ОГБПОУ "Томский техникум информационных технологий"

**Змейка**

Подготовил студент  
группы 682  
Лебедюк Егор

Проверил  
Павлов В.С

Томск 2020

**Задание:**

Игры разработка игры «змейку»

**Код программы:**

class Program

|  |
| --- |
|  |
|  | using System.Collections.Generic; |
|  | using System.Linq; |
|  | using System.Text; |
|  | using System.Threading.Tasks; |
|  |  |
|  | namespace Game |
|  | { |
|  | class Program |
|  | { |
|  | static void Main(string[] args) |
|  | { |
|  | string a; |
|  | int b=0; |
|  |  |
|  | Osnova XYI = new Osnova(); |
|  | Console.CursorVisible = false; |
|  | Console.Clear(); |
|  | do |
|  | { |
|  | Console.WriteLine("Введите уровень сложности (легко, нормально, сложно)"); |
|  | a = Console.ReadLine(); |
|  | if ("ЛЕГКО" == a.ToUpper() || "нормально" == a.ToLower() || "сложно" == a.ToLower()) |
|  | { |
|  | b = 1; |
|  | } |
|  | } while (b!=1); |
|  | Console.Clear(); |
|  | switch (a.ToLower()) |
|  | { |
|  | case "легко": |
|  | XYI.max\_x = 10; |
|  | XYI.max\_y = 10; |
|  | break; |
|  | case "нормально": |
|  | XYI.max\_x = 15; |
|  | XYI.max\_y = 15; |
|  | break; |
|  | case "сложно": |
|  | XYI.max\_x = 20; |
|  | XYI.max\_y = 20; |
|  | break; |
|  | default: |
|  | break; |
|  | } |
|  | do |
|  | { |
|  | XYI.write\_map\_next(); |
|  | XYI.New\_fruct(); |
|  | XYI.clear\_map(); |
|  | XYI.move\_pers(); |
|  | XYI.Game\_Over(); |
|  | } while (XYI.end); |
|  | } |
|  | } |
|  | } |

