.RightArrow:

someMovement = Movements.Moveright;

break;

case ConsoleKey.LeftArrow:

someMovement = Movements.Moveleft;

break;

case ConsoleKey.UpArrow:

someMovement = Movements.Movetop;

break;

case ConsoleKey.DownArrow:

someMovement = Movements.Movebottom;

break;

default:

break;

}

}

public void PlayerMove()

{

while(snakeflag)

{

for (int x = 0; x < field.Count;x++)

{

for (int y = 0; y< field[x].Length;y++)

{

if (field[x][y] == player)

{

while(true)

{

var key = Console.ReadKey();

CheckMovement(key);

}

while (someMovement == Movements.Moveright)

{

if ((y + 1) != field[x].Length && field[x][y + 1] == emptycell)

{

field[x][y] = emptycell;

field[x][y + 1] = player;

if (field[x][y + 1] == player)

{

render();

Thread.Sleep(500);

continue;

}

}

}

while (someMovement == Movements.Moveleft) {

if ((y - 1) != 0 && field[x][y - 1] == emptycell)

{

field[x][y] = emptycell;

field[x][y - 1] = player;

if (field[x][y - 1] == player)

{

render();

Thread.Sleep(500);

continue;

}

}

}

while (someMovement == Movements.Movetop) {

if ((x - 1) != 0 && field[x - 1][y] == emptycell)

{

field[x][y] = emptycell;

field[x - 1][y] = player;

if (field[x - 1][y] == player)

{

render();

Thread.Sleep(500);

continue;

}

}

}

while (someMovement == Movements.Movebottom) {

if ((x + 1) != field.Count && field[x + 1][y] == emptycell)

{

field[x][y] = emptycell;

field[x + 1][y] = player;

if (field[x + 1][y] == player)

{

render();

Thread.Sleep(500);

continue;

}

}

}

}

}

}

}

}

}

}

public void Rotation(ConsoleKey key) {

if (rotate) {

switch (someMovement) {

case Movements.Moveleft:

case Movements.Moveright:

if (key == ConsoleKey.DownArrow)

someMovement= Movements.Movebottom;

else if (key == ConsoleKey.UpArrow)

someMovement = Movements.Movetop;

break;

case Direction.UP:

case Direction.DOWN:

if (key == ConsoleKey.LeftArrow)

someMovement = Movements.Moveleft;

else if (key == ConsoleKey.RightArrow)

someMovement = Movements.Moveright;

break;

}

rotate = false;

}

}