Рязанский государственный радиотехнический университет им. В.Ф. Уткина

Кафедра «Вычислительная и прикладная математика» (ВПМ)

# Пояснительная записка к курсовому проекту

на тему: **«**Разработка игрового приложения Worms **»** по курсу

«Конструирование программного обеспечения»

Выполнил:

студент группы № 9413 Заикин С.Ю.

Проверил:

доцент кафедры ВПМ Столчнев В. К.

Рязань, 2023 г.

#### Оглавление

Вв	еден	ние		3
1	Ан	ализ	задачи	4
	1.1	Раз	работка иерархии классов	4
	1.1	l.1	Выделение сущностей	5
	1.1	1.2	Зависимости между классами. Диаграмма классов	5
	1.2	Алг	оритмы	8
	1.2	2.1	Алгоритм успешного передвижения червяка	8
	1.2	2.2	Алгоритм неудачного передвижения червяка	9
	1.2	2.3	Алгоритм столкновения червяка с поверхностью	.10
	1.2	2.4	Алгоритм столкновения снаряда базуки с червяком	.12
	1.3	Раз	работка интерфейса программы	.13
2	На	писан	ние программы	.15
	2.1	1.1	Описание разработанных процедур и функций	.15
	2.2	Раз	работка программы	.30
	2.2	2.1	Описание классов, перечислений и интерфейсов проекта	.30
	2.3	Опи	исание шаблонов проектирования, которые использовались при написании программы	ı 34
	2.3	3.1	Абстрактная фабрика (Abstract factory)	.34
	2.3	3.2	Одиночка (Singleton)	.34
	2.4	Опи	исание методов рефакторинга, которые использовались при оптимизации исходного ко	уда
	прог	рамм	Ы	.34
	2.4	4.1	Вынесение констант	.34
	2.4	1.2	Переименование метода	.35
	2.4	1.3	Вынесение кода в отдельный метод	.35
	2.5	Раз	работка тестов	.37
	2.5	5.1	Test Cases	.37
	2.5	5.2	Модульные тесты	.39
3	Pe	зульт	ат работы программы	.40
	3.1	Гра	фическая версия	.40
	3.2	Кон	сольная версия	.42
За	ключ	чение		.45
п,	NA 1101		1	15



#### Введение

Необходимо разработать игровое приложение Worms. Приложение должно иметь 2 реализации: консольную (используя консоль) и оконную (используя WPF). Необходимо использовать шаблон проектирования MVC (Модель-Вид-Контроллер) и два дополнительных шаблона проектирования. В приложении обязательно должна использоваться многопоточность и синхронизация между потоками. Для одного из классов приложения должны быть разработаны полноценные модульные тесты, обеспечивающие проверку корректности его работы.

Разработка производится с помощью языка программирования С# в среде разработки Visual Studio. Результатом работы является разработанная игра « Worms».



#### 1 Анализ задачи

Задача курсового проекта – разработка игрового приложения Worms.

Worms представляет собой пошаговую игру. В ней каждый игрок владеет армией червяков и инвентарем с оружием и вспомогательными предметами, которыми могут пользоваться червяки. Игроки по очереди выполняют ограниченный по времени ход, во время которого они должны попытаться нанести наибольший урон червякам противника, либо обезопасить своих червяков. Ход заканчивается тогда, когда текущий червяк получил урон или закончилось время на ход. После окончания своего хода игрок не может управлять червяком. Червяки могут двигаться вправо и влево, прыгать вперед и назад. Они погибают при здоровье, меньшем или равном 0, и при попадании в воду. Перед гибелью червяки производят Взрыв.

Целью игры является уничтожение всех червяков противника. Игра продолжается до тех пор, пока на поле имеются червяки по крайней мере 2-х разных коалиций или пока общий игровой таймер не дойдет до 0.

После окончания игры должно выводиться окно с результатом. Если установлен рекорд, запрашивается имя игрока.

В приложении должно быть меню, состоящее из пунктов: игра, рекорды, справка, выход.

#### 1.1 Разработка иерархии классов

Игровое приложение разрабатывается на основе шаблона проектирования MVC (модель-представление-контроллер). Визуальное представление шаблона MVC отображено на рисунке 1.

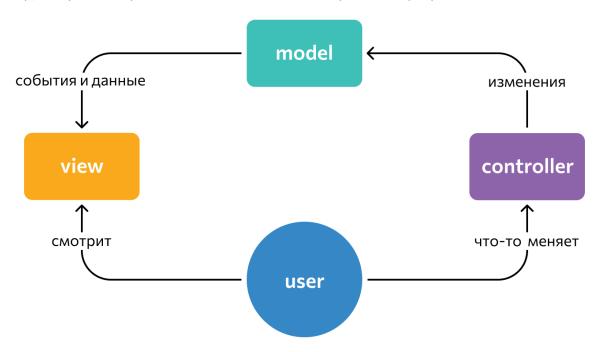


Рисунок 1 – Визуальное представление шаблона MVC

Концепция MVC разделяет данные, представление и обработку действий пользователя на компоненты:

- Модель предоставляет собой объектную модель некой предметной области, включает в себя данные и методы работы с этими данными, реагирует на запросы из контроллера, возвращая данные и/или изменяя своё состояние. При этом модель не содержит в себе информации о способах визуализации данных или форматах их представления, а также не взаимодействует с пользователем напрямую.
- Представление отвечает за отображение информации (визуализацию). Одни и те же данные могут представляться различными способами и в различных форматах.
- Контроллер обеспечивает связь между пользователем и системой, использует модель и представление для реализации необходимой реакции на действия пользователя.



Обощенно приложение состоит из следующих частей, базовые архитектуры для которых представлены на рисунке 2:

- меню;
- рекорды;
- справка;
- игра;
- конецигры.

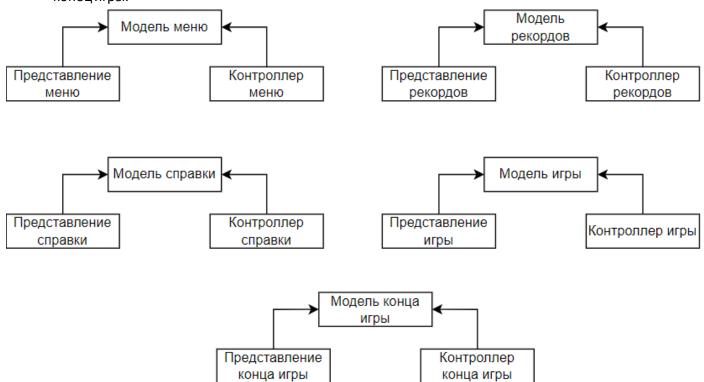


Рисунок 2 – Базовые архитектуры для основных частей приложения

#### 1.1.1 Выделение сущностей

Из предметной области игры Worms можно выделить следующие сущности:

- Червяк объект, умеющий пользоваться оружием, передвигаться и прыгать, которым управляет игрок.
- Оружие объект, который используется червяками для нанесения урона другим червякам и/или для разрушения карты.
- Физический объект объект, на который распространяется физика и логика игры.
- Инвентарь объект, хранящий список оружия.
- Снаряд физический объект, создаваемый оружием.

Для реализации шаблона MVC выделяются следующие сущности:

- Модель состояние объекта, которое может изменяться.
- Представление отображение модели.
- Контроллер управление моделью на основе действий пользователя.

#### 1.1.2 Зависимости между классами. Диаграмма классов

Диаграмма классов представлений приложения изображена на рисунке 3.



Рисунок 3 - Диаграмма классов представлений приложений

Диаграмма классов моделей приложения изображена на рисунке 4.

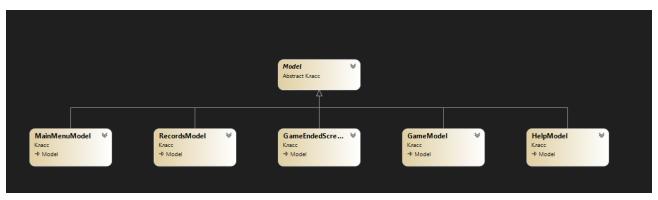


Рисунок 4 – Диаграмма классов моделей приложения

Диаграмма классов контроллеров приложения изображена на рисунке 5.

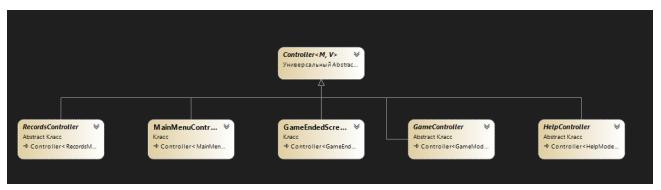


Рисунок 5 – Диаграмма классов контроллеров приложения

Диаграмма классов физических объектов изображена на рисунке 6.



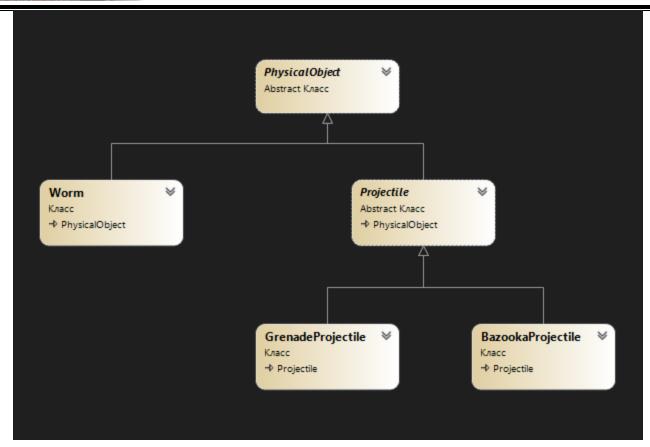


Рисунок 6 - Диаграмма классов физических объектов

Диаграмма классов оружия изображена на рисунке 7.



Рисунок 7 – Диаграмма классов оружия

Диаграмма классов представлений объектов изображена на рисунке 8.



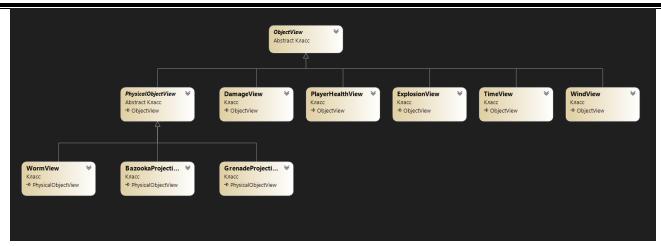


Рисунок 8 – Диаграмма классов представлений объектов

# 1.2 Алгоритмы

#### 1.2.1 Алгоритм успешного передвижения червяка

Логика передвижения червяка, при котором червяку не помешают препятствия, представлена на диаграмме последовательности успешного передвижения червяка (рисунок 9).



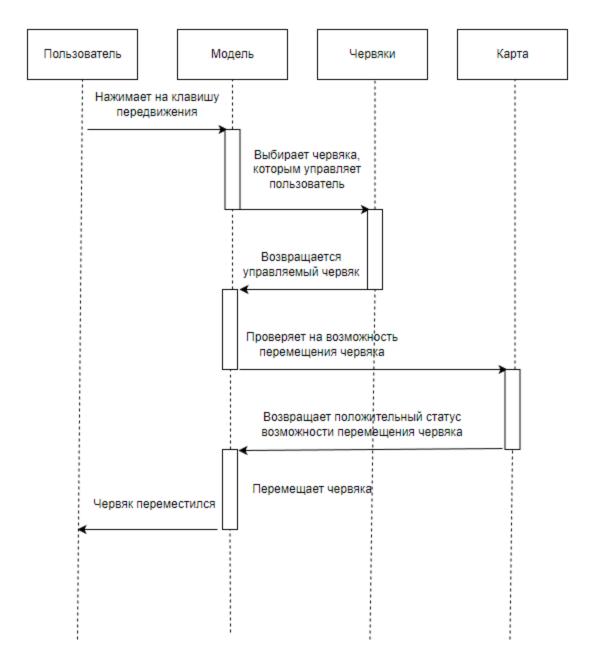


Рисунок 9 – Диаграмма последовательности успешного передвижения червяка

Как только пользователь нажмет клавишу А или D для передвижения червяком, модель выберет червяка, которым управляет игрок и попытается его переместить. Если червяку ничего не мешает, то он успешно переместится.

#### 1.2.2 Алгоритм неудачного передвижения червяка

Однако если червяк не может переместиться, то он останется на месте – данная логика представлена на диаграмме последовательности неудачного передвижения червяка (рисунок 10).



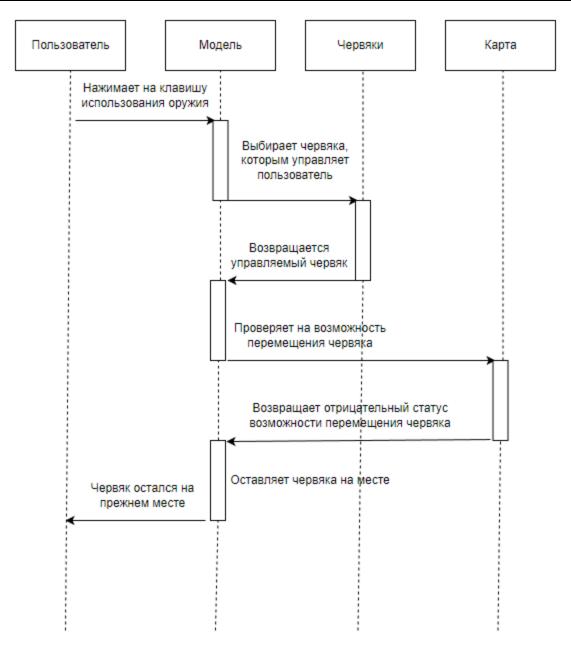


Рисунок 10 – Диаграмма последовательности неудачного передвижения червяка

#### 1.2.3 Алгоритм столкновения червяка с поверхностью

Логика столкновения червяка с поверхностью представлена на диаграмме последовательности столкновения червяка с поверхностью (рисунок 11).



