Департамент профессионального образования Томской области

ОГБПОУ «ТОМСКИЙ ТЕХНИКУМ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ»

Отчет по выполнению задания “Змейка”

Разработка кода информационных систем

Студент группы\_682 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Бледных Д.С.

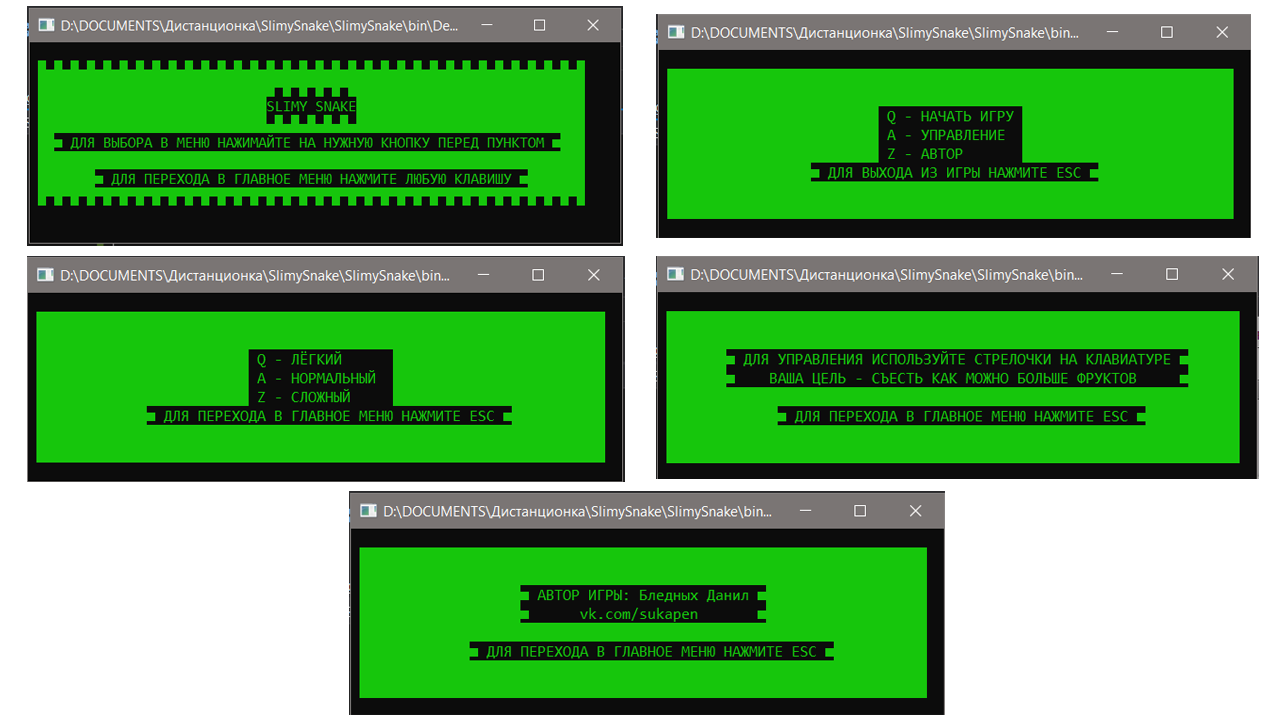
Преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Павлов В.С.

Томск, 2020 г.

**ЛАБОРАТОРНАЯ РАБОТА №6**

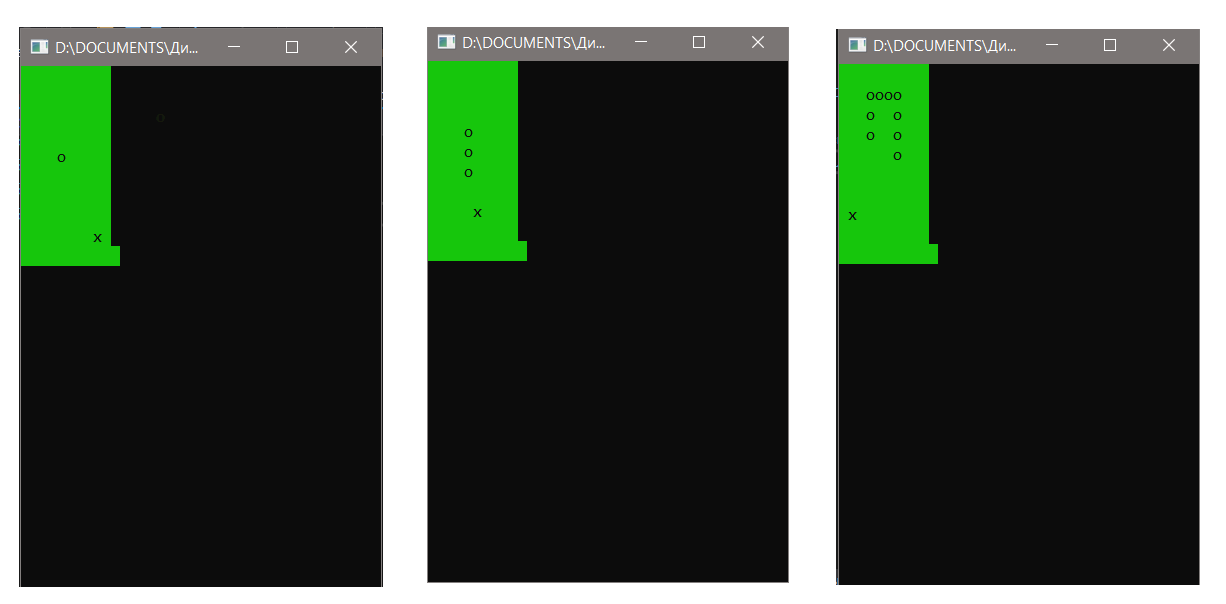
**Задание:**

Написать программную игру “Змейка” по определённым функциональным требованиям.

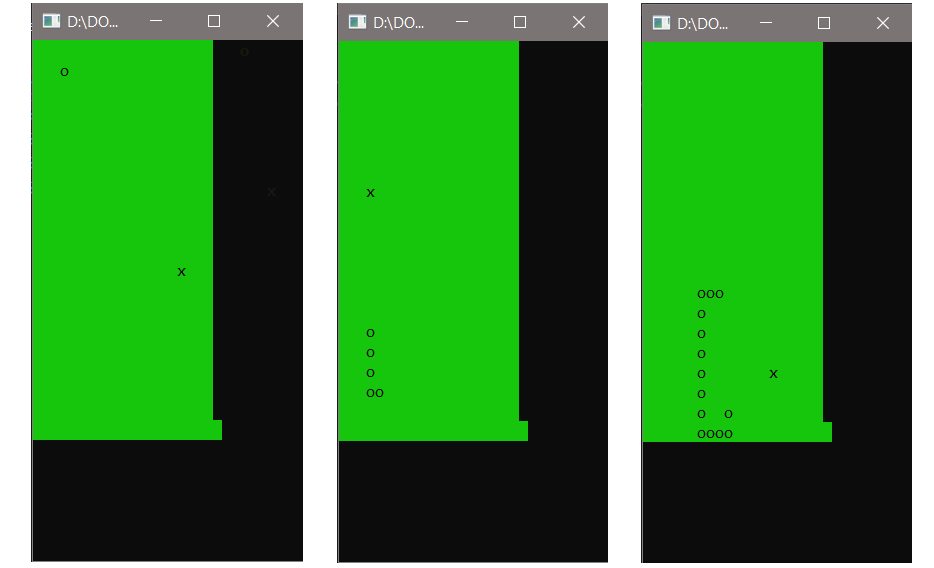
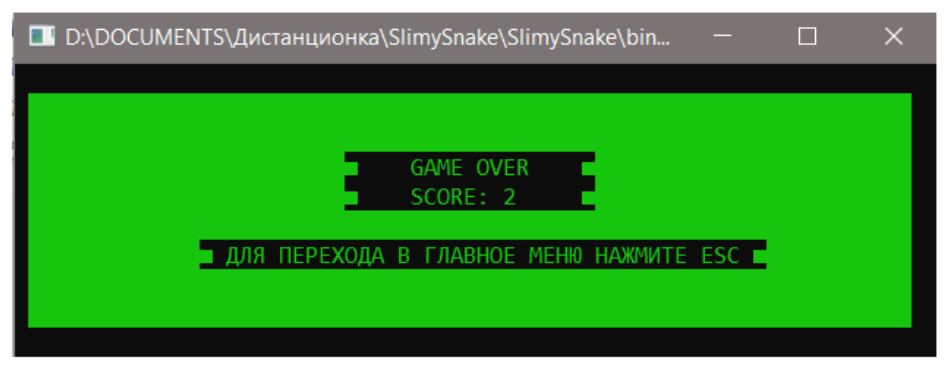


**Скриншоты:**

Скриншот №1 - Меню

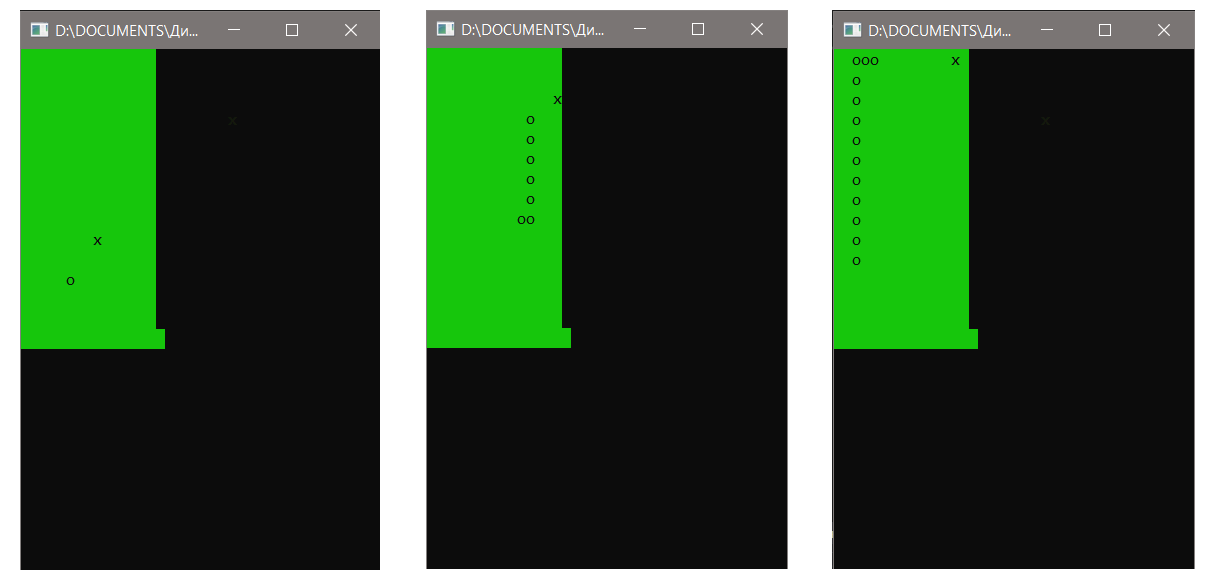


Скриншот №2 – Лёгкий уровень



Скриншот №5 – Game Over

Скриншот №4 – Сложный уровень



Скриншот №3 – Нормальный уровень

**Код программы:**

Класс Program:

using System;

namespace SlimySnake

{

public class Program

{

// задаём две переменные для изменения окна консоли:

static readonly int x = 69;

static readonly int y = 9;

static void Main(string[] args)

{

// отключем курсор:

Console.CursorVisible = false;

// задаём размер и буфер консоли:

Console.SetWindowSize(x + 1, y + 1);

Console.SetBufferSize(x + 1, y + 1);

// вызываем класс Menu и функцию в нём M.Present():

Menu M = new Menu();

M.Present();

}

}

}

Класс Menu:  
using System;

namespace SlimySnake

{

public class Menu

{

// задаём две переменные для изменения окна консоли:

static readonly int x = 69;

static readonly int y = 9;

public void Present()

{

// очищаем консоль:

Console.Clear();

// задаём цвета консоли и букв:

Console.BackgroundColor = ConsoleColor.Black;

Console.ForegroundColor = ConsoleColor.Green;

// текст в консоли:

Console.WriteLine(" ");

Console.WriteLine(" █▄█▄█▄█▄█▄█▄█▄█▄█▄█▄█▄█▄█▄█▄█▄█▄█▄█▄█▄█▄█▄█▄█▄█▄█▄█▄█▄█▄█▄█▄█▄█▄█▄█");

Console.WriteLine(" █████████████████████████████▀█▀█▀█▀█▀█████████████████████████████");

Console.WriteLine(" ████████████████████████████SLIMY SNAKE████████████████████████████");

Console.WriteLine(" ████████████████████████████▄█▄█▄█▄█▄█▄████████████████████████████");

Console.WriteLine(" ██■ ДЛЯ ВЫБОРА В МЕНЮ НАЖИМАЙТЕ НА НУЖНУЮ КНОПКУ ПЕРЕД ПУНКТОМ ■███");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

Console.WriteLine(" ███████■ ДЛЯ ПЕРЕХОДА В ГЛАВНОЕ МЕНЮ НАЖМИТЕ ЛЮБУЮ КЛАВИШУ ■███████");

Console.WriteLine(" █▀█▀█▀█▀█▀█▀█▀█▀█▀█▀█▀█▀█▀█▀█▀█▀█▀█▀█▀█▀█▀█▀█▀█▀█▀█▀█▀█▀█▀█▀█▀█▀█▀█");

// задаём параметр и назначаем на буковки перемещение по меню:

ConsoleKeyInfo Choise = new ConsoleKeyInfo();

Choise = Console.ReadKey();

// в данном случае пользователь может нажать на пробел, выполнится if, если он нажмёт на любую другую кнопку, выполнится else, они одинаковые:

if (Choise.Key == ConsoleKey.Spacebar)

{

Console.Clear();

MainMenu();

}

else

{

Console.Clear();

MainMenu();

}

}

public void MainMenu()

{

// задаём размер и буфер консоли (на всякий случай сделал, чтобы меньше глакков было, гы)

Console.SetWindowSize(x + 1, y + 1);

Console.SetBufferSize(x + 1, y + 1);

Console.WriteLine("");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

Console.WriteLine(" █████████████████████████ Q — НАЧАТЬ ИГРУ █████████████████████████");

Console.WriteLine(" █████████████████████████ A — УПРАВЛЕНИЕ █████████████████████████");

Console.WriteLine(" █████████████████████████ Z — АВТОР █████████████████████████");

Console.WriteLine(" █████████████████■ ДЛЯ ВЫХОДА ИЗ ИГРЫ НАЖМИТЕ ESC ■████████████████");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

// задаём параметр и назначаем на буковки перемещение по меню:

ConsoleKeyInfo Choise = new ConsoleKeyInfo();

Choise = Console.ReadKey();

// задаём клавиши для перемещения по меню с вызовами функций и очищением консоли:

if (Choise.Key == ConsoleKey.Q)

{

Console.Clear();

GameMenu();

}

else if (Choise.Key == ConsoleKey.A)

{

Console.Clear();

Upra();

}

else if (Choise.Key == ConsoleKey.Z)

{

Console.Clear();

Author();

}

// если пользователь нажмёт Escape, то консоль закроется, по сути он выйдет из игры

else if (Choise.Key == ConsoleKey.Escape)

{

Environment.Exit(0);

}

// так же предусмотрено нажатие на другие клавиши, если случится, издастся звук, якобы "ошибки" и его перебросит на ту же функцию

else

{

Console.Beep();

Console.Clear();

MainMenu();

}

}

public void GameMenu()

{

Console.WriteLine("");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

Console.WriteLine(" █████████████████████████ Q — ЛЁГКИЙ █████████████████████████");

Console.WriteLine(" █████████████████████████ A — НОРМАЛЬНЫЙ █████████████████████████");

Console.WriteLine(" █████████████████████████ Z — СЛОЖНЫЙ █████████████████████████");

Console.WriteLine(" █████████████■ ДЛЯ ПЕРЕХОДА В ГЛАВНОЕ МЕНЮ НАЖМИТЕ ESC ■███████████");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

ConsoleKeyInfo Choise = new ConsoleKeyInfo();

Choise = Console.ReadKey();

if (Choise.Key == ConsoleKey.Q)

{

// чистим консоль, дабы меню пропало, а то достало:

Console.Clear();

Easy E = new Easy();

// задаём цикл дувайл, дабы всё корректно работало:

do

{

Console.ForegroundColor = ConsoleColor.Black;

E.UpdateMap();

E.NewFood();

E.ClearMap();

E.MoveHero();

E.GameOver();

} while (E.end);

}

else if (Choise.Key == ConsoleKey.A)

{

Console.Clear();

Normal N = new Normal();

do

{

Console.ForegroundColor = ConsoleColor.Black;

N.UpdateMap();

N.NewFood();

N.ClearMap();

N.MoveHero();

N.GameOver();

} while (N.end);

}

else if (Choise.Key == ConsoleKey.Z)

{

Console.Clear();

Hard H = new Hard();

do

{

Console.ForegroundColor = ConsoleColor.Black;

H.UpdateMap();

H.NewFood();

H.ClearMap();

H.MoveHero();

H.GameOver();

} while (H.end);

}

// возвращает в главное меню:

else if (Choise.Key == ConsoleKey.Escape)

{

Console.Clear();

MainMenu();

}

// так же дабы не было мисскиликов или специально не сломали моё прекрасное меню

else

{

Console.Beep();

Console.Clear();

GameMenu();

}

}

public void Author()

{

// ну тут ясно, автор типо крутой

Console.WriteLine("");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

Console.WriteLine(" ███████████████████■ АВТОР ИГРЫ: Бледных Данил ■███████████████████");

Console.WriteLine(" ███████████████████■ vk.com/sukapen ■███████████████████");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

Console.WriteLine(" █████████████■ ДЛЯ ПЕРЕХОДА В ГЛАВНОЕ МЕНЮ НАЖМИТЕ ESC ■███████████");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

ConsoleKeyInfo Choise = new ConsoleKeyInfo();

Choise = Console.ReadKey();

if (Choise.Key == ConsoleKey.Escape)

{

Console.Clear();

MainMenu();

}

else

{

Console.Beep();

Console.Clear();

Author();

}

}

public void Upra()

{

// ну это так, чтобы было

Console.WriteLine("");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

Console.WriteLine(" ███████■ ДЛЯ УПРАВЛЕНИЯ ИСПОЛЬЗУЙТЕ СТРЕЛОЧКИ НА КЛАВИАТУРЕ ■██████");

Console.WriteLine(" ███████■ ВАША ЦЕЛЬ - СЪЕСТЬ КАК МОЖНО БОЛЬШЕ ФРУКТОВ ■██████");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

Console.WriteLine(" █████████████■ ДЛЯ ПЕРЕХОДА В ГЛАВНОЕ МЕНЮ НАЖМИТЕ ESC ■███████████");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

ConsoleKeyInfo Choise = new ConsoleKeyInfo();

Choise = Console.ReadKey();

if (Choise.Key == ConsoleKey.Escape)

{

Console.Clear();

MainMenu();

}

else

{

Console.Beep();

Console.Clear();

Upra();

}

}

}

}

Класс Easy:

using System;

using System.Collections.Generic;

namespace SlimySnake

{

public class Easy

{

// параметр лист, для отображения змейки

private List<int> snakeX = new List<int>();

private List<int> snakeY = new List<int>();

// невозвратная переменная булл, для завершения игры

public bool end = true;

// невозвратная переменная айтинг, для считывания жратвы

private bool eating = false;

// ещё полезные переменные:

private int eX, eY;

// переменная для посчёта жратвы, задаём ей нуль

public int point = 0;

// задаём переменные для считывая конца змейки

private int endsnakeX, endsnakeY;

// переменные для размера поля:

private const int x = 10, y = 10;

// переменные для меню, чтобы геймовер корректно отображалось

static readonly int xg = 69;

static readonly int yg = 9;

private double Time = 300;

// параметр для считывания клавишь

ConsoleKeyInfo KeyInfo = new ConsoleKeyInfo();

// задаём поле с помощью массива

private string[,] mass = new string[x,y];

// задаём символ для змейки

char snake = 'o';

// вводим переменные для логических операций

int heroX, heroY, foodX, foodY;

// ну и рандом, вся жизнь на рандоме

Random rand = new Random();

public Easy()

{

// задаём размер окна

Console.SetWindowSize(x + 30, y + 15);

Console.SetBufferSize(x + 30, y + 15);

// выключаем курсор, потому что бесит

Console.CursorVisible = false;

// вызываем нужные нам функции:

CompletionMap();

Starting();

Mapping();

}

private void Starting()

{

// спавн змейки и еды, с помощью рандома и логических операций

do

{

heroX = rand.Next(0, x);

heroY = rand.Next(0, y);

foodX = rand.Next(0, x);

foodY = rand.Next(0, y);

} while (heroX == foodX || heroX == foodY || heroY == foodY || heroY == foodX);

// добавление змейки на поле:

snakeX.Add(heroX);

snakeY.Add(heroY);

mass[snakeX[0], snakeY[0]] = snake.ToString();

// задаем символ для еды

mass[foodX, foodY] = "x";

}

private void ReversX()

{

// поворот змейки

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 CompletionMap()

{

// корректное отображение карты:

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

{

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

{

// задаём символ, в нашем случае это пустота, чтобы он занимал всё остальное поле

mass[i, j] = " ";

}

}

}

public void Mapping()

{

// отрисовка карты

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

{

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

{

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

}

}

// задаём нашей мапе зелённый цвет, чтобы глаза вырывало

Console.BackgroundColor = ConsoleColor.Green;

}

public void UpdateMap()

{

// так сказать рендеринг мапы

Console.SetCursorPosition(foodY, foodX);

Console.Write("x");

for (int i = snakeY.Count - 1; i >= 0; --i)

{

Console.SetCursorPosition(snakeY[i], snakeX[i]);

Console.Write("o");

}

}

public void ClearMap()

{

// очистка карты, после рендеринга

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

{

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

{

Console.SetCursorPosition(i, j);

Console.Write(" ");

}

}

}

public void MoveHero()

{

// цикл иф, для того чтобы змейка двигалась сама

if (Console.KeyAvailable == true)

{

KeyInfo = Console.ReadKey();

}

// ну и цикл свитч, для управления

switch (KeyInfo.Key)

{

case ConsoleKey.UpArrow:

if (snakeX[0] > 0)

{ ReversX(); snakeX[0]--; }

break;

case ConsoleKey.LeftArrow:

if (snakeY[0] > 0)

{ ReversX(); snakeY[0]--; }

break;

case ConsoleKey.DownArrow:

if (snakeX[0] < x - 1)

{ ReversX(); snakeX[0]++; }

break;

case ConsoleKey.RightArrow:

if (snakeY[0] < y - 1)

{ ReversX(); snakeY[0]++; }

break;

default:

break;

}

Eating();

UpdateSnake();

}

private void Eating()

{

// функция котороя считывает, что поинт был съеден

if (eating)

{

Time = Time - 1;

if (eY == endsnakeY && eX == endsnakeX)

{

// подсчёт поинтов

point++;

// добавление роста змейки:

snakeY.Add(endsnakeY);

snakeX.Add(endsnakeX);

eating = false;

}

}

}

private void UpdateSnake()

{

// обновление змейки

if (snakeY.Count == snakeX.Count)

{

// рендеринг мапы

CompletionMap();

mass[foodX, foodY] = "x";

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

{

mass[snakeX[i], snakeY[i]] = snake.ToString();

}

}

}

public void NewFood()

{

// указываем скорость обновления консоли (я это так понял)

System.Threading.Thread.Sleep(400);

// производятся подсчёты, был ли съедем поинт

if (snakeX[0] == foodX && snakeY[0] == foodY)

{

eating = true;

eX = foodX;

eY = foodY;

// спавн нового поинта, с учётом рандома и отдаления от змейки

do

{

foodX = rand.Next(0, x);

foodY = rand.Next(0, y);

} while (snakeX[0] == foodX || snakeX[0] == foodY || snakeY[0] == foodY || snakeY[0] == foodX);

}

}

public void GameOver()

{

// цикл фор для логической операции, которая считывает, если змейка начнёт "заезжать" на саму себя

for (int i = snakeX.Count - 1; i > 0; --i)

{

if (i > 0)

{

if (snakeX[0] == snakeX[i] && snakeY[0] == snakeY[i])

{

// прекращение игры:

end = false;

// задаём цвет консоли и букв, который соответсвует меню

Console.BackgroundColor = ConsoleColor.Black;

Console.ForegroundColor = ConsoleColor.Green;

// чистим консоль

Console.Clear();

// задаём размер и буфер консоли, которые у меню

Console.SetWindowSize(xg + 1, yg + 1);

Console.SetBufferSize(xg + 1, yg + 1);

// геймовер:

Console.WriteLine("");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

Console.WriteLine(" ████████████████████████■ GAME OVER ■████████████████████████");

// так же здесь отображаются сколько поинтов съел игрок

Console.WriteLine($" ████████████████████████■ SCORE: {point} ■████████████████████████");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

Console.WriteLine(" █████████████■ ДЛЯ ПЕРЕХОДА В ГЛАВНОЕ МЕНЮ НАЖМИТЕ ESC ■███████████");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

ConsoleKeyInfo Choise = new ConsoleKeyInfo();

Choise = Console.ReadKey();

Menu M = new Menu();

// ну и возвращение в главное меню

if (Choise.Key == ConsoleKey.Escape)

{

Console.Clear();

M.MainMenu();

}

else

{

Console.Beep();

Console.Clear();

GameOver();

}

}

}

}

}

}

}

Класс Normal:

using System;

using System.Collections.Generic;

namespace SlimySnake

{

public class Normal

{

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, eY;

public int point = 0;

private int endsnakeX, endsnakeY;

private const int x = 15, y = 15;

static readonly int xg = 69;

static readonly int yg = 9;

private double Time = 300;

ConsoleKeyInfo KeyInfo = new ConsoleKeyInfo();

private string[,] mass = new string[x, y];

char snake = 'o';

int heroX, heroY, foodX, foodY;

Random rand = new Random();

public Normal()

{

Console.SetWindowSize(x + 25, y + 10);

Console.SetBufferSize(x + 25, y + 10);

Console.CursorVisible = false;

CompletionMap();

Starting();

Mapping();

}

private void Starting()

{

do

{

heroX = rand.Next(0, x);

heroY = rand.Next(0, y);

foodX = rand.Next(0, x);

foodY = rand.Next(0, y);

} while (heroX == foodX || heroX == foodY || heroY == foodY || heroY == foodX);

snakeX.Add(heroX);

snakeY.Add(heroY);

mass[snakeX[0], snakeY[0]] = snake.ToString();

mass[foodX, foodY] = "x";

}

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 CompletionMap()

{

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

{

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

{

mass[i, j] = " ";

}

}

}

public void Mapping()

{

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

{

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

{

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

}

}

Console.BackgroundColor = ConsoleColor.Green;

}

public void UpdateMap()

{

Console.SetCursorPosition(foodY, foodX);

Console.Write("x");

for (int i = snakeY.Count - 1; i >= 0; --i)

{

Console.SetCursorPosition(snakeY[i], snakeX[i]);

Console.Write("o");

}

}

public void ClearMap()

{

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

{

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

{

Console.SetCursorPosition(i, j);

Console.Write(" ");

}

}

}

public void MoveHero()

{

if (Console.KeyAvailable == true)

{

KeyInfo = Console.ReadKey();

}

switch (KeyInfo.Key)

{

case ConsoleKey.UpArrow:

if (snakeX[0] > 0)

{ ReversX(); snakeX[0]--; }

break;

case ConsoleKey.LeftArrow:

if (snakeY[0] > 0)

{ ReversX(); snakeY[0]--; }

break;

case ConsoleKey.DownArrow:

if (snakeX[0] < x - 1)

{ ReversX(); snakeX[0]++; }

break;

case ConsoleKey.RightArrow:

if (snakeY[0] < y - 1)

{ ReversX(); snakeY[0]++; }

break;

default:

break;

}

Eating();

UpdateSnake();

}

private void Eating()

{

if (eating)

{

Time = Time - 1;

if (eY == endsnakeY && eX == endsnakeX)

{

point++;

snakeY.Add(endsnakeY);

snakeX.Add(endsnakeX);

eating = false;

}

}

}

private void UpdateSnake()

{

if (snakeY.Count == snakeX.Count)

