«Томский техникум информационных технологий»

Отчет «Игра змейка»

Выполнил студент 682 группы

Кокин Кирилл Денисович

Томск,2020

Задание:

Нужно разработать игру «змейку»

Функциональные требования:

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

- Змейка должна неприрывно 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 очерченым 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

Описание работы программы:

После запуска программы, открывается меню игры, в котором кратко описано управление, необходимость играть на английской раскладке клавиатуры, а также выбор уровня сложности(легко[поле10х10, скорость медленная], нормально[поле15х15,скорость нормальная], сложно [после20х20,скорость быстрая]. Цель игры увеличить длину змейки до 20. Игра заканчивается при столкновении с оградой.

Код программы

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Threading;

namespace змейка

{

class Snake

{

int[] X = new int[20];//отслеживание координат по х каждой части змейки

int[] Y = new int[20];//отслеживание координат по y каждой части змейки

int fruitX; //отслеживание координаты по х фрукта

int fruitY; //отслеживание координаты по y фрукта

int parts = 4;//начальная длина змейки -1

ConsoleKeyInfo KeyInfo = new ConsoleKeyInfo();

char key = 's';

Random random = new Random();

//первоначальное отрисовывание змейки и фрукта

Snake(int razmer)

{

X[0] = 5;

Y[0] = 5;

Console.CursorVisible = false;

fruitX = random.Next(1, (razmer - 1));

fruitY = random.Next(1, (razmer- 1));

}

//Отрисовка поля

public void WritePole(int razmer)

{

int i, j, k;

for (i = 0, j = 0, k = 0; i < razmer; k++, i++)

{

Thread.Sleep(75);

Console.SetCursorPosition(i, j);

Console.Write("#");

Console.SetCursorPosition(j, k);

Console.Write("#");

}

for (i = 0, j = razmer, k = 0; i < razmer; i++, k++)

{

Thread.Sleep(75);

Console.SetCursorPosition(i, j);

Console.Write("#");

Console.SetCursorPosition(j, k);

Console.Write("#");

}

}

public void input()

{

if (Console.KeyAvailable)

{

KeyInfo = Console.ReadKey(true);

key = KeyInfo.KeyChar;

}

}

//отрисовка зчейки

public void WritePointSnake(int x, int y)

{

Console.SetCursorPosition(x, y);

Console.Write("O");

}

//отрисовка фрукта

public void WritePointFruit(int x, int y)

{

Console.SetCursorPosition(x, y);

Console.Write("@");

}

public bool Logik(int speed, int razmer)

{

bool sostoyanie = true;

//проверка на сьедание фрукта

if (X[0] == fruitX)

{

if (Y[0] == fruitY)

{

parts++;

fruitX = random.Next(1, razmer - 1);

fruitY = random.Next(1, razmer - 1);

}

}

if (X[0] == 0 || Y[0] == 0 || X[0] == razmer || Y[0] == razmer)

{

sostoyanie = false;

Console.Clear();

Console.WriteLine("игра оконцена");

Console.ReadKey();

return sostoyanie;

}

// проверка на победу

if (parts == 20)

{

sostoyanie = false;

Console.Clear();

Console.WriteLine("You Win");

Console.ReadKey();

return sostoyanie;

}

//передвижение координат туловища

for (int i = parts; i > 1; i--)

{

X[i - 1] = X[i - 2];

Y[i - 1] = Y[i - 2];

}

//изменение направления

switch (key)

{

case 'w': Y[0]--; break;

case 's': Y[0]++; break;

case 'a': X[0]--; break;

case 'd': X[0]++; break;

}

//отрисовывает змейку и фрукт

for (int i = 0; i <= (parts - 1); i++)

{

WritePointSnake(X[i], Y[i]);

WritePointFruit(fruitX, fruitY);

if (parts - 1 == i)

{

Console.SetCursorPosition(X[parts - 1], Y[parts - 1]);

Console.Write(" ");

}

}

Thread.Sleep(speed);//интервал действия

return sostoyanie;

}

public static void Main()

{

bool sostoyanie = true;//переменная отвечающая за состояние игры

int speed = 0, razmer = 0, levl;

levl = Menu();//вызов функции меню и присвоение уровня сложности

switch (levl)//выбор характеристик сложности

{

case 1: speed = 600; razmer = 12; break;

case 2: speed = 400; razmer = 17; break;

case 3: speed = 250; razmer = 22; break;

default: Console.Clear(); Console.WriteLine("введено недопустимое значение"); break;

}

Snake snake = new Snake(razmer);

snake.WritePole(razmer);

while (sostoyanie)

{

snake.input();

sostoyanie = snake.Logik(speed, razmer);//выполнение основных логических проверок

}

Console.Clear();

Console.WriteLine("спасибо за игру");

Console.ReadKey();

}

public static int Menu()

{

int i, k, j;

int levl;

for (i = 0, j = 0, k = 0; i < 24; k++, i++)

{

Thread.Sleep(70);

Console.SetCursorPosition(i, j);

Console.Write("#");

Console.SetCursorPosition(j, k);

Console.Write("#");

}

for (i = 0, j = 24, k = 0; i < 24; i++, k++)

{

Thread.Sleep(70);

Console.SetCursorPosition(i, j);

Console.Write("#");

Console.SetCursorPosition(j, k);

Console.Write("#");

}

Console.SetCursorPosition(1, 2);

Console.WriteLine("Клавиши управления:\t");

Console.WriteLine("#w - движение вверх\t");

Console.WriteLine("#s - движение вниз\t");

Console.WriteLine("#a - движение влево\t");

Console.WriteLine("#d - движение вправо\t");

Console.WriteLine("#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\t");

Console.WriteLine("#играть с ENG раскладкой\t");

Console.WriteLine("#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\t");

Console.WriteLine("#выберите сложность:\t");

Console.WriteLine("#введите цифру\t");

Console.WriteLine("#1.легко\t");

Console.WriteLine("#2.нормально\t");

Console.WriteLine("#3.сложно");

Console.SetCursorPosition(1, 15);

levl = Convert.ToInt32(Console.ReadLine());

Console.Clear();

return levl;

}

}

}

Скриншоты