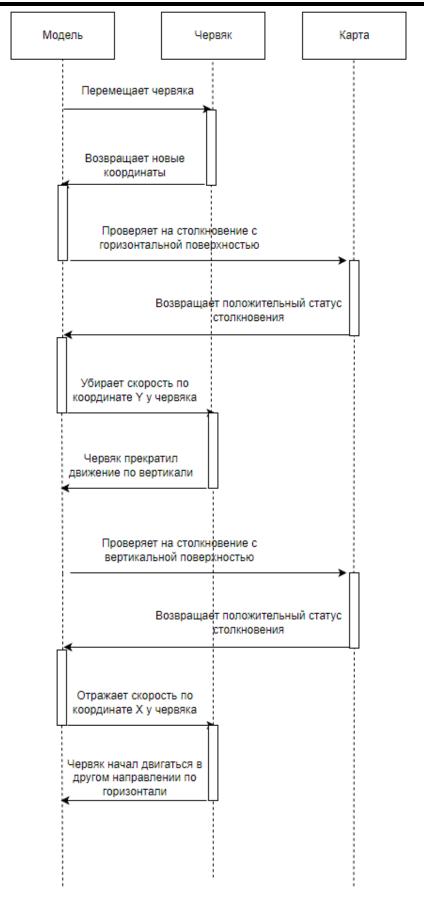


Рисунок 11 – Диаграмма последовательности столкновения червяка с поверхностью

Модель перемещает червяка в зависимости от его скорости по X и по Y. После этого она проверяет червяка на столкновение с горизонтальной поверхностью. Если он столкнулся с горизонтальной поверхностью, то прекращает его движение по вертикали. После этого она проверяет червяка на столкновение с вертикальной



поверхностью. Если червяк столкнулся с вертикальной поверхностью, то она отражает его скорость по X, из-за чего он начинает двигаться в обратном направлении по горизонтали.

#### 1.2.4 Алгоритм столкновения снаряда базуки с червяком

Логика столкновения снаряда базуки с червяком представлена на диаграмме последовательности столкновения снаряда базуки с червяком (рисунок 12).

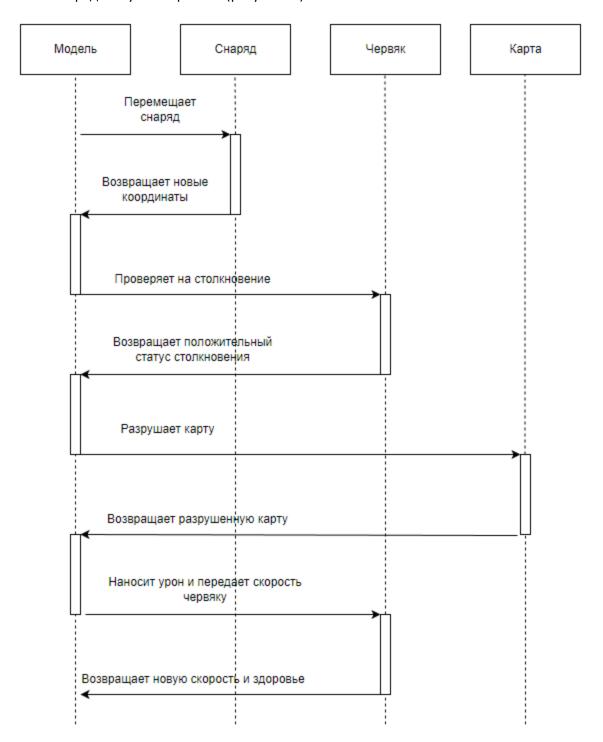


Рисунок 12 – Диаграмма последовательности столкновения снаряда базуки с червяком

Модель перемещает снаряд в зависимости от его скорости по X и по Y. После этого она проверяет снаряд базуки на столкновение с червяком. Если он столкнулся с червяком, то модель разрушает карту, а также наносит червяку урон и передает ему скорость по X и по Y.



# 1.3 Разработка интерфейса программы

Интерфейс главного меню программы состоит из 4 пунктов: играть, рекорды, справка, выход (рисунок 13).



Рисунок 13 – Интерфейс главного меню

Интерфейс справки содержит текстовую информацию (рисунок 14).

Справка. Нажмите Esc, чтобы вернуться в меню
<Текст справки>

Рисунок 14 – Интерфейс справки

Интерфейс таблицы рекордов содержит номер в списке, имя игрока и количество набранных очков (рисунок 15).



# Рекорды. Нажмите Esc, чтобы вернуться в меню 1. <Игрок> - <Очки> 2. <Игрок> - <Очки> ...

Рисунок 15 – Интерфейс таблицы рекордов

Интерфейс игры представлен на рисунке 16. На рисунке представлены следующие обозначения:

- прямоугольник-червяк;
- X таймер игры;
- N таймер хода;
- Н здоровье червяка;
- img иконка выбранного оружия;
- К количество оружия;
- стрелка над прямоугольником указатель на текущего червяка;
- стрелка сверху справа направление ветра.

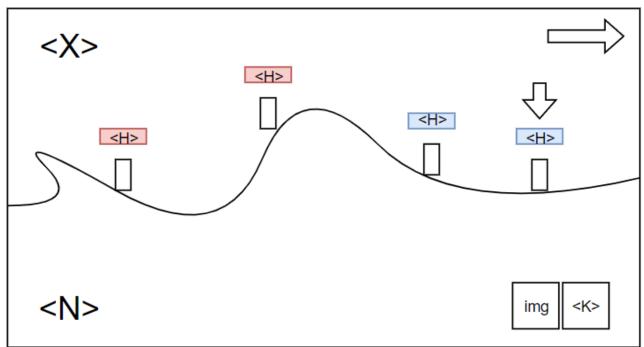


Рисунок 16 – Интерфейс игры

Интерфейс окна с результатом игры представлен на рисунке 17. В случае достижения рекорда выигравший игрок должен ввести свое имя.



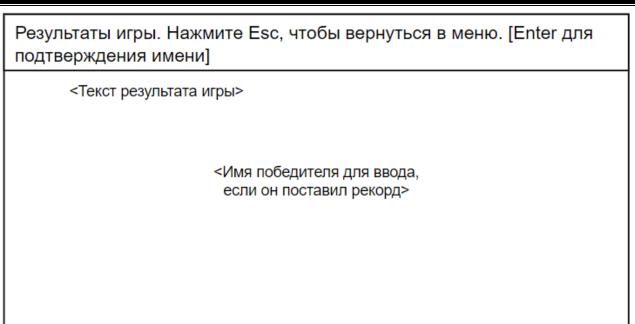


Рисунок 17 – Интерфейс окна с результатом игры

# 2 Написание программы

#### 2.1.1 Описание разработанных процедур и функций

Таблица 1 - Описание разработанных процедур и функций

Класс/Абстрактный класс	Метод	Описание	Параметры
	Worms Game	1	
	<pre>public abstract MainMenuController CreateMenuController()</pre>	Создает контроллер меню	-
	<pre>public abstract GameController CreateGameController()</pre>	Создает контроллер игры	-
A hatmatControllers Foatowy	<pre>public abstract RecordsController CreateRecordsController()</pre>	Создает контроллер рекордов	-
AbstractControllersFactory	<pre>public abstract HelpController CreateHelpController()</pre>	Создает контроллер справки	-
	<pre>public abstract GameEndedScreenController CreateGameEndedScreenController()</pre>	Создает контроллер экрана конца игры	-
	private void Init()	Инициализациру ет приложение	-
	<pre>private void InitMainMenuController()</pre>	Иницилизирует котроллер главного меню	-
Worms Application	<pre>private void InitGameController()</pre>	Иницилизирует котроллер игры	-
	<pre>private void InitRecordsController()</pre>	Иницилизирует котроллер рекордов	-
	<pre>private void InitHelpController()</pre>	Иницилизирует котроллер справки	-



	private void	Иницилизирует	
	InitGameEndedScreenController()	котроллер экрана	-
		конца игры	
	public virtual void Start()	Запускает	_
		приложение	
	Worms Game. BaseMVC		
	public virtual void Start()	Запускает	_
Controller M V	public virtual void Start()	контроллер	
Controller <m, v=""></m,>	<pre>public virtual void Stop()</pre>	Останавливает	
	public virtual void Stop()	контроллер	_
	multiplian all above at a social Durant	Рисует	
	<pre>public abstract void Draw()</pre>	представление	-
7.7 N.	121 1 2 1 2 1 2 1 2 1 1 2 1 1 2	Запускает	
View <m></m>	public virtual void Start()	представление	-
		Останавливает	
	public virtual void Stop()	представление	-
	Worms Game. Game. Controller	1 / /	<u> </u>
	Shine, Shine, Shine	Выбирает	
	private void SelectWorm(Worm	червяка для	parNextWorm -
	parNextWorm)	управления	червяк
		Возвращает в	
GameController	<pre>public void BackToMenu()</pre>	меню	-
Same Controller		Запускает	
	<pre>public override void Start()</pre>	контроллер	-
		Останавливает	
	<pre>public override void Stop()</pre>	контроллер	-
	Worms Game. Game. Model	контрольтер	
	World State: Galle: Would		
		Инициализирует	
	public static void Init()	игру	_
		Инициализирует	
	public static void InitPlayers()	игроков	-
	private static Inventory	Инициализирует	
	InitInventory()	инвентарь	-
	private static List <worm> InitWorms(Player parPlayer)</worm>	Инициализирует	
GameInitializer		червяков	parPlayer - игрок
		Получает Ү	
		червяка, при	
		котором он будет	
	private static int GetYWorm(int	стоять на	parX - координата
	parX)	поверхности	
		карты в	
		координате Х	
	public static void Start()	Запускает игру	-
		Останавливает	
	public static void Stop()	игру	-
	<pre>public static List<worm></worm></pre>	Получает всех	
	GetAllWorms()	червяков	-
		Добавляет новый	
	public static void	физический	parPhysicalObject
	AddObject (PhysicalObject	объект в список	- физический
	parPhysicalObject)	всех физических	объект
		объектов	
	public static void	Удаляет	parPhysicalObject
	RemoveObject(PhysicalObject	физический	- физический
		физический	4



GameModel  CameModel  Discrete that is void static voi		parPhysicalObject)	объект из списка	объект
GameModel    CameModel   CameM		[		1 2 2 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2
GumeModel    AddDamagedMoorm (Worm parWorm)   AddDDamagedMoorm (Worm)   AddDDAMagedMoorm (Wor			-	
GameModel  public static void AddDamagedWorm(Norm parNorm)  public static void RemovePamagedWorm(Norm parNorm)  private static void Static Word Static Wor				
### AddibmogedWorm(Norm parKorm)  ### AddibmogedWorm(Norm parKorm)  ### AddibmogedWorm(Norm parKorm)  ### Washer vepska  ### weepska  ### Washer vepska  ### weepska  ### weep			, ,	
AddbamagedWorm(Worm parWorm)   Topiskon, Korppskin Hallechi   ParWorm   Topiskon   ParWorm   Topiskon   ParWorm   Pa	GameModel	nublic static word	червяка в список	
SOTOPIAN HIRCORM   STATE TYPESTED			червяков,	parWorm - червяк
public static woid RemoveDamagedNorm(Norm parWorm)  private static void RemoveDamagedNorm(Norm parWorm)  private static void RemoveDayersWithoutNorms()  private static void RemoveDayersWithoutNorms()  public static void SndSame(Player  private static void SneNextPlayerTonde()  private static void SneNextPlayerTonde()  private static void RemoveDayerSund()  private static void RemoveDayerSund()  private static void PrepareNormForturn()  private static void PrepareNormForturn()  private static void PrepareNormSundsTrength()  private static void PrepareNormSundsTrength()  private static void RemoveDayerTond()  public static void NormSundsTrength()  public static void RemoveContron()  public static void NormSundsTrength()  public static void TurnCompletion()  public static void RemoveContron()  public static void RemoveContron()  public static void TurnCompletion()  public static void TurnCompletion()  public static void CountDown(object source, ElapsedEventArgs e)  public static void CountDown(object source, ElapsedEventArgs e)  public static void CountDown(object source, ElapsedEventArgs e)  public static void RemoveCourton()  RemoveDawaer  Public static void CountDown(object source, ElapsedEventArgs e)  public static void CountDown(object source, ElapsedEventArgs e)  public static void CountDown(object source, ElapsedEventArgs e)  public static void RemoveCourton()  RemoveDawaer  PropareNorm  Propare		naabamageaworm(worm parworm)	которым нанесли	
public static woid RemoveDamagedNorm(Norm parWorm)  private static void RemoveDamagedNorm(Norm parWorm)  private static void RemoveDayersWithoutNorms()  private static void RemoveDayersWithoutNorms()  public static void SndSame(Player  private static void SneNextPlayerTonde()  private static void SneNextPlayerTonde()  private static void RemoveDayerSund()  private static void RemoveDayerSund()  private static void PrepareNormForturn()  private static void PrepareNormForturn()  private static void PrepareNormSundsTrength()  private static void PrepareNormSundsTrength()  private static void RemoveDayerTond()  public static void NormSundsTrength()  public static void RemoveContron()  public static void NormSundsTrength()  public static void TurnCompletion()  public static void RemoveContron()  public static void RemoveContron()  public static void TurnCompletion()  public static void TurnCompletion()  public static void CountDown(object source, ElapsedEventArgs e)  public static void CountDown(object source, ElapsedEventArgs e)  public static void CountDown(object source, ElapsedEventArgs e)  public static void RemoveCourton()  RemoveDawaer  Public static void CountDown(object source, ElapsedEventArgs e)  public static void CountDown(object source, ElapsedEventArgs e)  public static void CountDown(object source, ElapsedEventArgs e)  public static void RemoveCourton()  RemoveDawaer  PropareNorm  Propare			урон	
public static void RemoveRemaqedWorm(Worm perWorm)  private static void RemovePlayersWithoutWorms()  private static void RemovePlayersWithoutWorms()  private static void EndGame(Player parFlayer)  public static void EndGame(Player parFlayer)  private static void SetNextPlayerIndex()  private static void PrepareWormForTurn()  private static void RemovePlayersWithoutWorms()  private static void PrepareWormForTurn()  private static void Remove				
public static void RemoveDamagedWorm(Worm parNorm)  private static void RemoveDamagedWorm(Worm parNorm)  private static void RemoveDayerWithoutWorms()  private static bool IsGameEnd()  public static void EndGame(Player parPlayer)  private static void EndGame(Player private static void GetNextPlayerIndex()  private static void GetNextPlayerIndex()  private static void RemoveDamagement private static void GetNextPlayerIndex()  private static void RemoveDamagement public static void RemoveDamagement public static void RemoveDamagement public static void RemoveDamagement public static void RemoveControl()  public static void RemoveControl()  public static void CountDown(object source, FlapsedEventArgs e)  private static void RemoveDamagement public static void CountDownGamage(object sender, ElapsedEventArgs e)  public static void RemoveControl()  COpachmant public static void RemoveDamagement public static void RemoveDamagement private static void RemoveDamagement private static void CountDownGamagement private static void RemoveDamagement private static void CountDownGamagement private static void Removement private static void Rem			•	
RemoveDamagedWorm(Worm parWorm) private static void RemovePlayersWithoutWorms()  private static void RemovePlayersWithoutWorms()  private static bool TaGameRend()  private static void EndGame(Player parPlayer)  public static void EndGame(Player parPlayer)  private static void PrepareMormPorturn()  private static void PrepareWormPorturn()  private static void PrepareWormPorturn()  private static void PrepareWormPorturn()  private static void PrepareMormPorturn()  private static void RetRandomWindStrength()  private static void PrepareMortPurn()  public static void PrepareMortPurn()  public static void RetRandomWindStrength()  public static void WaitingPorChangingTurn()  public static void WaitingPorChangingTurn()  public static void TurnCompletion()  public static void RemoveControl()  public static void TurnCompletion()  public static void TurnComple		public static void		narMorm - Honnau
private static void RemovePlayersWithoutWorms()  private static bool IsGameEnd()  private static bool IsGameEnd()  public static void EndGame(Player continue static woid EndGame(Player continue static woid Static woid with static woid Static woid with static woid Static woid Static woid Static woid PrepareWormPorTurn()  private static void PropareWormPorTurn()  private static void Static woid With static woid Static woid With static woid Static woid With static woid TurnCompletion()  public static woid TurnCompletion()  public static woid RemoveControl()  public static woid TurnCompletion()  public static woid TurnCompletion()  public static woid TurnCompletic word with static word CountDown(object word with static word CountDown(object word with with static word RemoveCountDown(object word with with word RemoveCountDown(object word with with word RemoveCountDown(object word with with word RemoveCountDown(object word with word RemoveCountDown(object word with RemoveCountDown(object word with RemoveCountDown(object word with RemoveCountDown(object word with RemoveCountDown(object word wi		RemoveDamagedWorm(Worm parWorm)	-	ратмотш - червяк
Private static void   SenovePlayermithoutNorms()   Sen uppnaxon   SenovePlayermithoutNorms()   SenovePlayerminate static void   SenovePlayerminate   SenovePla			_	
RemovePlayeraWithoutNorms()  Private static bool IsGameEnd()  public static void EndGame(Player consumus urpa    Public static void EndGame(Player consumus urpa   parplayer			· -	
private static bool IsGameEnd()  public static void EndGame(Flayer sakonumumeb. ии игра  public static void EndGame(Flayer coбытие overnauma игра  private static void  SetNextPlayerIndex()  private static void  SetNextPlayerIndex()  private static void  private static void  private static void  SetRandomWindStrength()  private static void  SetRandomWindStrength()  private static void  SetRandomWindStrength()  private static void  SetRandomWindStrength()  public static void  NaitingSorChangingTurn()  public static void  Samyckaer  oxilamine cwells xoxia  public static void  WaitingSorChangingTurn()  public static void TurnCompletion()  public static void TurnCompletion()  public static void RemoveControl()  public static void TurnCompletion()  public static void CountDown(object source, ElapsedEventArgs e)  private static void Samplex rupu orcere taifMepa  public static void Samplex rupu orcere taifMepa  public static void RemoveContril()  Bansmaers mpu  orcere taifMepa  public static void Samplex rupu orcere taifMepa  public static void RemoveContrilmer()  public static void RemoveContrilmer()		private static void	Удаляет игроков	_
private static bool IsGameEnd()  public static void EndGame(Flayer parFlayer)  public static void EndGame(Flayer coGmine onGegramumid empore onGegramumid engage onGeg		RemovePlayersWithoutWorms()	без червяков	
private static bool IsGameEnd()  public static void EndGame(Flayer parFlayer)  public static void EndGame(Flayer coGmine onGegramumid empore onGegramumid engage onGeg			Проверяет	
### public static void EndGsme(Player parPlayer)    Public static void EndGsme(Player parPlayer)   ParPlayer   Pa		private static bool IsGameEnd()	* *	_
public static void EndGame (Player parPlayer)         BASMBBET COGNITURE (COGNITURE ON CONTINUATION)         parPlayer - nodeдивший игрох           private static void SetNextPlayerIndex()         Veranammuser индек (следующего червяжа         - следующего червяжа           private static void PrepareNormForTurn()         Indomoranumaer червяжа для хода (следующий ход (следующи		- ''		
public static void EndSame(Player cooburne okonwannamum игрок    ParPlayer   ParPlayer   ParPlayer   ParPlayer   ParPlayer   ParPlayer   Pooceximens with proximate static void   Windex   Proximate static void   Mingrox   Proximate static void   Mingrox   Proximate static void   Mingrox   Proximate static void   Mingrox   Proximate static void   Proximate static v			-	
parPlayer)  окончания игры  устанавливает индекс спедующего червяка  ргіvate static void  ргіvate static void  прираговорна при		public static void EndGame(Player		parPlayer -
private static void SetNextPlayerIndex()  private static void  private static void  private static void  private static void  private static void StartNextTurn()  private static void StartNextTurn()  public static void  private static void  public static worm GetCurrentWorm()  public static void TurnCompletion()  public static void RemoveControl()  public static void TurnCompletion()  Bushasacra при orcuere таймера voice taimepa voice		parPlayer)		победивший игрок
private static void SetNextFlayerIndex()  private static void private static void private static void PrepareWormForTurn()  private static void StartNextTurn()  private static void StartNextTurn()  public static void SetRandomWindStrength()  private static void SetRandomWindStrength()  private static void PrepareNextTurn()  public static void WaitingForChangingTurn()  public static Worm GetCurrentWorm()  public static void TurnCompletion()  public static void TurnCompletion()  public static void RemoveControl()  ferymum quepaskom public static void InitTimers()  public static void CountDown(object source, ElapsedEventArgs e)  public static void ResetTurnTimer()  public static void ResetTurnTimer()  public static void ResetTurnTimer()  CopachBaet  CopachBaet  CopachBaet  CopachBaet  Public static void ResetTurnTimer()  CopachBaet  Public static void ResetTurnTimer()				
SetNextPlayerIndex()  private static void private static void PrepareMormforTurn()  private static void StartNextTurn()  public static void StartNextTurn()  SetRandomWindStrength()  private static void SetRandomWindStrength()  private static void SetRandomWindStrength()  private static void PrepareNextTurn()  public static void WaitingForChangingTurn()  public static void  public static worm GetCurrentWorm()  public static Worm GetCurrentWorm()  public static void TurnCompletion()  Sampwart  public static void TurnCompletion()  Sampmart xop  your passeom  public static void RemoveControl()  Formposs Hand Texymum Te			Устанавливает	
ргіvate static void PrepareWormForTurn()  private static void StartNextTurn()  private static void StartNextTurn()  public static void SetRandomWindStrength()  private static void PrepareNextTurn()  public static void PrepareNextTurn()  public static void PrepareNextTurn()  public static void WaitingForChangingTurn()  public static Worm GetCurrentWorm()  public static woid TurnCompletion()  public static void TurnCompletion()  public static void RemoveControl()  public static void RemoveControl()  GameTimers  GameTimers  Texpuero  public static void CountDown(object source, ElapsedEventArgs e)  public static void ResetTurnTimer()  public static void ResetTurnTimer()  public static void ResetTurnTimer()  Copacusaer  - CeneyNomuMi xoд - CeneyNomuMi xod - CeneyNomuMi xod - CeneyNomuMi xod - Cene			индекс	_
private static void PrepareWormForTurn() private static void StartNextTurn()  private static void StartNextTurn()  public static void SetRandomWindStrength()  private static void PrepareNextTurn()  private static void PrepareNextTurn()  public static void PrepareNextTurn()  public static void WaitingForChangingTurn()  public static Worm GetCurrentWorm()  public static Worm GetCurrentWorm()  public static void TurnCompletion()  PrepareNextTurn()  public static void TurnCompletion()  Absolute the void TurnCompletion()  Samepumer void Wongar very with the void very with the very with		SetNextPlayerIndex()	следующего	
private static void PrepareWormForTurn()  private static void StartNextTurn()  private static void StartNextTurn()  public static void SetRandomWindStrength()  private static void SetRandomWindStrength()  private static void PrepareNextTurn()  public static void WaitingForChangingTurn()  public static worm GetCurrentWorm()  public static word TurnCompletion()  public static void RemoveControl()  public static void RemoveControl()  public static void InitTimers()  public static void CountDown(object source, ElapsedEventArgs e)  public static void ResetTurnTimer()  public static void ResetTurnTimer()  public static void ResetTurnTimer()  public static void ResetTurnTimer()  CGpacabaer  - Agnycaar  - Careyvonuni xoд - Crayvainus			червяка	
PrepareWormForTurn()  private static void StartNextTurn()  public static void SetRandomWindStrength()  private static void SetRandomWindStrength()  private static void PrepareNextTurn()  public static void PrepareNextTurn()  public static void WaitingForChangingTurn()  public static Worm GetCurrentWorm()  public static Worm GetCurrentWorm()  public static void TurnCompletion()  public static void RemoveControl()  public static void RemoveControl()  public static void InitTimers()  public static void CountDown(object source, ElapsedEventArgs e)  public static void ResetTurnTimer()  public static void ResetTurnTimer()  public static void ResetTurnTimer()  public static void ResetTurnTimer()  CGpachBaet  CGpachBaet  - CEAPNORMEM A  - CEAPNORME		private static void		
private static void StartNextTurn()    Sanyckaer   Cncgyomuni жод   Pytanosmmsaer   Cnyvainyo cuny   SetRandomWindStrength()   Sanyckaer   Cncgyomuni жод   Sanyckaer   Oxoga   O				_
Private static void StartNextTurn()  public static void SetRandomWindStrength()  PrepareNextTurn()  public static void PrepareNextTurn()  public static void WaitingForChangingTurn()  public static Worm GetCurrentWorm()  public static void TurnCompletion()  public static void TurnCompletion()  public static void RemoveControl()  GameTimers  Private static void CountDown(object source, ElapsedEventArgs e)  public static void ResetTurnTimer()  Public static void ResetTurnTimer()  Public static void ResetTurnTimer()  Copachman  Conceptomina xoд  Vorahoвливает  - следующий ход  Валускает  ожидание смены ход  гекущего  управляемого  червика  - червика  Получает  текущего  управляемого  червика  - червика  Инициализация  - таймеров  Вызывается при отечете таймера хода  Вызывается при отечете таймера колечете таймера игры  public static void ResetTurnTimer()  Public static void ResetTurnTimer()				
public static void SetRandomWindStrength() SetRandomWindStrength() SetRandomWindStrength() SetRandomWindStrength() SetRandomWindStrength() SetPa  private static void PrepareNextTurn()  Samyckaer Oжидание смены жода  Получает текущего управляемого червика  public static void TurnCompletion()  SameDiaer жод  public static void RemoveControl()  Fund Static void TurnCompletion()  SameDiaer жод  public static void InitTimers()  SameDiaer жод  public static void CountDown(object source, ElapsedEventArgs e)  public static void ResetTurnTimer()  SetRandomWindStrength()  Fund Static void ResetTurnTimer()  Poblic static void ResetTurnTimer()  Copacidade  ContaboundSame(object sender, ElapsedEventArgs e)  Fund Static void ResetTurnTimer()  Copacidade  Copacidade  Copacidade  Copacidade  -  Copacidade		<pre>private static void StartNextTurn()</pre>		-
public static void SetRandomWindStrength()       случайную силу ветра       -         private static void PrepareNextTurn()       Подготавливает следующий ход       -         public static void WaitingForChangingTurn()       Запускает ожидание смены хода       -         public static Worm GetCurrentWorm()       Получает текущего управляемого червяка       -         public static void TurnCompletion()       Завершает ход       -         Vбирает контроль над текущим червяком       -         public static void RemoveControl()       Инициализация таймеров       -         public static void CountDown(object source, ElapsedEventArgs e)       Вызывается при отсчете таймера хода       -         private static void CountDownGame (object sender, ElapsedEventArgs e)       Вызывается при отсчете таймера игры       -         public static void ResetTurnTimer()       Сбрасывает       -				
SetRandomWindStrength()  Private static void  PrepareNextTurn()  public static void  WaitingForChangingTurn()  Public static Worm GetCurrentWorm()  Public static void TurnCompletion()  Public static void TurnCompletion()  Public static void RemoveControl()  Sabephaer xod  Public static void RemoveControl()  Public static void InitTimers()  CameTimers  Public static void CountDown(object source, ElapsedEventArgs e)  Public static void ResetTurnTimer()  Public static void ResetTurnTimer()  Public static void ResetTurnTimer()  Concept and ResetTurnTimer()		public static void		
ргіvate static void PrepareNextTurn()  public static void WaitingForChangingTurn()  public static Worm GetCurrentWorm()  public static Worm GetCurrentWorm()  public static void TurnCompletion()  public static void RemoveControl()  GameTimers  Betpa    Полутавти сисны кода   Получает текущего управляемого червяка		-	случайную силу	-
PrepareNextTurn()  public static void  WaitingForChangingTurn()  public static Worm GetCurrentWorm()  public static Worm GetCurrentWorm()  public static woid TurnCompletion()  public static void TurnCompletion()  ABREPHIMET XOД  public static void RemoveControl()  public static void RemoveControl()  Formal Static woid InitTimers()  public static void CountDown(object source, ElapsedEventArgs e)  public static void RemoveControl()  Formal Static woid CountDownCompletion()  BUSINBARETS INDU  TOWNERS TO THE TOWNERS INDU  TOWNERS TOWNERS INDU  TOWNERS TOWNERS TOWNERS TOWNERS INDU  TOWNERS TOW			ветра	
CameTimers       public static void WaitingForChangingTurn()       Запускает ожидание смены хода       -         Public static Worm GetCurrentWorm()       Получает текущего управляемого червяка       -         public static void TurnCompletion()       Завершает ход -         Vбирает контроль над текущим червяком       -         public static void RemoveControl()       Инициализация таймеров         public static void TnitTimers()       Вызывается при отсчете таймера хода         private static void CountDown(object source, ElapsedEventArgs e)       Вызывается при отсчете таймера хода         private static void CountDownGame(object sender, ElapsedEventArgs e)       Вызывается при отсчете таймера игры         public static void ResetTurnTimer()       Сбрасывает		private static void	Подготавливает	
CameTimers       public static void WaitingForChangingTurn()       Запускает ожидание смены хода       -         Public static Worm GetCurrentWorm()       Получает текущего управляемого червяка       -         public static void TurnCompletion()       Завершает ход -         Vбирает контроль над текущим червяком       -         public static void RemoveControl()       Инициализация таймеров         public static void TnitTimers()       Вызывается при отсчете таймера хода         private static void CountDown(object source, ElapsedEventArgs e)       Вызывается при отсчете таймера хода         private static void CountDownGame(object sender, ElapsedEventArgs e)       Вызывается при отсчете таймера игры         public static void ResetTurnTimer()       Сбрасывает		PrepareNextTurn()	следующий хол	_
GameTimers				
WaitingForChangingTurn()   xoда		_	*	_
Diagram   Dia		WaitingForChangingTurn()	. ,	
public static Worm GetCurrentWorm()       текущего управляемого червяка       -         рublic static void TurnCompletion()       Завершает ход       -         Убирает контроль над текущим червяком       -         рublic static void RemoveControl()       Иницализация таймеров       -         рublic static void InitTimers()       Иницализация таймеров       -         рublic static void CountDown(object source, ElapsedEventArgs e)       Вызывается при отсчете таймера хода       -         рrivate static void CountDownGame(object sender, ElapsedEventArgs e)       Вызывается при отсчете таймера игры       -         рublic static void ResetTurnTimer()       Сбрасывает       -			<u>' ' '</u>	
Public static worm Getcurrentworm()  yправляемого червяка  public static void TurnCompletion()  3авершает ход  Убирает контроль над текущим червяком  Public static void InitTimers()  Public static void CountDown(object source, ElapsedEventArgs e)  Private static void CountDownGame(object sender, ElapsedEventArgs e)  Public static void RemoveControl()  Bызывается при отсчете таймера хода  Private static void CountDownGame(object sender, ElapsedEventArgs e)  Public static void ResetTurnTimer()  Copacывает			•	
управляемого червяка  public static void TurnCompletion()  3aвершает ход  Убирает контроль над текущим червяком  public static void InitTimers()  Taймеров  Bызывается при отсчете таймера хода  private static void CountDownGame(object sender, ElapsedEventArgs e)  public static void RemoveControl()  Fundic static void CountDown(object source, ElapsedEventArgs e)  public static void CountDownGame(object sender, ElapsedEventArgs e)  public static void ResetTurnTimer()  Copacывает  -  -  Copacывает  -  Copacывает  -  Copacывает  -  Copacывает  -  -  Copacывает  -  Copacывае		public static Worm GetCurrentWorm()	текущего	_
GameTimers       public static void TurnCompletion()       Завершает ход       -         Public static void RemoveControl()       Контроль над текущим червяком       -         public static void InitTimers()       Инициализация таймеров       -         public static void CountDown(object source, ElapsedEventArgs e)       Вызывается при отсчете таймера хода       -         private static void CountDownGame(object sender, ElapsedEventArgs e)       Вызывается при отсчете таймера игры       -         public static void ResetTurnTimer()       Сбрасывает       -			управляемого	
Public static void RemoveControl()  Public static void RemoveControl()  Public static void InitTimers()  Public static void CountDown(object source, ElapsedEventArgs e)  Private static void  CountDownGame(object sender, ElapsedEventArgs e)  Public static void ResetTurnTimer()  Copacывает  Copacывает  Copacывает  Copacывает  Copacывает  Public static void ResetTurnTimer()			червяка	
Public static void RemoveControl()  Public static void RemoveControl()  Public static void InitTimers()  Public static void InitTimers()  Public static void CountDown(object source, ElapsedEventArgs e)  Private static void  CountDownGame(object sender, ElapsedEventArgs e)  Public static void ResetTurnTimer()  CopachBaet  CopachBaet  CopachBaet  CopachBaet  CopachBaet  CopachBaet  CopachBaet  Public static void ResetTurnTimer()		<pre>public static void TurnCompletion()</pre>	Завершает ход	-
рublic static void RemoveControl()       контроль над текущим червяком         рublic static void InitTimers()       Инициализация таймеров         Вызывается при отсчете таймера хода         рublic static void CountDown(object source, ElapsedEventArgs e)       Вызывается при отсчете таймера гаймера игры         рrivate static void CountDownGame(object sender, ElapsedEventArgs e)       Вызывается при отсчете таймера игры         public static void ResetTurnTimer()       Сбрасывает			_	
Public static void RemoveControl()  текущим червяком  рublic static void InitTimers()  Public static void CountDown(object source, ElapsedEventArgs e)  Private static void CountDownGame(object sender, ElapsedEventArgs e)  Public static void CountDownGame(object sender, ElapsedEventArgs e)  Public static void ResetTurnTimer()  Copacывает			-	
GameTimers		<pre>public static void RemoveControl()</pre>	=	-
Public static void InitTimers()       Инициализация таймеров         Public static void CountDown(object source, ElapsedEventArgs e)       Вызывается при отсчете таймера хода         Private static void CountDownGame(object sender, ElapsedEventArgs e)       Вызывается при отсчете таймера отсчете таймера игры         Public static void ResetTurnTimer()       Сбрасывает			-	
Public static void InitTimers()  Taймеров  Public static void CountDown(object source, ElapsedEventArgs e)  Private static void  CountDownGame(object sender, otcuete таймера ElapsedEventArgs e)  Public static void ResetTurnTimer()  Copacывает			_	
GameTimers       рublic static void CountDown(object source, ElapsedEventArgs e)       Вызывается при отсчете таймера хода       -         private static void CountDownGame(object sender, ElapsedEventArgs e)       Вызывается при отсчете таймера игры       -         public static void ResetTurnTimer()       Сбрасывает       -		<pre>public static void InitTimers()</pre>		_
GameTimers    Public static void CountDown(object source, ElapsedEventArgs e)   Отсчете таймера хода			таймеров	
GameTimers       source, ElapsedEventArgs e)       отсчете таимера хода       -         private static void CountDownGame(object sender, ElapsedEventArgs e)       отсчете таймера игры       -         public static void ResetTurnTimer()       Сбрасывает       -			Вызывается при	
Source, ErapsedEventArgs e)   хода   рrivate static void   Вызывается при   СоuntDownGame(object sender, отсчете таймера   ЕгapsedEventArgs e)   игры   рublic static void ResetTurnTimer()   Сбрасывает   -	G		отсчете таймера	-
private static void CountDownGame(object sender, отсчете таймера ElapsedEventArgs e)  public static void ResetTurnTimer()  Bызывается при отсчете таймера игры  -	GameTimers	Source, LiapsedEventArgs e)	•	
CountDownGame(object sender, отсчете таймера - ыгры  public static void ResetTurnTimer()  Contract таймера - ыгры  Сбрасывает -		private static void		
ElapsedEventArgs e) игры  public static void ResetTurnTimer() Сбрасывает			_	_
public static void ResetTurnTimer()  Сбрасывает -			-	_
public static void ResetTurnTimer()			_	
таймер хода		<pre>public static void ResetTurnTimer()</pre>	Сбрасывает	_
		The state of the s	таймер хода	



			ı
	<pre>public static void StopTimers()</pre>	Останавливает таймеры	-
	public static void	Читает карту из	parFilename -
MapReader	ReadMapFromFile(string parFilename)	файла	название файла
		Получает	
	<pre>public int GetCenterX()</pre>	координату Х	-
		центра объекта	
		Получает	
	<pre>public int GetCenterY()</pre>	координату Ү	_
	public ine detecneeli ()	* *	
		центра объекта	
		Проверяет	
	public bool IsOnGround()	находится ли на	_
	· ·	поверхности	
		объект	
		Проверяет	parX - координата
PhysicalObject	public bool IsCollapse(int parX, int	столкнулся ли с	Х,
	parY)	картой объект	parY - координата
		_	Y
		Обновляет	
	public void Update()	состояние	-
		объекта	
		Обрабатывает	
	<pre>public abstract void OnCollapse()</pre>	столкновения с	-
		картой	
		Обрабатывает	
	public abstract void	-	_
	OnGoingOutOfBounds()	выхода за	_
		границы карты	
		Получает	
	<pre>public Worm GetCurrentWorm()</pre>	управляемого	-
		червяка	
Player		Выбирает	
	nublic resid Nove House ()	следующего	
	<pre>public void NextWorm()</pre>	червяка для	-
		управления	
		Установливает	
		случайное	
	private void SetRandomState()	состояние	-
		червяка	
		Обрабатывает	
	<pre>public override void OnCollapse()</pre>	столкновения с	-
		картой	
	public override void	Обрабатывает	
	OnGoingOutOfBounds()	выхода за	-
		границы карты	
		Провяеряет	
	<pre>private bool IsWormControlling()</pre>	управляется ли	-
	_	червяк	
		Убивает червяка	
		без взрыва,	
		•	
	public void Kill()	используется при	-
		выходе червяка	
		за границы	
		экрана	
***	<pre>public void KillWithExplosion()</pre>	Убивает червяка	_
Worm		со взрывом	
	private int FindObstacleHeight(int	Находит высоту	рагX — координата X
	parX)	палодии высоту	- s noop,mara n
<del></del>		<del></del>	·



		препятствия в	
		координате Х	
		Червяк делает	
	<pre>public void MoveRight()</pre>	шаг вправо	-
		Червяк пытается	
	private void TryRightMove()	передвинуть	_
		червяка вправо	
		Червяк делает	
	<pre>public void MoveLeft()</pre>	шаг влево	-
		Червяк пытается	
	<pre>private void TryLeftMove()</pre>	передвинуть	_
		червяка влево	
		Червяк	
	<pre>public void JumpForward()</pre>	выполняет	-
		прыжок вперед	
		Червяк	
	public void BackFlip()	выполняет	-
		прыжок назад	
		Червяк	
	<pre>public void UseWeapon()</pre>	использует	-
		оружие	
	Worms Game. Game. Model. We apons	1.0	
Bazooka	public override void Use (Worm	Использует	W
	parWorm)	базуку	parWorm - червяк
		Обрабатывает	
BazookaProjectile	<pre>public override void OnCollapse()</pre>	столкновения с	_
		картой	
		-	рагХ - X центра взрыва, рагҮ - Y центра взрыва,
	<pre>public static void MakeExplosion(int parX, int parY, int parRadius, int parTransmittedSpeed, int parDamage)</pre>	Создает взрыв	parRadius - радиус
			взрыва,
			parTransmittedSpeed -
			передаваемая скорость, parDamage - урон
			раг Даппаде - урон раг X - X центра взрыва,
	private static void		рагУ - У центра взрыва,
	<pre>MapDestruction(int parX, int parY, int parRadius)</pre>	Разрушает карту	parRadius - радиус
	ine painaurus)		взрыва
			рагХ - X центра взрыва,
	private static void	Проявляет	parY - Y центра взрыва, parRadius - радиус
	WormsInfluence(int parX, int parY,	воздействие на	взрыва,
	<pre>int parRadius, int parTransmittedSpeed, int parDamage)</pre>	червяков	parTransmittedSpeed -
ExplosionMaker	paritansmittedspeed, int parbamage)		передаваемая скорость,
			parDamage - урон
			parX1 -
		_	координата X1,
	private static double	Получает	координата Ү1,
	GetDistance(int parX1, int parY1,	расстояние	parX2 -
	int parX2, int parY2)	между тчоками	координата Х2,
			parY2 -
			координата Y2 parXPoint - X
			TOURN,
	<pre>public static bool IsInExplosion(int parXPoint, int parYPoint, int parXExplosion, int parYExplosion, int parRadius)</pre>	Проверяет	parYPoint - Y
		находится ли	точки,
		точка в радиусе	parXExplosion - X
		взрыва	центра взрыва,
			parYExplosion - Y центра взрыва,
		1	1



			parRadius -
			рагкастиз -
~ .	public override void Use(Worm	Использует	
Grenade	parWorm)	гранату	parWorm - червяк
GrenadeProjectile	public void CountDown(object source,	Обрабатывает	
2-3114001 13,000110	System.Timers.ElapsedEventArgs e)	отсчет таймера	-
		Обрабатывает	
	<pre>public override void OnCollapse()</pre>	столкновения с	_
		картой	
		Обрабатывает	
	public override void	выход за	
Projectile	OnGoingOutOfBounds()	границы игровой	
	(,		
		карты	
	<pre>public void RiseAngle()</pre>	Поднимает угол	-
		оружия	
	<pre>public void DownAngle()</pre>	Опускает угол	-
	-	оружия	
	<pre>public void PowerUp()</pre>	Добавляет заряд	_
Weapon		оружия	
		Устанавливает	
	<pre>public void SetInitialState()</pre>	начальное	-
		состояние	
	public abstract void Use(Worm	Использует	narWorm
	parWorm)	оружие	parWorm - червяк
	Worms Game. Game Ende dS creen. Contro		
	public void SetGameResult(Player	Устанавливает	parPlayer - игрок,
	parPlayer, int parScore)	результат игры	parScore - очки
GameEndedScreenController		Возвращает в	
	public void BackToMenu()	меню	-
	Worms Game. Game En de dS creen. Mod		<u> </u>
		Установает	
	<pre>public void SetIsRecord()</pre>	свойство рекорд	_
		ли	
GameEndedScreenModel		Получает текст	
	<pre>public string GetEndText()</pre>	конца игры	-
		Сохраняет	
	<pre>public bool SaveRecord()</pre>	_	-
	Worms Game. Game Ended Screen. Vie	рекорд	
	worms Game. Game Ende discreen. V16		1
GameEndedScreenView	private void Redraw()	Перерисовать	-
		представление	
	Worms Game. Help. Controller		T
HelpController	ublic void BackToMenu()	Возвращает в	-
		меню	
	Worms Game. Help. Model	Ι	T
	<pre>public string GetHelpText()</pre>	Получает текст	_
	, , , , , , , , , , , , , , , , , , ,	справки	
HelpModel		Вызывает	
1 icipiviouci	<pre>public void NeedRedraw()</pre>	событие о	
	Paptic vota Meedvediam()	необходимости	_
		перерисовки	
	Worms Game. Help. View	•	•
Перерисовывает			
HelpView	private void Redraw()	представление	-
•		справки	
	Worms Game. Main Menu	1 -	l
Menu	public MenuItem	Получает пункт	parIndex - пункт
1.10110	<u> </u>	1 1001 115 1161	



	GetMenuItemByIndex(int parIndex)		меню
	Worms Game, Main Menu. Controller	меню по индексу	Meno
	worms Game, Main Menu. Controller		T
		Запускает	
	<pre>public override void Start()</pre>	контроллер	-
		главного меню	
MainMenuController		Останавливает	
Transitiona Controller	<pre>public override void Stop()</pre>	контроллер	-
		главного меню	
		Обрабатывает	
	private void MenuClickHandler(MenuItem	нажатия на	parSelectedMenuIt
	parSelectedMenuItem)	выбранный	em - выбранный пункт меню
	parsored sound nar com,	пункт меню	
	Worms Game. Main Menu. Model	·	
		Выбирает	
	<pre>public void NextMenuItem()</pre>	следующий	_
		пункт меню	
		Выбирает	
	<pre>public void PreviousMenuItem()</pre>	предыдущий	_
	paste total feviousimitem()	пункт меню	
	public void FocusMenuItem(int	Выделяет пункт	nanTnda
MainMenuModel	public void FocusMenuitem(int parIndex)	меню	parIndex - пункт меню
	parinacky		MCIIIO
	muhlin maid Datasa()	Нажимает на	
	<pre>public void Enter()</pre>	выбранный	-
		пункт меню	
		Вызывает	
	<pre>public void NeedRedraw()</pre>	событие о	_
	public void necalediam ()	необходимости	
		перерисовки	
	public static string	Получает текст	parMenuItem -
MenuItemTextFormer	GetTitle (MenuItem parMenuItem)	для пункта меню	пункт меню
	Worms Game. Main Menu. View		
		Перерисовывает	
MainMenuView	private void Redraw()	представление	_
		главного меню	
	Worms Game. Rec or ds	Transpirer of Service	
	,, 32112 SIMMOVATOR 02 W	Проверяет	
		является ли	
	<pre>public bool IsRecord(int parScore)</pre>	рекордом данное	parScore - очки
		количество очков	
	public void SaveRecord(GameRecord	Сохраняет	parRecord -
	parRecord)	_	рагкесога -
Pagarda Papas itam	public void	рекорд	
RecordsRepository	SaveRecords (List <gamerecord></gamerecord>	Сохраняет	parRecords -
	parRecords)	рекорды	рекорды
	<pre>public List<gamerecord></gamerecord></pre>	Получает все	
	GetAllRecords()	рекорды	-
	private void	Сортирует	narPecords
	SortRecords (List <gamerecord></gamerecord>	= = -	parRecords - рекорды
	parRecords)	рекорды	11,,
Worms Game.Rec or ds. Controller			
		Запускает	
	<pre>public override void Start()</pre>	контроллер	-
		рекордов	
Records Controller	<pre>public void BackToMenu()</pre>	Возвращает в	_
		меню	
	Worms Game. Rec or ds . Model		
			•



		_	
		Получает строку,	
GameRecord	<pre>public override string ToString()</pre>	представляющую	-
		объект	
		Вызывает	
		событие о	
	<pre>public void NeedRedraw()</pre>	необходимости	-
		перерисовки	
RecordsModel			
	<pre>public List<gamerecord> GetRecords()</gamerecord></pre>	Получает	-
		рекорды	
	<pre>public void UpdateRecords()</pre>	Обновляет	_
	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	рекорды	
	WormsWPF		
	public override GameController	Создает	
	CreateGameController()	контроллер игры	-
		Создает	
	public override	контроллер	
	GameEndedScreenController		-
	<pre>CreateGameEndedScreenController()</pre>	экрана конца	
WPFControllersFactory		игры	
	public override HelpController	Создает	-
	CreateHelpController()	контроллер	
	<u> </u>	справки	
	public override MainMenuController	Создает	-
	CreateMenuController()	контроллер меню	
		Создает	_
	public override RecordsController	контроллер	
	CreateRecordsController()	рекордов	
	<pre>private void ConfigureWindow()</pre>	Конфигурирует	-
WPFW orms Application		ОКНО	
r r	<pre>public override void Start()</pre>	Запускает	-
		приложение	
	Worms WPF.Game.Controller		
	private void OnClose(object? sender,	Обработывает	
	EventArgs e)	закрытие окна	-
	<pre>public void KeyDownHandler(object sender, KeyEventArgs e)</pre>	Обработывает	
		нажатие клавиши	_
		клавиатуры	
		• 1	
WPFGameController		Обработывает	-
	public void Window_KeyUp(object	отпускание	
	sender, KeyEventArgs e)	клавиши	
		клавиатуры	
	<pre>public override void Start()</pre>	Запускает WPF	-
	pastro override voia blaic()	контроллер игры	
		Останавливает	-
	public override void Stop()	WPF контроллер	
		игры	
	Worms WPF.Game.View		
	vv of his vv i i . Gaine. vie w	П	
	public static PhysicalObjectView	Получает	parPhysicalObject
PhysicalObjectViewFormer	GetShape (PhysicalObject	представление	- физический
	parPhysicalObject)	для физического	объект
		объекта	
	<pre>private void ViewInit()</pre>	Инициализирует	
MDEC 1.	biliare Aoid Aleminic()	представление	-
WPFGameView		Настроивает	-
	<pre>private void SetWindow()</pre>	окно	
		Присваивает	
	private void SetHandlers()	обработчики	_
		оораобтчики	



private void UnSetHandlers()	Убирает обработчики	-
<pre>private void ChangeGameTime(int parTime)</pre>	Изменяет время у представления таймера игры	parTime - время
<pre>private void ChangeTurnTime(int parTime)</pre>	Изменяет время у представления	parTime - время
<pre>private void DeletePhysicalObjectView(PhysicalObj ect parPhycicalObject)</pre>	Удаляет представление физического объекта из списка представлений	parPhycicalObject - физический объект
Worms WPF.Game.View.Objects View	v	
<pre>public override void Draw(Canvas parCanvas)</pre>	Рисует снаряд базуки	parCanvas - канвас
<pre>public override void Draw(Canvas parCanvas)</pre>	Рисует представление нанесенного по червяку урона	parCanvas - канвас
<pre>public override void Draw(Canvas parCanvas)</pre>	Рисует представление взрыва	parCanvas - канвас
<pre>public override void Draw(Canvas parCanvas)</pre>	Рисует представление гранаты	parCanvas - канвас
<pre>public abstract void Draw(Canvas parCanvas)</pre>	Рисует представление объекта	parCanvas - канвас
<pre>public override void Draw(Canvas parCanvas)</pre>	Рисует представление здоровья команды игрока	parCanvas - канвас
<pre>public override void Draw(Canvas parCanvas)</pre>	Рисует представление таймера	parCanvas - канвас
<pre>public override void Draw(Canvas parCanvas)</pre>	Рисует представление ветра	parCanvas - канвас
<pre>public override void Draw(Canvas parCanvas)</pre>	представление червяка	parCanvas - канвас
parCanvas)	Рисует тело червяка	parCanvas - канвас
parCanvas)	червяка	parCanvas - канвас
parCanvas)	червяка	parCanvas - канвас
<pre>private void DrawWeaponPower(Canvas parCanvas)</pre>	Рисует заряд оружия	parCanvas - канвас
<pre>private void DrawWeaponIcon(Canvas parCanvas)</pre>	оружия	parCanvas - канвас
<pre>private static void DrawWeaponCount(Canvas parCanvas)</pre>	Рисует количество оружия	parCanvas - канвас
	private void ChangeGameTime (int parTime)  private void ChangeTurnTime (int parTime)  private void Draw(PhysicalObjectView(PhysicalObject parPhycicalObject)  WormsWPF.Game.View.ObjectsView public override void Draw(Canvas parCanvas)  private void DrawWormShape(Canvas parCanvas)  private void DrawWormHealth(Canvas parCanvas)  private void DrawWormWeapon(Canvas parCanvas)  private void DrawWeaponPower(Canvas parCanvas)  private void DrawWeaponPower(Canvas parCanvas)  private void DrawWeaponIcon(Canvas parCanvas)  private static void	private void Unsethandlers()  private void ChangeGameTime (int parTime)  private void ChangeGameTime (int parTime)  private void ChangeTurnTime (int parTime)  private void ChangeTurnTime (int parTime)  private void ChangeTurnTime (int parTime)  private void DeleteFhysicalObjectView (PhysicalObject parPhycicalObject)  private void DeleteFhysicalObjectView (PhysicalObject parPhycialObject)  WormsWPE.Game.View.ObjectsView  public override void Draw (Canvas parCanvas)  private void DrawMormShape (Canvas par



	private void DrawAim(Canvas		parCanvas -
	parCanvas)	Рисует прицел	рагсапуаs - канвас
	private void DrawWormPointer(Canvas	Рисует указатель	parCanvas -
	parCanvas)	на червяка	канвас
	Worms WPF.Game EndedScreen.Contro	•	1
		Обрабатывает	
	<pre>private void KeyDownHandler(object sender, KeyEventArgs e)</pre>	нажатие клавиши	-
	Sender, Reynvenerings e,	клавиатуры	
		Запускает WPF	-
	public override void Start()	контроллер	
WPFGameEndedScreenController	pastio overtide vera seate()	экрана конца	
		игры	
		Останавливает	-
	public override void Stop()	WPF контроллер	
		экрана конца	
		игры	
	Worms WPF.Game EndedScreen. View		I
	private void SetupCanvas()	Настраивает	-
		канвас	
		Рисует WPF	
	public override void Draw()	представление	-
		экрана конца	
WPFGameEndedScreenView		игры Запускает WPF	
W FFGameEndedScreen vie w		представление	_
	public override void Start()	экрана конца	
		игры	
		Останавливает	_
		WPF	
	public override void Stop()	представление	
		экрана конца	
		игры	
	Worms WPF.Help.Controller	1	
		Обрабатывает	
	<pre>private void KeyDownHandler(object sender, KeyEventArgs e)</pre>	нажатие клавиши	-
	sender, ReyEventarys e)	клавиатуры	
		Запускает WPF	-
WPFHelpController	public override void Start()	контроллер	
		справки	
		Останавливает	-
	<pre>public override void Stop()</pre>	WPF контроллер	
		справки	
	Worms WPF.Help. View	Lyx	T
	private void SetupCanvas()	Настраивает	-
		канвас	
	public override void Dec.()	Рисует WPF	
	<pre>public override void Draw()</pre>	представление	-
		справки Запискает WPF	
WPFHelpView	<pre>public override void Start()</pre>	Запускает WPF представление	-
	pastic overtide void statt()	справки	
		Останавливает	_
	public override void Stop()	WPF	
		представление	
		справки	
	Worms WPF.Mai n Me nu. Contr oller	1	<u>l</u>



	private void KeyDownHandler(object	Обрабатывает	
	sender, KeyEventArgs e)	нажатие клавиши	-
	John Jan Stranger Str	клавиатуры	
		Запускает WPF	-
WPFMainMenuController	<pre>public override void Start()</pre>	контроллер	
		главного меню	
		Останавливает	-
	<pre>public override void Stop()</pre>	WPF контроллер	
	1111111	главного меню	
	Worms WPF,Mai n Me nu. Vie w	inabiloto memo	
	Worms WFF.Mannwienu. View	11	
	private void SetupCanvas()	Настраивает	-
		канвас	
		Рисует WPF	-
	public override void Draw()	представление	
		главного меню	
WPFMainMenuView		Запускает WPF	-
W FI Walliwellu view	public override void Start()	представление	
		главного меню	
		Останавливает	-
		WPF	
	<pre>public override void Stop()</pre>	представление	
		главного меню	
	Worms WPF.Records.Controller	13mbilo10 WORK	
	VVOI IIS VVI F.RECOI US. COILU OITEI	066	
	<pre>private void KeyDownHandler(object sender, KeyEventArgs e)</pre>	Обрабатывает	
		нажатие клавиши	-
		клавиатуры	
	<pre>public override void Start()</pre>	Запускает WPF	
WPFRecordsController		контроллер	-
		рекордов	
		Останавливает	
	public override void Stop()	WPF контроллер	-
		рекордов	
	Worms WPF.Records.View		
	animate maid Catum Camana ()	Настраивает	
	private void SetupCanvas()	канвас	-
		Рисует WPF	-
	<pre>public override void Draw()</pre>	представление	
		рекордов	
		Запускает WPF	-
WPFRecordsView	public override void Start()	представление	
	1.7222 2.7222 2.722	рекордов	
		+	
		Останавливает	-
	public override void Stop()	WPF	
		представление	
		рекордов	
	Worms Cons ole		
	public override GameController	Создает	<u>-</u>
	CreateGameController()	контроллер игры	
		Создает	-
	public override	контроллер	
	GameEndedScreenController	экрана конца	
ConsoleControllersFactory	CreateGameEndedScreenController()	игры	
		Создает	
	<pre>public override HelpController CreateHelpController()</pre>	контроллер	
		справки	
	public override MainMenuController	+ + + + + + +	
	Pastic overtide mathmendeconcrotter	Создает	-



	CreateMenuController()	-	
	createmenucontroller()	контроллер	
		главного меню	
	public override RecordsController	Создает	-
	CreateRecordsController()	контроллер	
		рекордов	
			parX - координата
			Х,
			parY - координата
	public static void DrawRectangle(int	D.	Υ,
ConsoleDrawer	parX, int parY, int parWidth, int	Рисует	parWidth - ширина
	parHeight, ConsoleColor color)	прямоугольник	прямоугольника, parHeight -
			высота
			прямоугольника,
			color - цвет
		Конфигурирует	
ConsoleWorms Application	<pre>private void ConfigureWindow()</pre>	окно	-
			parX - координата
		V	X,
	public static void SetPixel(int	Установливает	рагҮ - координата
	parX, int parY, short attribute)	пиксель	Υ,
			attribute - цвет
		Инициализирует	
		Помощник по	
	public static void Init(short	работе с	parWidth -
	parWidth, short parHeight)	быстрым	ширина, parHeight
		выводом в	- высота
		консоль	
		Рисует все	
	<pre>public static void Draw()</pre>	символы из	_
FastConsoleWorker	public beacle void blaw()		
		буфера в консоли	parX - координата
			Х,
			parY - координата
	public static void DrawRectangle(int	Рисует	Υ,
	parX, int parY, int parWidth, int	прямоуголник в	parWidth -
	parHeight, short parColor)	буфере	ширина,
			parHeight -
			выста, parColor - цвет
			parColor - цвет parXSize - размер
	public static void SetFontSize(short	Устанавливает	по Х,
	parXSize, short parYSize)	размер шрифта	parYSize - размер
		1	по У
	Worms Console. Game. Controller	•	•
		Запускат	
		считывание	
	<pre>public void ReadKeysStart()</pre>	нажатых клавиш	_
		в консоли	
		Останавливает	
	public word PoodVoveCtor()		
	<pre>public void ReadKeysStop()</pre>	считывание	_
ConsoleGameController		клавиш	
		Запускает	_
	<pre>public override void Start()</pre>	консольный	
		контроллер игры	
		Останвливает	-
	<pre>public override void Stop()</pre>	консольный	
		контроллер игры	
	Worms Console. Game. View	·	
ConsoleGameView	<pre>public void ViewInit()</pre>	Инициализирует	-
	<u>.1</u>	, -FJ	1



	I	1	
		консольное	
		представление	
		игры	
		Удаляет	
		представление	
	private void	физического	parPhycicalObject
	DeletePhysicalObjectView(PhysicalObj	объекта из	- физический
	ect parPhycicalObject)		объект
		списка	
		представлений	
		Добавление	
	private void	представления	parPhycicalObject
	AddNewView(PhysicalObject	физического	- физический
	parPhysicalObject)	объекта в список	объект
	F,		
		представлений	
	<pre>private void DrawMap()</pre>	Рисует карту	=
		Запускает	
		отрисовку	
	<pre>private void StartDrawing()</pre>	консольного	-
	-	представления	
		•	
		игры	
		Рисует	
	<pre>public override void Draw()</pre>	консольное	_
	pastio overtide vota braw()	представление	_
		игры	
		Запускает	_
		консольное	
	public override void Start()		
		представление	
		игры	
		Останавливает	-
	nublic everyide void Cten()	консольное	
	<pre>public override void Stop()</pre>	представление	
		игры	
		Получает	
	public static PhysicalObjectView	1	parPhysicalObject
PhysicalObjectViewFormer	GetShape(PhysicalObject	представление	- физический
	parPhysicalObject)	для физического	объект
		объекта	
	Worms Console. Game EndedScreen. Contr	oller	<del></del>
		Запускает	
		считывание	
	<pre>public void ReadKeysStart()</pre>	нажатых клавиш	_
	partie tota houding social ( )		
		в консоли конца	
		игры	
		Останавливает	
		считывание	
	<pre>public void ReadKeysStop()</pre>	клавиш в	-
		консоли конца	
ConsoleGameEndedScreenControl		игры	
ler			
		Запускает	-
		консольный	
	public override void Start()	контроллер	
		экрана конца	
		игры	
		Останавливает	-
		консольный	
İ	public override void Stop()	ROHEOMBIBIN	
	r	TO LITTO O TITO	
		контроллер экрана конца	



		игры
	Worms Console.Game EndedScreen.Vi	=
	World Console. Gaille Laide do ei een. Vi	Настраивает
	<pre>public void SetupWindow()</pre>	окно
	animate and I Traitical Durant	Начальная
	private void InitialDraw()	отрисовка при -
		запуске
	public override void Draw()	Рисует введенное
		имя победителя
		Запускает -
ConsoleGameEndedScreenView		консольное
	public override void Start()	представление
		экрана конца
		игры
		Останавливает -
		консольное
	public override void Stop()	представление
		экрана конца
		игры
	Worms Console. Help. Controller	
		Запускает
		считывание
	<pre>public void ReadKeysStart()</pre>	нажатых клавиш -
		в консоли
		справки
		Останавливает
	<pre>public void ReadKeysStop()</pre>	считывание
		клавиш в
ConsoleHelpController		консоли справки
		Запускает -
	public override void Start()	консольный
		контроллер
		справки
		Останавливает -
	<pre>public override void Stop()</pre>	консольный
	pastis evertide veta seep()	контроллер
		справки
	Worms Console.Help.View	•
		Рисует
	mublic arranging resid D	консольное
	<pre>public override void Draw()</pre>	представление
		справки
		Запускает -
		консольное
ConsoleHelpView	public override void Start()	представление
		справки
		Останавливает -
	public override void Stop()	КОНСОЛЬНОЕ
		представление
	Wasse C. L. M. L. C	справки
	Worms Console.MainMenu.Controlle	
		Dominario em
		Запускает
		считывание
ConsoleMainMenuController	<pre>public void ReadKeysStart()</pre>	
ConsoleMainMenuController	<pre>public void ReadKeysStart()</pre>	считывание



		Останавливает	
		считывание	
	<pre>public void ReadKeysStop()</pre>	клавиш в	-
		консоли главного	
		меню	
		Запускает	-
		консольный	
	<pre>public override void Start()</pre>	контроллер	
		главного меню	
		Останавливает	-
	<pre>public override void Stop()</pre>	консольный	
		контроллер	
		главного меню	
	<pre>public void SetupWindow()</pre>	Настраивает	
	public void Setupwindow()	окно	-
		Рисует	-
		консольное	
	public override void Draw()	представление	
		главного меню	
		Запускает	_
Worms Console. Main Menu. View		-	_
	<pre>public override void Start()</pre>	консольное	
		представление	
		главного меню	
		Останавливает	-
	<pre>public override void Stop()</pre>	консольное	
	pastic overlide void scop()	представление	
		главного меню	
	Worms Console.Records.Controller		
		Запускает	
		считывание	
	<pre>public void ReadKeysStart()</pre>	нажатых клавиш	_
		в консоли	
		рекордов	
		Останавливает	-
		считывание	
	<pre>public void ReadKeysStop()</pre>	клавиш в	
ConsoleRecordsController		консоли	
Consoleracondictional		рекордов	
		Запускает	-
	nublic orrorrida maid Chart ()	консольный	
	<pre>public override void Start()</pre>	контроллер	
		рекордов	
		Останавливает	_
		консольный	
	<pre>public override void Stop()</pre>	контроллер	
	Wassach P. 1 W	рекордов	
	Worms Console.Records.View	T p	T
		Рисует	-
	public override void Draw()	консольное	
	-	представление	
		рекордов	
ConsoleRecords View		Запускает	-
		консольное	
	<pre>public override void Start()</pre>	представление	
		рекордов	
	İ	r r A	
	<pre>public override void Stop()</pre>	Останавливает	_



	консольное	
	представление	
	рекордов	

#### 2.2 Разработка программы

Описание последовательности разработки:

- 1. Анализ предметной области.
- 2. Составление технического задания.
- 3. Проектирование игры.
- 4. Разработка игры.
- 5. Тестирование программы.
- 6. Написание документации.

#### 2.2.1 Описание классов, перечислений и интерфейсов проекта

Пространство имен	Название класса / интерфейса / перечисления	Назначение
WormsGame	AbstractControllersFactory	Абстрактный класс фабрики контроллеров
WormsGame	WormsApplication	Класс приложения
WormsGame.BaseMVC	Controller	Абстрактный контроллер MVC
WormsGame.BaseMVC	Model	Абстрактная модель MVC
WormsGame.BaseMVC	View	Абстрактное представлени MVC
WormsGame.Controller	GameController	Абстрактный контроллер игры
WormsGame.Base	GameInitializer	Класс инициализации игры
WormsGame.Base	GameModel	Класс модели игры
WormsGame.Base	GameTimers	Класс игрового таймера
WormsGame.Base	Inventory	Класс инвентаря
WormsGame.Base	MapReader	Класс считывателя карты
WormsGame.Base	PhysicalObject PhysicalObject	Абстрактный класс физического объекта
WormsGame.Base	Player	Класс игрока
WormsGame.Base	PlayersColor	Класс перечислений цвета игроков
WormsGame.Base	State	Класс перечислений состояний червяка
WormsGame.Base	Worm	Класс червяка
WormsGame.Base.Weapons	Bazooka	Класс базуки
WormsGame.Base.Weapons	BazookaProjectile	Класс снаряда базуки
WormsGame.Base.Weapons	ExplosionMaker	Класс создателя



		взрывов
WormsGame.Base.Weapons	Grenade	Класс гранаты
WormsGame.Base.Weapons	GrenadeProjectile	Класс снаряда игры
WormsGame.Base.Weapons	Projectile	Класс снаряда
WormsGame.Base.Weapons	Weapon	Класс оружия
WormsGame.GameEndedScreen	GameEndedScreenController	Класс контроллера
.Controller		экрана конца игры
WormsGame.GameEndedScreen	GameEndedScreenModel	Класс модели экрана
.Model		конца игры
WormsGame.GameEndedScreen	GameEndedScreenView	Абстрактный класс
.View		представления
		экрана конца игры
WormsGame.Help.Controller	HelpController	Абстрактный класс
wormsdame.nerp.concrorrer	neipconcionei	контроллера справки
WormsGame.Help.Model	HelpModel	Класс модели
wormsdame.nerp.moder	neipmodei	
WormsGame.Help.View	HelpView	справки Абстрактный класс
wormsdame.nerp.view	петрутем	
		представления
W. C. W. M. M.	Maria Maria Carata and Illiano	справки
WormsGame.MainMenu	MainMenuController	Класс контроллера
.Controller	10.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.	главного меню
WormsGame.MainMenu.Model	MainMenuModel	Класс модели
		главного меню
WormsGame.MainMenu.Model	MenuItemTextFormer	Класс формирователя
		текста для пунктов
		меню
WormsGame.MainMenu.View	MainMenuView	Абстрактный класс
		представления
		главного меню
WormsGame.MainMenu	Menu	Класс главного меню
WormsGame.MainMenu	MenuItem	Класс перечислений
		пунктов главного
		меню
WormsGame.Records	RecordsController	Абстрактный класс
.Controller		контроллера
		рекордов
WormsGame.Records.Model	GameRecord	Класс игрового
		рекорда
WormsGame.Records.Model	RecordsModel	Класс модели
		рекордов
WormsGame.Records.View	RecordsView	Абстрактный класс
		представления
		рекордов
WormsGame.Records	RecordsRepository	Класс репозитория
		рекордов
WormsWPF	WPFControllersFactory	Класс фабрики WPF
MOTITIONALE	WITCOMCLOTICESFACCOLY	контроллеров
WormsWPF	WPFWormsApplication	Класс WPF
MOTIUSMIT	MII MOTHBY PATTCACTOH	
WormsWPF.WPFController	WPFGameController	приложения
wormswrf.wrfController	wrrgamecontroller	Класс WPF
	1	контроллера игры



	Γ	
WPFWorms.WPFView	PhysicalObjectViewFormer	Класс формирователя
		представления для
		физичеких объектов
WormsWPF	WPFGameView	Класс WPF
		представления игры
WPFWorms.WPFView	BazookaProjectileView	Класс представления
.ObjectsView		снаряда базуки
WPFWorms.WPFView	DamageView	Класс представления
.ObjectsView		нанесенного по
		червяку урона
WPFWorms.WPFView	ExplosionView	Класс представления
.ObjectsView		взрыва
WPFWorms.WPFView	GrenadeProjectileView	Класс представления
.ObjectsView		гранаты
WPFWorms.WPFView	ObjectView	Абстрактный класс
.ObjectsView		представления
		объекта
WPFWorms.WPFView	PhysicalObjectView	Абстрактный класс
.ObjectsView		представления
		физического объекта
WPFWorms.WPFView	PlayerHealthView	Класс представления
.ObjectsView		здоровья команды
		игрока
WPFWorms.WPFView	TimeView	Класс представления
.ObjectsView		таймера
WPFWorms.WPFView	WindView	Класс представления
.ObjectsView		ветра
WPFWorms.WPFView	WormView	Класс представления
.ObjectsView		червяка
WormsWPF.GameEndedScreen	WPFGameEndedScreen	Класс WPF
.Controller	Controller	контроллера экрана
		конца игры
WormsWPF.GameEndedScreen	WPFGameEndedScreenView	Класс WPF
.Model		представления
		экрана конца игры
WormsWPF.Help.Controller	WPFHelpController	Класс WPF
		контроллера справки
WormsWPF.Help.View	WPFHelpView	Класс WPF
		представления
		справки
WormsWPF.MainMenu	WPFMainMenuController	Класс WPF
.Controller		контроллера
		главного меню
WormsWPF.MainMenu.View	WPFMainMenuView	Класс WPF
		представления
		главного меню
WormsWPF.Records.Controller	WPFRecordsController	Класс WPF
		контроллера
		рекордов
WormsWPF.Records.View	WPFRecordsView	Класс WPF
		представления
	<u> </u>	



		рекордов
WormsConsole	ConsoleControllersFactory	Класс фабрики
	1	консольных
		контроллеров
WormsConsole	ConsoleDrawer	Класс консольного
		рисовальщика
WormsConsole	ConsoleUtils	Класс средств для
		осуществления
		работы помощника
		для работы с
		быстрым выводом
		консоли
WormsConsole	ConsoleWormsApplication	Класс консольного
		приложения
WormsConsole	FastConsoleWorker	Класс помощника по
		работе с быстрым
		выводом в консоль
WormsConsole.ControllerConsole	ConsoleGameController	Класс консольного
		контроллера игры
WormsConsole.ViewConsole	ConsoleGameView	Класс консольного
		представления игры
WormsConsole.ViewConsole	PhysicalObjectViewFormer	Класс формирователя
		представлений
		физических объектов
WormsConsole.ViewConsole	BazookaProjectileView	Класс представления
.ViewObjects		снаряда базуки
WormsConsole.ViewConsole	GameTimeView	Класс представления
.ViewObjects		игрового таймера
WormsConsole.ViewConsole	GrenadeProjectileView	Класс представления
.ViewObjects		гранаты
WormsConsole.ViewConsole	ObjectView	Абстрактный класс
.ViewObjects		представления
		объекта
WormsConsole.ViewConsole	PhysicalObjectView	Абстрактный класс
.ViewObjects		представления
	m m' 11'	физического объекта
WormsConsole.ViewConsole	TurnTimeView	Класс представления
.ViewObjects	57	таймера хода
WormsConsole.ViewConsole	WormView	Класс представления
.ViewObjects	G l. G T l lG	червяка
WormsConsole.GameEndedScreen	ConsoleGameEndedScreen	Класс консольного
.Controller	Controller	контроллера экрана
Warma Canaala Cama Endad Canaan		конца игры
WormsConsole.GameEndedScreen .View	ConsoleGameEndedScreenView	Класс консольного
·view		представления
WormsConsolo Holm Controller	ConsoleHelpController	экрана конца игры
WormsConsole.Help.Controller	consoremerbeonergree	Класс консольного
WormsConsolo Holm Wicz	ConsolaHalnWay	контроллера справки
WormsConsole.Help.View	ConsoleHelpView	Класс представления
WormsConsole.MainMenu	ConsoleMainMenuController	справки Класс консольного
WOTHISCOHSOTE . MIGTHIMEHU	COMPOTEMATHMENUCONCLOTTEL	MITACE KUHCUIIBHUI'O



.Controller		контроллера
		главного меню
WormsConsole.MainMenu.View	ConsoleMainMenuView	Класс консольного
		представления
		главного меню
WormsConsole.Records	ConsoleRecordsController	Класс консольного
.Controller		контроллера
		рекордов
WormsConsole.Records.View	ConsoleRecordsView	Класс консольного
		представления
		рекордов

# 2.3 Описание шаблонов проектирования, которые использовались при написании программы

#### 2.3.1 Абстрактная фабрика (Abstract factory)

Абстрактная фабрика — это порождающий паттерн проектирования, который позволяет создавать семейства связанных объектов, не привязываясь к конкретным классам создаваемых объектов. Шаблон реализуется созданием абстрактного класса, который представляет собой интерфейс для создания компонентов системы. Затем пишутся классы, реализующие этот интерфейс.

Фабрика контроллеров реализуется в классе AbstractControllersFactory, фабрика WPF контроллеров — в классе WPFControllersFactory, фабрика консольных контроллеров — в классе ConsoleControllersFactory. Шаблон используется в классе WormsApplication.

#### 2.3.2 Одиночка (Singleton)

Одиночка — порождающий шаблон проектирования, гарантирующий, что в однопоточном приложении будет единственный экземпляр некоторого класса, и предоставляющий глобальную точку доступа к этому экземпляру.

Шаблон реализуется в классе RecordsRepository, используется в классе RecordsModel.

## 2.4 Описание методов рефакторинга, которые использовались при оптимизации исходного кода программы

#### 2.4.1 Вынесение констант

Программный код до вынесения константы:

```
/// <summary>
/// Убить червяка со взрывом
/// </summary>
public void KillWithExplosion()
{
   Player.Worms.Remove(this);
   GameModel.RemoveObject(this);
   ExplosionMaker.MakeExplosion(GetCenterX(), GetCenterY(), 50, 10, 25);
}
```

#### Программный код после вынесения константы:

```
/// <summary>
/// Радиус взрыва червяка
/// </summary>
private const int WORM_EXPLOSION_RADIUS = 50;
/// <summary>
/// Передаваемая скорость взрывом червяка
/// </summary>
private const int WORM EXPLOSION TRANSMITTED SPEED = 10;
/// <summary>
/// Урон взрыва червяка
```



```
/// </summary>
private const int WORM_EXPLOSION_DAMAGE = 25;

/// <summary>
/// YOUTH wepbska co bspbbom
/// </summary>
public void KillWithExplosion()
{
    Player.Worms.Remove(this);
    GameModel.RemoveObject(this);
    ExplosionMaker.MakeExplosion(GetCenterX(), GetCenterY(), WORM_EXPLOSION_RADIUS,
WORM_EXPLOSION_TRANSMITTED_SPEED, WORM_EXPLOSION_DAMAGE);
}
```

#### 2.4.2 Переименование метода

Программный код до переименования метода:

```
/// <summary>
/// Удалить игроков без червяков
/// </summary>
private static void RemovePlayers()
{
  int i = 0;
  while (i < Players.Count)
  {
   if (Players[i].Worms.Count == 0)
      {
       Players.RemoveAt(i);
       }
       else
       {
       i++;
       }
   }
  }
}
```

Программный код после переименования метода:

```
/// <summary>
/// Удалить игроков без червяков
/// </summary>
private static void RemovePlayersWithoutWorms()
{
  int i = 0;
  while (i < Players.Count)
  {
   if (Players[i].Worms.Count == 0)
   {
     Players.RemoveAt(i);
   }
   else
   {
     i++;
   }
}
```

#### 2.4.3 Вынесение кода в отдельный метод

Программный код до вынесения кода в отдельный метод:

```
/// <summary>
/// Приготовить следующий ход
/// </summary>
private static void PrepareNextTurn()
{
  int i = 0;
  while (i < Players.Count)
  {
   if (Players[i].Worms.Count == 0)
    {
      Players.RemoveAt(i);
   }
   else
   {
      i++;
   }
  }
  if (IsGameEnd())
  {
```



```
return;
}
Thread.Sleep(2000);
WindStrength = _random.Next(WIND_STRENGTH_MIN, WIND_STRENGTH_MAX + 1);
_currentPlayerIndex++;
if (_currentPlayerIndex >= Players.Count)
{
    _currentPlayerIndex = 0;
}
Players[_currentPlayerIndex].NextWorm();
Players[_currentPlayerIndex].GetCurrentWorm().IsUsedWeapon = false;
_isSelectedWormControlling = true;
WormControlChangedEvent?.Invoke(GetCurrentWorm());
GameTimers.TurnTimer.Enabled = true;
GameTimers.ResetTurnTimer();
}
```

#### Программный код после вынесения кода в отдельный метод:

```
/// <summary>
    /// Приготовить следующий ход
    /// </summary>
    private static void PrepareNextTurn()
      RemovePlayersWithoutWorms();
      if (IsGameEnd())
        return;
      Thread.Sleep (2000);
      SetRandomWindStrength();
      SetNextPlayerIndex();
      PrepareWormForTurn();
      StartNextTurn();
    /// <summary>
    /// Удалить игроков без червяков
/// </summary>
   private static void RemovePlayersWithoutWorms()
      int i = 0:
      while (i < Players.Count)
        if (Players[i].Worms.Count == 0)
         Players.RemoveAt(i);
        else
          i++;
    /// <summary>
    /// Установить индекс следующего червяка
    /// </summary>
    private static void SetNextPlayerIndex()
      currentPlayerIndex++;
      if ( currentPlayerIndex >= Players.Count)
         currentPlayerIndex = 0;
    /// <summary>
    /// Приготовить червяка для хода
    /// </summary>
   private static void PrepareWormForTurn()
      Players[ currentPlayerIndex].NextWorm();
      Players[_currentPlayerIndex].GetCurrentWorm().IsUsedWeapon = false;
       isSelectedWormControlling = true;
```



```
}

/// <summary>
/// Запустить следующий ход
/// </summary>
private static void StartNextTurn()
{
    WormControlChangedEvent?.Invoke(GetCurrentWorm());
    GameTimers.TurnTimer.Enabled = true;
    GameTimers.ResetTurnTimer();
}

/// <summary>
/// Установить случайную силу ветра
/// </summary>
public static void SetRandomWindStrength()
{
    WindStrength = _random.Next(WIND_STRENGTH_MIN, WIND_STRENGTH_MAX + 1);
}
```

# 2.5 Разработка тестов

#### 2.5.1 Test Cases

№	Название	Предусловия	Шаги	Ожидаемый результат
1	Тест на успешное движение червяка влево	Карта инициализ ирова на, червяк находится на земле	Вызывается метод передвижения червяка влево	Координата X червяка уменьшается на один пиксель
2	Тест на успешное движение червяка вправо	Карта инициализ ирова на, червяк находится на земле	Вызывается метод передвижения червяка вправо	Координата X червяка увеличивается на один пиксель
3	Тест на успешное движение червяка вправо через низкое препятствие	Карта инициализирова на с низким препятствием, червяк находится на земле	Вызывается метод передвижения червяка вправо через низкое препятствие	Координата X червяка увеличивается на один пиксель, Координата Y увеличивается на высоту препятствия
4	Тест на успешное движение червяка влево через низкое препятствие	Карта инициализирова на с низким препятствием, червяк находится на земле	Вызывается метод передвижения червяка влево через низкое препятствие	Координата X червяка уменьшается на один пиксель, Координата Y увеличивается на высоту препятствия
5	Тест на движение червяка вправо через высокое препятствие	Карта инициализирова на с высоким препятствием, червяк находится на земле	Вызывается метод передвижения червяка вправо через высокое препятствие	Координаты Х,Ү червяка не меняются
6	Тест на движение червяка влево через высокое препятствие	Карта инициализирова на с высоким препятствием, червяк находится на земле	Вызывается метод передвижения червяка влево через высокое препятствие	Координата Х,Ү червяка не меняются
7	Тест на состояние	Карта	Вызывается метод	Метод проверки



	Hennaka Ha partia	инини шилимово ис	HINDRANGI OTOUT THE	DODDOUGAT topo
	червяка на земле	инициализирована,	проверки стоит ли	возвращает true
		червяк находится	червяк на земле	
		на земле		
	Тест на состояние	Карта	Вызывается метод	M
8	червяка, находящегося не	инициализ ирова на,	проверки стоит ли	Метод проверки
	на земле	червяк находится	червяк на земле	возвращает false
		не на земле	1	
		Карта		Скорость по Х и по Ү
9	Тест на прыжок червяка	инициализ ирова на,	Вызывается метод	червяка равны скоростям,
	вперед, когда он	червяк находится	прыжка червяка	которую придает прыжок
	повернут влево	на земле, червяк	вперед	вперед, когда червяк
		повернут влево		повернут влево
		Карта		Скорость по Х и по Ү
	Тест на прыжок червяка	инициализ ирова на,	Вызывается метод	червяка равны скоростям,
10	вперед, когда он	червяк находится	прыжка червяка	которую придает прыжок
	повернут вправо	на земле, червяк	вперед	вперед, когда червяк
		повернут вправо		повернут вправо
	T	Карта	D	
4.4	Тест на прыжок червяка	инициализ ирова на,	Вызывается метод	Скорость по Х и по Ү не
11	вперед, когда он не на	червяк находится	прыжка червяка	изменились – равны 0
	земле	не на земле	вперед	•
		Карта		Скорость по Х и по Ү
	Тест на прыжок червяка	инициализ ирова на,	_	червяка равны скоростям,
12	назад, когда он повернут	червяк находится	Вызывается метод	которую придает прыжок
	влево	на земле, червяк	прыжка червяка назад	назад, когда червяк
	2.1020	повернут влево		повернут влево
		Карта		Скорость по Х и по Ү
	Тест на прыжок червяка	инициализ ирова на,		червяка равны скоростям,
13	назад, когда он повернут	червяк находится	Вызывается метод	которую придает прыжок
	вправо	на земле, червяк	прыжка червяка назад	назад, когда червяк
	Bilpube	повернут вправо		повернут вправо
		Карта		повернут вприве
	Тест на прыжок червяка	инициализирована,	Вызывается метод	Скорость по Х и по У не
14	назад, когда он находится	червяк находится	прыжка червяка назад	изменились – равны 0
	не на земле	не на земле	прымка черыка назад	изменились — равны о
	Тест на состояние	Карта	Вызывается метод	Matail Harran
15	червяка, столкнувшегося	инициализирована,	проверки столкнулся	Метод проверки
	с землей	червяк находится	ли червяк с землей	возвращает true
		на земле		
	Тест на состояние	Карта	Вызывается метод	Morrowania
16	червяка, не	инициализ ирова на,	проверки столкнулся	Метод проверки
	столкнувшегося с землей	червяк находится	ли червяк с землей	возвращает false
	•	не на земле	*	
	Тест на увеличение	Карта	Вызывается метод	
17	скорости червяка по Y,	инициализ ирова на,	обновления состояния	Скорость червяка по Ү
	когда он в воздухе	червяк находится	червяка	увеличилась на 1
		не на земле	10,22	
	Тест на отсутствие		Вызывается метод	
18	увеличения скорости	Червяк находится	обновления состояния	Скорость червяка по Ү не
10	червяка по Y, когда он на	на земле	червяка	изменилась
	земле		Торыка	



19	Тест на остановку червяка на земле, когда его скорость превышает расстояние до земли	Карта инициализирована, червяк находится не на земле	Вызывается метод обновления червяка	Червяк находится на земле
20	Тест на перемещение червяка посредством скорости	Карта инициализ ирова на	Вызывается метод обновления состояния червяка	Червяк переместился на значения скоростей по X и по Y

#### 2.5.2 Модульные тесты

Модульное тестирование или юнит-тестирование — это процесс проверки программного кода, при котором проверяют работоспособность отдельных компонентов или модулей разработанной программы. Цель модульного тестирования состоит в выявлении локализованных в модуле ошибок в реализации алгоритмов, а также в определении степени готовности системы к переходу на следующий уровень разработки и тестирования. Модульное тестирование проводится по принципу «белого ящика», то есть основывается на знании внутренней структуры программы, и часто включает те или иные методы анализа покрытия кода. Ниже представлен пример тестирования успешного движения червяка вправо. Инициализируется карта с землей, задаются стартовые значения координат X и Y червяка, а также ожидаемое значение координаты X. Создается червяк, для которого вызывается метод движения влево. С помощью метода Assert.AreEqual проверяется совпадение ожидаемого и реального значения координат X и Y.

```
<summarv>
/// Координата У земли
/// </summary>
public const int GROUND Y = 50;
/// <summary>
/// Высота червяка
/// </summary>
public const int WORM HEIGHT = 18;
/// <summarv>
/// Тестирование успешного движения червяка влево
/// </summary>
[TestMethod]
public void TestSuccessWormMoveLeft()
  InitializeDefaultTestMap();
  int startX = 50;
  int expectedX = 49;
  int startY = GROUND Y - WORM HEIGHT;
  Worm worm = new Worm(null, startX, startY, WORM WIDTH, WORM HEIGHT, 100);
  worm.MoveLeft();
  Assert.AreEqual(expectedX, worm.X);
  Assert.AreEqual(startY, worm.Y);
```



# 3 Результат работы программы

# 3.1 Графическая версия

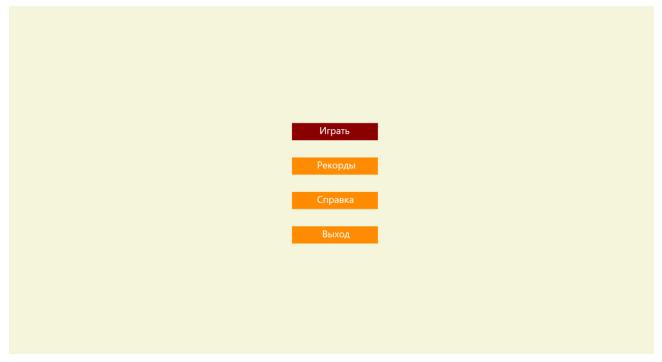


Рисунок 18 – Главное меню графической версии приложения

Рекорды. Нажмите Esc для возврата в меню
1. Unknown - 500 сек.

Рисунок 19 – Таблица рекордов графической версии приложения



#### Справка по игре Worms. Нажмите Esc для возврата в меню

Worms представляет собой пошаговую игру. В ней каждый игрок владеет армией червяков и инвентарем с оружием и вспомогательными предметами, которыми могут пользоваться червяки. Игроки по очереди выполняют ограниченный по времени ход, во время которого они должны попытаться нанести наибольший урон червякам противника, либо обезопасить своих червяков. Целью игры является уничтожение всех червяков противника.

Игра продолжается до тех пор, пока на поле имеются червяки по крайней мере 2-х разных коалиций, либо до тех пор, пока игровой таймер не дойдет до 0.

Управление: A, D - ходьба червяком влево, вправо. W, S - поднять, опустить прицел оружия. R, F - прыжок назад, вперед. Клавиша 1 экипировать базуку, Клавиша 2 - экипировать гранату. Удерживание пробела - заряд экипированного оружия, отпускание пробела - выстрел экипированным оружием оружием. Escape - выйти в меню.

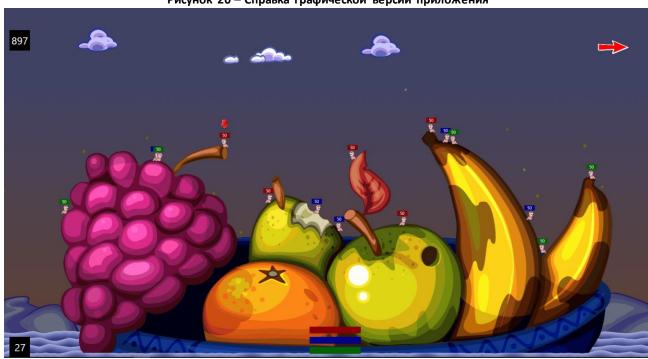


Рисунок 20 - Справка графической версии приложения

Рисунок 21 – Игра графической версии приложения



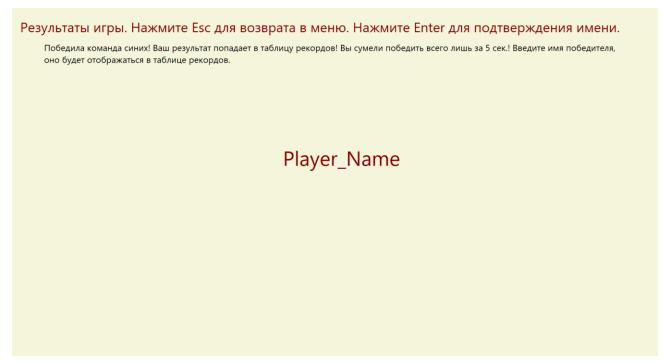


Рисунок 22 - Окно с результатом игры графической версии приложения

#### 3.2 Консольная версия

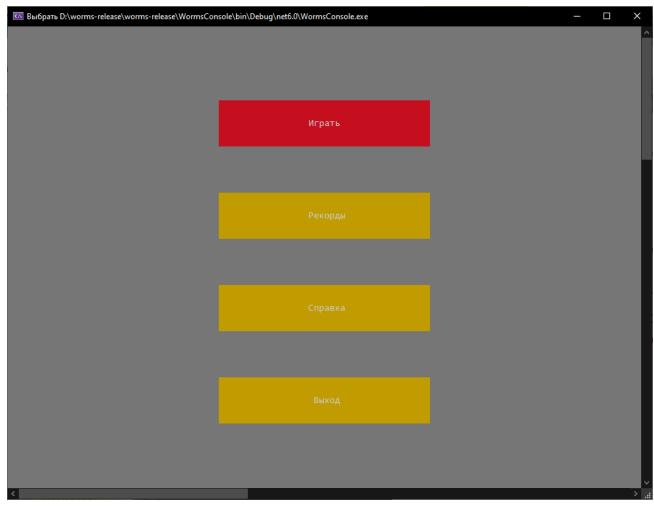


Рисунок 23 - Главное меню консольной версии приложения



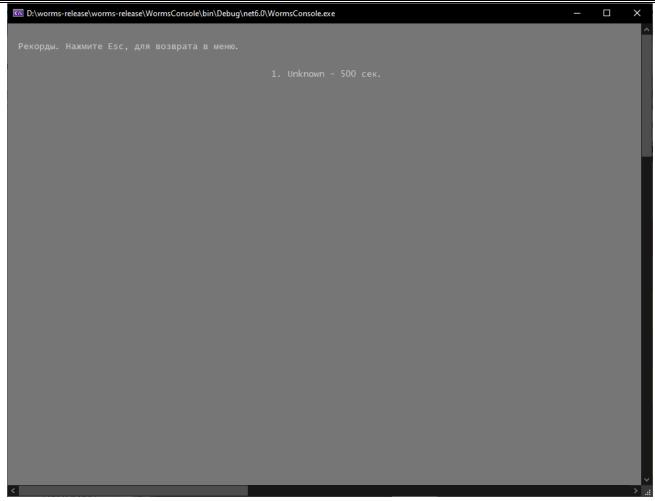


Рисунок 24 – Таблица рекордов консольной версии приложения



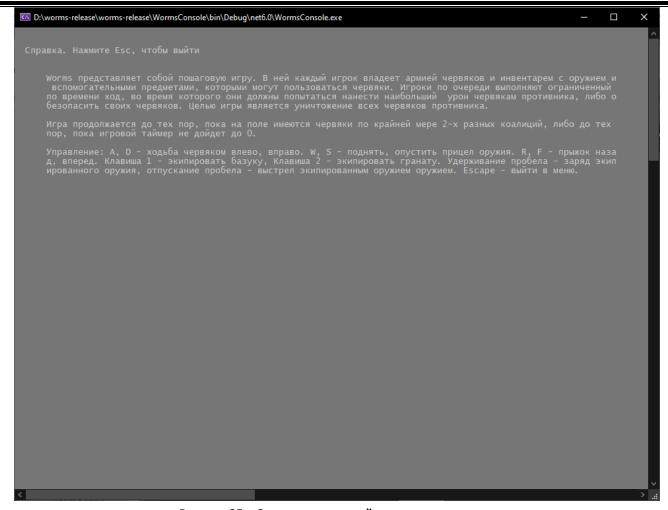


Рисунок 25 – Справка консольной версии приложения

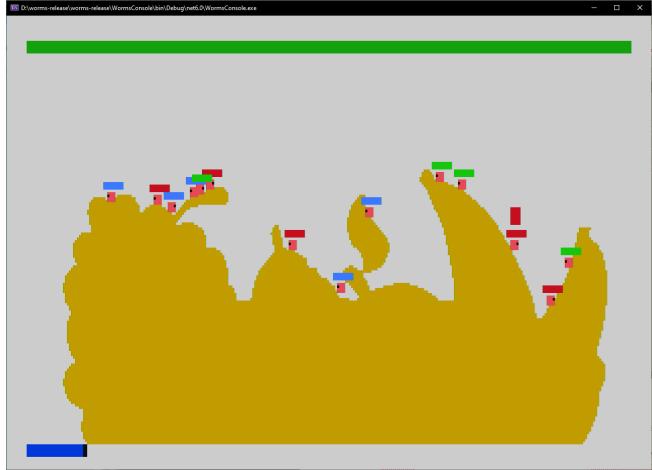


Рисунок 26 - Игра консольной версии приложения



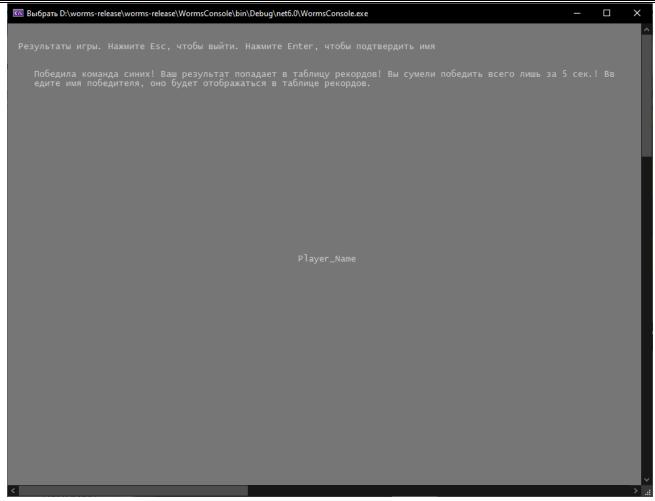


Рисунок 27 – Окно с результатом игры консольной версии приложения

#### Заключение

В результате выполнения курсовой работы было разработано игровое приложение Worms с 2 реализациями: консольная (используя консоль) и оконая (используя WPF). Были использованы шаблоны проектирования MVC (Модель-Вид-Контроллер) и два дополнительных шаблона проектирования: Abstract factory (Абстрактная фабрика) и Singleton (Одиночка). В приложении использовалась многопоточность и синхронизация между потоками. Для класса Worm разработаны полноценные модульные тесты, обеспечивающие проверку корректности его работы.

## Приложения