{

CompletionMap();

mass[foodX, foodY] = "x";

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

{

mass[snakeX[i], snakeY[i]] = snake.ToString();

}

}

}

public void NewFood()

{

System.Threading.Thread.Sleep(300);

if (snakeX[0] == foodX && snakeY[0] == foodY)

{

eating = true;

eX = foodX;

eY = foodY;

do

{

foodX = rand.Next(0, x);

foodY = rand.Next(0, y);

} while (snakeX[0] == foodX || snakeX[0] == foodY || snakeY[0] == foodY || snakeY[0] == foodX);

}

}

public void GameOver()

{

for (int i = snakeX.Count - 1; i > 0; --i)

{

if (i > 0)

{

if (snakeX[0] == snakeX[i] && snakeY[0] == snakeY[i])

{

end = false;

Console.BackgroundColor = ConsoleColor.Black;

Console.ForegroundColor = ConsoleColor.Green;

Console.Clear();

Console.SetWindowSize(xg + 1, yg + 1);

Console.SetBufferSize(xg + 1, yg + 1);

Console.WriteLine("");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

Console.WriteLine(" ████████████████████████■ GAME OVER ■████████████████████████");

Console.WriteLine($" ████████████████████████■ SCORE: {point} ■████████████████████████");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

Console.WriteLine(" █████████████■ ДЛЯ ПЕРЕХОДА В ГЛАВНОЕ МЕНЮ НАЖМИТЕ ESC ■███████████");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

ConsoleKeyInfo Choise = new ConsoleKeyInfo();

Choise = Console.ReadKey();

Menu M = new Menu();

if (Choise.Key == ConsoleKey.Escape)

{

Console.Clear();

M.MainMenu();

}

else

{

Console.Beep();

Console.Clear();

GameOver();

}

}

}

}

}

}

}

Класс Hard:

using System;

using System.Collections.Generic;

namespace SlimySnake

{

public class Hard

{

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, eY;

public int point = 0;

private int endsnakeX, endsnakeY;

private const int x = 20, y = 20;

static readonly int xg = 69;

static readonly int yg = 9;

private double Time = 300;

ConsoleKeyInfo KeyInfo = new ConsoleKeyInfo();

private string[,] mass = new string[x, y];

char snake = 'o';

int heroX, heroY, foodX, foodY;

Random rand = new Random();

public Hard()

{

Console.SetWindowSize(x + 10, y + 5);

Console.SetBufferSize(x + 10, y + 5);

Console.CursorVisible = false;

CompletionMap();

Starting();

Mapping();

}

private void Starting()

{

do

{

heroX = rand.Next(0, x);

heroY = rand.Next(0, y);

foodX = rand.Next(0, x);

foodY = rand.Next(0, y);

} while (heroX == foodX || heroX == foodY || heroY == foodY || heroY == foodX);

snakeX.Add(heroX);

snakeY.Add(heroY);

mass[snakeX[0], snakeY[0]] = snake.ToString();

mass[foodX, foodY] = "x";

}

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 CompletionMap()

{

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

{

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

{

mass[i, j] = " ";

}

}

}

public void Mapping()

{

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

{

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

{

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

}

}

Console.BackgroundColor = ConsoleColor.Green;

}

public void UpdateMap()

{

Console.SetCursorPosition(foodY, foodX);

Console.Write("x");

for (int i = snakeY.Count - 1; i >= 0; --i)

{

Console.SetCursorPosition(snakeY[i], snakeX[i]);

Console.Write("o");

}

}

public void ClearMap()

{

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

{

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

{

Console.SetCursorPosition(i, j);

Console.Write(" ");

}

}

}

public void MoveHero()

{

if (Console.KeyAvailable == true)

{

KeyInfo = Console.ReadKey();

}

switch (KeyInfo.Key)

{

case ConsoleKey.UpArrow:

if (snakeX[0] > 0)

{ ReversX(); snakeX[0]--; }

break;

case ConsoleKey.LeftArrow:

if (snakeY[0] > 0)

{ ReversX(); snakeY[0]--; }

break;

case ConsoleKey.DownArrow:

if (snakeX[0] < x - 1)

{ ReversX(); snakeX[0]++; }

break;

case ConsoleKey.RightArrow:

if (snakeY[0] < y - 1)

{ ReversX(); snakeY[0]++; }

break;

default:

break;

}

Eating();

UpdateSnake();

}

private void Eating()

{

if (eating)

{

Time = Time - 1;

if (eY == endsnakeY && eX == endsnakeX)

{

point++;

snakeY.Add(endsnakeY);

snakeX.Add(endsnakeX);

eating = false;

}

}

}

private void UpdateSnake()

{

if (snakeY.Count == snakeX.Count)

{

CompletionMap();

mass[foodX, foodY] = "x";

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

{

mass[snakeX[i], snakeY[i]] = snake.ToString();

}

}

}

public void NewFood()

{

System.Threading.Thread.Sleep(200);

if (snakeX[0] == foodX && snakeY[0] == foodY)

{

eating = true;

eX = foodX;

eY = foodY;

do

{

foodX = rand.Next(0, x);

foodY = rand.Next(0, y);

} while (snakeX[0] == foodX || snakeX[0] == foodY || snakeY[0] == foodY || snakeY[0] == foodX);

}

}

public void GameOver()

{

for (int i = snakeX.Count - 1; i > 0; --i)

{

if (i > 0)

{

if (snakeX[0] == snakeX[i] && snakeY[0] == snakeY[i])

{

end = false;

Console.BackgroundColor = ConsoleColor.Black;

Console.ForegroundColor = ConsoleColor.Green;

Console.Clear();

Console.SetWindowSize(xg + 1, yg + 1);

Console.SetBufferSize(xg + 1, yg + 1);

Console.WriteLine("");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

Console.WriteLine(" ████████████████████████■ GAME OVER ■████████████████████████");

Console.WriteLine($" ████████████████████████■ SCORE: {point} ■████████████████████████");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

Console.WriteLine(" █████████████■ ДЛЯ ПЕРЕХОДА В ГЛАВНОЕ МЕНЮ НАЖМИТЕ ESC ■███████████");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

Console.WriteLine(" ███████████████████████████████████████████████████████████████████");

ConsoleKeyInfo Choise = new ConsoleKeyInfo();

Choise = Console.ReadKey();

Menu M = new Menu();

if (Choise.Key == ConsoleKey.Escape)

{

Console.Clear();

M.MainMenu();

}

else

{

Console.Beep();

Console.Clear();

GameOver();

}

}

}

}

}

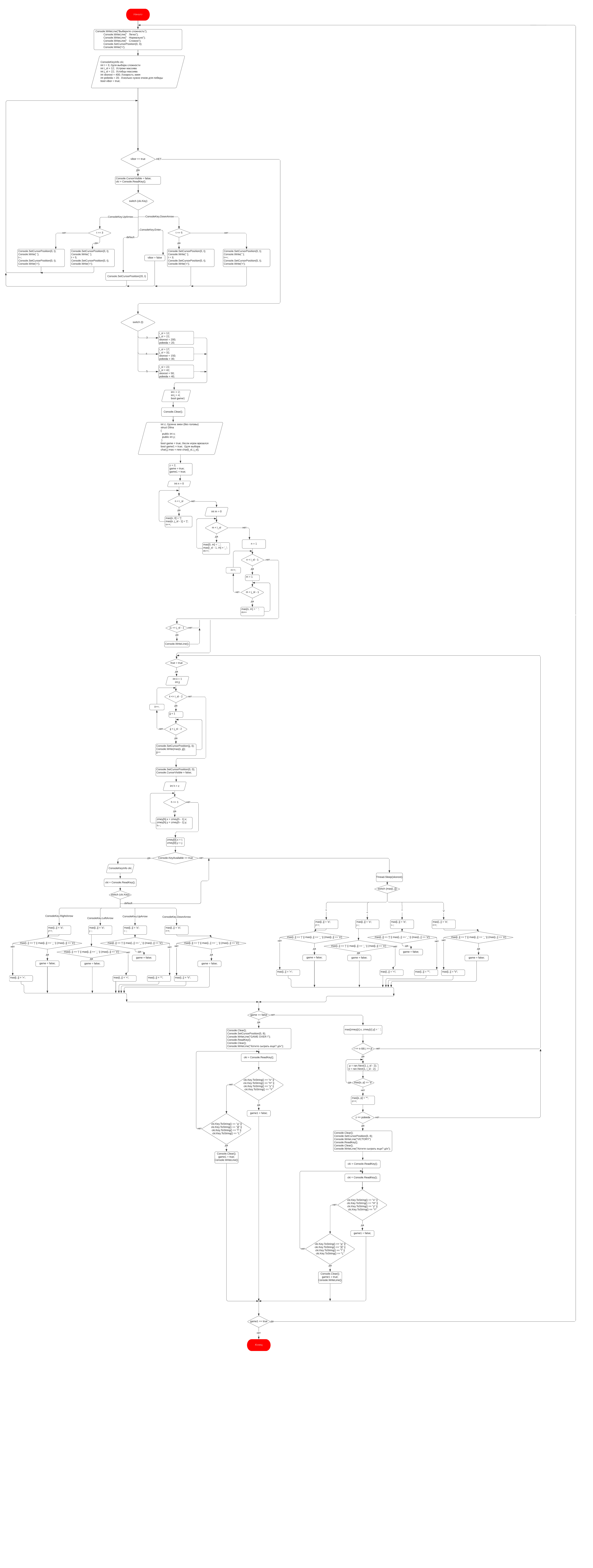
}

}

**Ссылка на github.com:**

<https://github.com/sukapen/snk>

**Блок схема:**

****