class Osnova

|  |
| --- |
| using System; |
|  | using System.Collections.Generic; |
|  | using System.Text; |
|  |  |
|  | namespace Game |
|  | { |
|  | public class Osnova |
|  | { |
|  | private List<int> snakeX = new List<int>(); |
|  | private List<int> snakeY = new List<int>(); |
|  | public bool end = true; |
|  | private bool eating = false; |
|  | private int EX; |
|  | private int EY; |
|  | private int endSnakeX; |
|  | private int endSnakeY; |
|  | public int max\_x = 10; |
|  | public int max\_y = 10; |
|  | private double Time = 300; |
|  | ConsoleKeyInfo mKey = new ConsoleKeyInfo('Q', ConsoleKey.Q, false, false, false); |
|  | private string[,] mass = new string[20, 20]; |
|  | char hero = '@'; |
|  | int pers\_x, pers\_y; |
|  | int fruct\_x, fruct\_y; |
|  | Random rand = new Random(); |
|  | public Osnova() |
|  | { |
|  | this.completion\_map(); |
|  | this.start\_poz(); |
|  | this.write\_map(); |
|  | } |
|  | private void start\_poz() |
|  | { |
|  | do |
|  | { |
|  | pers\_x = rand.Next(0, max\_x); |
|  | pers\_y = rand.Next(0, max\_y); |
|  | fruct\_x = rand.Next(0, max\_x); |
|  | fruct\_y = rand.Next(0, max\_y); |
|  | } while (pers\_x == fruct\_x || pers\_x == fruct\_y || pers\_y == fruct\_y || pers\_y == fruct\_x); |
|  | snakeX.Add(pers\_x); |
|  | snakeY.Add(pers\_y); |
|  |  |
|  | mass[snakeX[0], snakeY[0]] = hero.ToString(); |
|  | mass[fruct\_x, fruct\_y] = "\*"; |
|  | } |
|  | private void reversX() |
|  | { |
|  | endSnakeX = snakeX[snakeX.Count - 1]; |
|  | endSnakeY = snakeY[snakeY.Count - 1]; |
|  | for (int i = snakeX.Count - 1; i > 0; --i) |
|  | { |
|  | snakeX[i] = snakeX[i - 1]; |
|  | snakeY[i] = snakeY[i - 1]; |
|  | } |
|  | } |
|  | public void completion\_map() |
|  | { |
|  | for (int i = 0; i < max\_x; i++) |
|  | { |
|  | for (int j = 0; j < max\_y; j++) |
|  | { |
|  | mass[i, j] = " "; |
|  | } |
|  | } |
|  | } |
|  | public void write\_map() |
|  | { |
|  | for (int i = 0; i < max\_x; i++) |
|  | { |
|  | for (int j = 0; j < max\_y; j++) |
|  | { |
|  | Console.BackgroundColor = ConsoleColor.Red; |
|  | Console.Write(mass[i, j]); |
|  | } |
|  | Console.WriteLine(); |
|  | } |
|  | Console.BackgroundColor = ConsoleColor.Black; |
|  | } |
|  | public void write\_map\_next() |
|  | { |
|  | Console.SetCursorPosition(fruct\_y, fruct\_x); |
|  | Console.BackgroundColor = ConsoleColor.Red; |
|  | Console.Write("\*"); |
|  | for (int i = snakeY.Count - 1; i >= 0; --i) |
|  | { |
|  | Console.SetCursorPosition(snakeY[i], snakeX[i]); |
|  | Console.BackgroundColor = ConsoleColor.Red; |
|  | Console.Write("@"); |
|  | } |
|  |  |
|  | } |
|  | public void clear\_map() |
|  | { |
|  | for (int i = 0; i < max\_x; i++) |
|  | { |
|  | for (int j = 0; j < max\_y; j++) |
|  | { |
|  | Console.SetCursorPosition(i, j); |
|  | Console.BackgroundColor = ConsoleColor.Red; |
|  | Console.Write(" "); |
|  | } |
|  | } |
|  | } |
|  | public void move\_pers() |
|  | { |
|  | if (mKey.Key == ConsoleKey.Q || Console.KeyAvailable == true) |
|  | { |
|  | mKey = Console.ReadKey(); |
|  | } |
|  | switch (mKey.Key) |
|  | { |
|  | case ConsoleKey.W: |
|  | if (snakeX[0] > 0) |
|  | { reversX(); snakeX[0]--; } |
|  | break; |
|  | case ConsoleKey.A: |
|  | if (snakeY[0] > 0) |
|  | { reversX(); snakeY[0]--; } |
|  | break; |
|  | case ConsoleKey.S: |
|  | if (snakeX[0] < max\_x - 1) |
|  | { reversX(); snakeX[0]++; } |
|  | break; |
|  | case ConsoleKey.D: |
|  | if (snakeY[0] < max\_y - 1) |
|  | { reversX(); snakeY[0]++; } |
|  | break; |
|  | default: |
|  | break; |
|  | } |
|  | // Console.Clear(); |
|  | eatingSeeng(); |
|  | paintSnake(); |
|  | //this.write\_map(); |
|  | } |
|  | private void eatingSeeng() |
|  | { |
|  | if (eating) |
|  | { |
|  | Time = Time - 1; |
|  | if (EY == endSnakeY && EX == endSnakeX) |
|  | { |
|  | snakeY.Add(endSnakeY); |
|  | snakeX.Add(endSnakeX); |
|  | eating = false; |
|  | } |
|  | } |
|  | } |
|  | private void paintSnake() |
|  | { |
|  | if (snakeY.Count == snakeX.Count) |
|  | { |
|  | completion\_map(); |
|  | mass[fruct\_x, fruct\_y] = "\*"; |
|  | for (int i = 0; i < snakeX.Count; ++i) |
|  | { |
|  | mass[snakeX[i], snakeY[i]] = hero.ToString(); |
|  | } |
|  | } |
|  | } |
|  | public void New\_fruct() |
|  | { |
|  | System.Threading.Thread.Sleep((int)Time); |
|  | if (snakeX[0] == fruct\_x && snakeY[0] == fruct\_y) |
|  | { |
|  | eating = true; |
|  | EX = fruct\_x; |
|  | EY = fruct\_y; |
|  | do |
|  | { |
|  | fruct\_x = rand.Next(0, max\_x); |
|  | fruct\_y = rand.Next(0, max\_y); |
|  | } while (snakeX[0] == fruct\_x || snakeX[0] == fruct\_y || snakeY[0] == fruct\_y || snakeY[0] == fruct\_x); |
|  | } |
|  | } |
|  | public void Game\_Over() |
|  | { |
|  | for (int i = snakeX.Count - 1; i > 0; --i) |
|  | { |
|  | if (i > 0) |
|  | { |
|  | if (snakeX[0] == snakeX[i] && snakeY[0] == snakeY[i]) |
|  | { |
|  | Console.WriteLine("Game Over"); |
|  | end = false; |
|  | break; |
|  | } |
|  |  |
|  | } |
|  | } |
|  | } |
|  | } |
|  | } |

Скриншот работы программы:













