ООО «КРУ»

**Протокол**

**Совещание №1 – анализ кода программы**

26.02.2025 г. Котлас

Председатель Скворцов С.Е.

Присутствовали:

Автор – Симонов И.И.

Рецензенты – Аксеновский Р.С., Алиев К.И.

Секретарь – Якубук С.Д.

**Повестка дня**

1. Ревьюирование программного кода в игре «Змейка»

* Анализ исходного кода игры
* Консультация с автором по программному коду
* Исправления и поправки исходного кода

2. Написание рецензии на программу

* Анализ проведенной работы
* Написание рецензии

Председательствующий

Подпись

Секретарь

Подпись

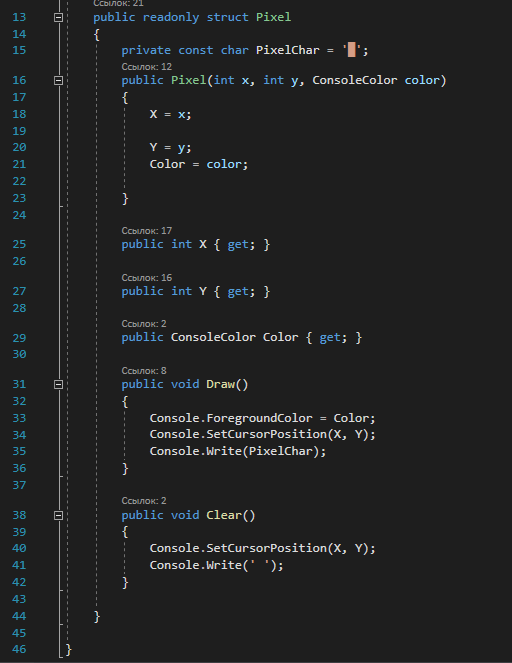
**Заседание**

Вопрос №1 . Ревьюирование программного кода в игре «Змейка»

Код программы состоит из нескольких классов, таких как:

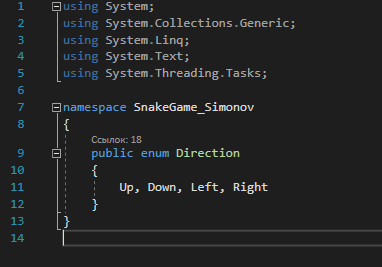
* Snake.cs
* Pixel.cs
* Direction.cs
* Program.cs

В классе Pixel.cs происходит создание графики для игры, за счет генерации и очистки символов по осям координат X,Y. Так же создается переменная для хранения цвета.



Ошибок в коде не наблюдается, код в исправлении не нуждается.

В Direction.cs хранится список направлений.



Ошибок в коде не наблюдается, код в исправлении не нуждается.

Класс Snake.cs инициализирует создание змейки и ее поведение. Он отвечает за ее отрисовку и перемещение.

\_headcolor и \_bodycolor хранят в себе цвет головы и тела змейки.

Голова змейки создается как объект pixel с координатами x и y.

Тело змейки создается как очередь из объектов pixel. Оно состоит из нескольких сегментов, которые располагаются слева от головы по оси х.

За очистку головы и тела змейки отвечает метод Clear.

Класс имеет следующий код:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace SnakeGame\_Simonov

{

public class Snake

{

private readonly ConsoleColor \_headColor;

private readonly ConsoleColor \_bodyColor;

public Snake(int initialX, int initialY, ConsoleColor headColor, ConsoleColor bodyColor, int bodyLength = 2)

{

\_headColor = headColor;

\_bodyColor = bodyColor;

Head = new Pixel(initialX, initialY, \_headColor);

for (int i=bodyLength; i >= 0; i--)

{

Body.Enqueue(new Pixel(Head.X - i -1, initialY, \_bodyColor));

}

Draw();

}

public Pixel Head { get; private set; }

public Queue<Pixel> Body { get; } = new Queue<Pixel>();

public void Move(Direction direction, bool eat = false)

{

Clear();

Body.Enqueue(new Pixel(Head.X, Head.Y, \_bodyColor));

if(!eat)

Body.Dequeue();

Head = direction switch

{

Direction.Right => new Pixel(Head.X + 1, Head.Y, \_headColor),

Direction.Left => new Pixel(Head.X - 1, Head.Y, \_headColor),

Direction.Up => new Pixel(Head.X, Head.Y - 1, \_headColor),

Direction.Down => new Pixel(Head.X, Head.Y + 1, \_headColor),

\_ => Head

};

Draw();

}

public void Draw()

{

Head.Draw();

foreach (Pixel pixel in Body)

{

pixel.Draw();

}

}

public void Clear()

{

Head.Clear();

foreach (Pixel pixel in Body)

{

pixel.Clear();

}

}

}

}

Функция Move() отсиживает движения змейки. Благодаря ему голова змейки может менять свое направление, а вслед за ней и тело.

Ошибок в коде не наблюдается, код в исправлении не нуждается.

Класс Program.cs отвечает за компоновку и объединение всех классов в полноценную игру.

С помощью числовых переменных задается ширина и высота карты. Задаются цвета для всех объектов.

При столкновении с объектом Food, змейка продолжает движения выполняя условие eat: true, пример кода:

if (snake.Head.X == food.X && snake.Head.Y == food.Y)

{

snake.Move(currentMovement, eat: true);

food = GenFood(snake);

food.Draw();

Score++;

}

else

{

snake.Move(currentMovement);

}

Если голова змейки сталкивается с ее телом или границами карты, то обработка события приводит к концу игры, пример кода:

if (snake.Head.X == MapWidth - 1 || snake.Head.X == 0 ||

snake.Head.Y == MapHeight - 1 || snake.Head.Y == 0 ||

snake.Body.Any(b => b.X == snake.Head.X && b.Y == snake.Head.Y))

break;

}

lagMS = (int)sw.ElapsedMilliseconds;

snake.Clear();

SetCursorPosition(MapWidth / 4, MapHeight / 2);

WriteLine($"Помер, Счёт: {Score}");

WriteLine("Для повтора нажмите Enter");

}

Функция GenFood() отвечает за генерацию еды путем проверки координат расположения змейки, пример кода:

static Pixel GenFood(Snake snake) // Генерация еды

{

Pixel food;

do

{

food = new Pixel(x:Random.Next(1, MapWidth - 2), y:Random.Next(1, MapHeight - 2), FoodColor);

} while (snake.Head.X == food.X && snake.Head.Y == food.Y

|| snake.Body.Any(b=> b.X == food.X && b.Y == food.Y));

return food;

}

Функция ReadMovement() отвечает за движение змейки, считывая нажатие клавиши, пример кода:

static Direction ReadMovement(Direction currentDirection ) // Считывание движения

{

if (!KeyAvailable)

return currentDirection;

var key = Console.ReadKey(true).Key;

currentDirection = key switch

{

ConsoleKey.UpArrow when currentDirection != Direction.Down => Direction.Up,

ConsoleKey.DownArrow when currentDirection != Direction.Up => Direction.Down,

ConsoleKey.LeftArrow when currentDirection != Direction.Right => Direction.Left,

ConsoleKey.RightArrow when currentDirection != Direction.Left => Direction.Right,

\_ => currentDirection

};

return currentDirection;

}

Генерация стен в игре в соответствии с заданными параметрами ширины и длины рамок, пример кода:

static void DrawBorder()

{

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

{

new Pixel(x:i, y:0, BorderColor).Draw();

new Pixel(x:i, y:MapHeight - 1, BorderColor).Draw();

}

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

{

new Pixel(x:0, y: i, BorderColor).Draw();

new Pixel(MapWidth - 1, y: i, BorderColor).Draw();

}

}

Вопрос №2 . Написание рецензии на программу

Оформление рецензии на программу на основании исходного кода и анализе первого вопроса.