Бондаренко Борис 682гр.

***Задание:***

Сделать змейку.

**Функциональные требования:**

- управление змейкой должно происходить по клавишам (вверх, вниз, влево, вправо).

- Змейка должна непрерывно двигаться в сторону, куда нажал пользователь, по нажатии на другую клавишу, змейка должна поменять свое движение в сторону нажатой клавиши

- скорость и размер поля должен меняться в зависимости от выбранной пользователем сложности игры

Легко [поле10х10, скорость медленная]

Нормально [поле15х15, скорость нормальная]

Сложно [после20х20, скорость быстрая]

**Требования к оформлению:**

- поле должно быть очерченным и понятным для пользователя

- змейка должна обладать головой и телом (символы ASCII)

- «фрукты» должны так же иметь свои собственные символы (ASCII)

**Требования к выполнению работы(общ)**

- работа должна быть выполнена на функциях и классах

- поле и змейка не должны «мерцать» при движении

***Код:***

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading;

using System.Threading.Tasks;

using System.Timers;

namespace Snake

{

class Program

{

static void Main(string[] args)

{

// поле

int n = 0, m = 0;

Console.SetCursorPosition(20, 5);

Console.WriteLine("Управление стрелочками");

// Уровень сложности

Console.SetCursorPosition(50, 5);

Console.WriteLine("Выберите сложность:");

Console.SetCursorPosition(50, 6);

Console.ForegroundColor = ConsoleColor.Cyan;

Console.WriteLine("1) Легкая - [Размер поля 10х10, низкая скорость]");

Console.SetCursorPosition(50, 7);

Console.ForegroundColor = ConsoleColor.DarkMagenta;

Console.WriteLine("2) Нормальная - [Размер поля 15х15, нормальная скорость]");

Console.SetCursorPosition(50, 8);

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("3) Сложная - [Размер поля 20х20, высокая скорость]");

Console.SetCursorPosition(65, 10);

Console.ForegroundColor = ConsoleColor.Blue;

Console.Write("Выбор: ");

int IdOfComplexity = Convert.ToInt32(Console.ReadLine());

switch (IdOfComplexity)

{

case 1:

Console.SetBufferSize(200, 50);

n = 10;

m= 10;

Thread.Sleep(300);

Console.Clear();

break;

case 2:

Console.SetBufferSize(200, 50);

n = 15;

m = 15;

Thread.Sleep(100);

Console.Clear();

break;

case 3:

Console.SetBufferSize(200, 50);

n = 20;

m = 20;

Thread.Sleep(30);

Console.Clear();

break;

default:

Console.WriteLine("Id does not exist");

Console.WriteLine("Game Over");

break;

}

// фрукт

int xz = 0, yz = 0, xv = 0, yv = 0, size = 1;

ConsoleKeyInfo f;

string move = "";

bool eat = false;

String[,] map = new string[n, m];

int[] masx = new int[size];

int[] masy = new int[size];

bool game = true;

Game.Otrisovka(n, m, ref xz, ref yz, ref yv, ref xv, ref map);

while (game == true) // зацикливаем

{

Console.WriteLine();

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

{

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

{

Console.Write(map[i, j]);

}

Console.WriteLine();

}

// движение

if (Console.KeyAvailable == true)

{

f = Console.ReadKey();

if (f.Key == ConsoleKey.UpArrow)

if (move != "Down")

move = "Up";

if (f.Key == ConsoleKey.DownArrow)

if (move != "Up")

move = "Down";

if (f.Key == ConsoleKey.RightArrow)

if (move != "Left")

move = "Right";

if (f.Key == ConsoleKey.LeftArrow)

if (move != "Right")

move = "Left";

}

Game.Move(n, m, ref xz, ref yz, move, ref map);

Game.EatGenerate(n, m, ref map, ref xv, ref yv, xz, yz, ref eat, masx, masy);

Game.Tail(ref masx, ref masy, ref map, ref eat, ref size, xz, yz);

Game.Lose(n, m, masx, masy, xz, yz, ref game, size);

Game.Win(n, m, map, ref game);

Thread.Sleep(400);

Console.Clear();

}

Console.ReadKey();

}

}

}

***Класс:***

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Snake

{

class Game

{

public static void Otrisovka(int n, int m, ref int xz, ref int yz, ref int yv, ref int xv, ref string[,] map)

{

Random rand = new Random(); // функцию случайной генерации с обьектом rand

for (int i = 0; i < n; i++) // от рисовать границы поля

{

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

{

map[i, j] = " ";

if (i == 0 || j == 0 || i == n - 1 || j == m - 1)

map[i, j] = "#";

}

}

yz = n - 2; xz = 1;

map[yz, xz] = "X"; // поставить на поле «голову» змеи

while (true) // и создать в нем «пищу»

{

xv = rand.Next(1, m - 2);

yv = rand.Next(1, n - 2);

if (xv == xz && yv != yz || xv != xz && yv == yz || xv != xz && yv != yz)

{

break;

}

}

map[yv, xv] = "\*";

}

public static void Move(int n, int m, ref int xz, ref int yz, string move, ref string[,] map)

{

map[yz, xz] = " ";

switch (move)

{

case "Up":

if (yz - 1 >= 0) yz -= 1;

break;

case "Down":

if (yz + 1 < n) yz += 1;

break;

case "Left":

if (xz - 1 >= 0) xz -= 1;

break;

case "Right":

if (xz + 1 < m) xz += 1;

break;

}

map[yz, xz] = "X";

}

public static void EatGenerate(int n, int m, ref string[,] map, ref int xv, ref int yv, int xz, int yz, ref bool eat, int[] masx, int[] masy)

{

if (xv == xz && yv == yz)

{

eat = true;

Random rand = new Random();

bool s = true;

while (s == true)

{

xv = rand.Next(1, m - 2);

yv = rand.Next(1, n - 2);

int p = 0;

if (xv != xz && yv == yz || yv != yz && xv == xz || xv != xz && yv != yz)

{

for (int i = 0; i < masx.Length; i++)

{

if (xv != masx[i] && yv == masy[i] || yv != masy[i] && xv == masx[i] || xv != masx[i] && yv != masy[i])

p++;

if (p == masx.Length)

s = false;

}

}

}

map[yv, xv] = "\*";

}

}

public static void Tail(ref int[] masx, ref int[] masy, ref string[,] map, ref bool eat, ref int size, int xz, int yz)

{

if (eat == true)

{

size++;

Array.Resize<int>(ref masx, size);

Array.Resize<int>(ref masy, size);

eat = false;

}

map[masy[size - 1], masx[size - 1]] = " ";

for (int i = masx.Length - 1; i > 0; i--)

{

masx[i] = masx[i - 1];

masy[i] = masy[i - 1];

}

masx[0] = xz;

masy[0] = yz;

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

{

map[masy[i], masx[i]] = "X";

}

}

public static void Lose(int n, int m, int[] masx, int[] masy, int xz, int yz, ref bool game, int size)

{

if (xz == 0 || xz == m - 1 || yz == 0 || yz == n - 1)

{

game = false;

Console.WriteLine("You LOSE!!!");

}

for (int i = 1; i < size; i++)

{

if (masx[i] == xz && masy[i] == yz)

{

game = false;

Console.WriteLine("You LOSE!!!");

}

}

}

public static void Win(int n, int m, string[,] map, ref bool game)

{

int p = 0;

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

{

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

{

if (i != 0 || j != 0 || i != n - 1 || j != m - 1)

{

if (map[i, j] == "X")

{

p++;

}

}

}

}

if (p == map.Length - n \* 2 - m \* 2 + 4)

{

game = false;

Console.WriteLine("You WIN!!!");

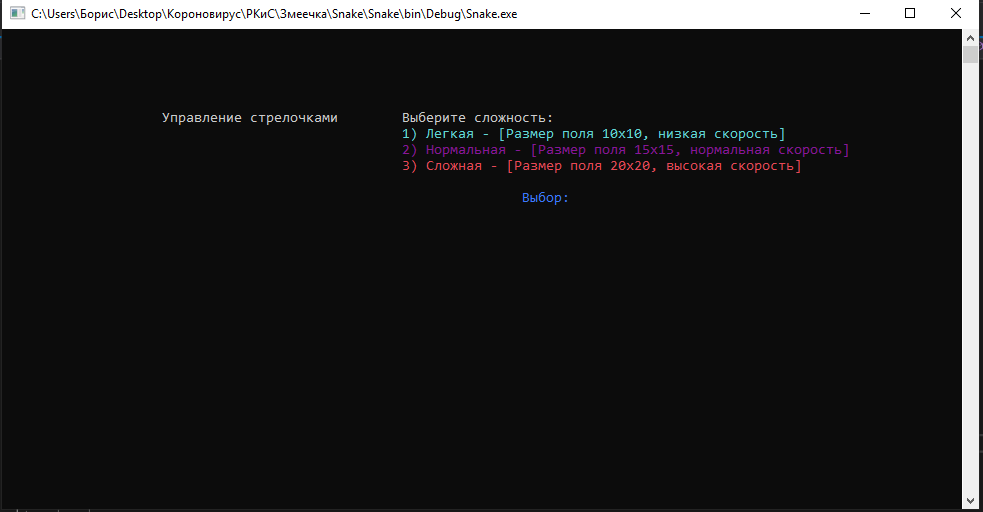
}

}

}

}

***Скриншоты:***

******

