

## Содержание

Введение.....	3
1. Краткие сведения о решаемых задачах.....	4
1.1. Разновидность компьютерных игр.....	5
1.2. Список игровых жанров.....	5
2. Техническое задание.....	6
2.1. Назначение разработки.....	7
2.2. Требование к составу выполняемых функций.....	7
2.3. Требования к техническому обеспечению.....	7
3. Выбор средств разработки.....	8
4. Создание программы .....	10
4.1. Процесс создания программы.....	10
4.2. Описание интерфейса.....	11
4.3. Краткие сведения о проделанной работе.....	13
Заключение.....	13
Список литературы.....	14
Приложение А.....	15
Приложение Б.....	16

## Введение

Наука об искусственном интеллекте ведёт своё начало с середины XX века. Начиная с того времени, во многих исследовательских лабораториях учёные ведут работу над созданием компьютеров, обладающих способностью думать на таком же уровне, что и человек. В то время уже существовали предпосылки к возникновению искусственного разума. Так, психологами была создана модель мозга человека и изучены процессы мышления. Учёные-математики создали теорию алгоритмов, ставшую фундаментом математической теории вычислений, были упорядочены и структурированы знания о мире, решены вопросы оптимальных расчетов и созданы самые первые компьютеры.

Новые машины были способны производить вычисления гораздо быстрее человека, поэтому учёные задумались о возможности создания вычислительных машин, достигших уровня развития людей. В 1950 английским учёным Аланом Тьюрингом была опубликована статья «Способна ли машина мыслить?». В этой статье он предлагает определять степень разумности машины с помощью разработанного им теста, впоследствии получившего название «тест Тьюринга».

Но учёным и до сих пор не удаётся прийти к единому мнению относительно того, что же есть интеллект. Одни считают признаком интеллекта способность решать задачи высокой сложности; для других интеллект - это, прежде всего способность к обучению, обобщению и анализу информации; третьи считают, что это возможность эффективно взаимодействовать с окружающим миром, способность к общению, восприятию и осознанию воспринятой информации.

## **Краткие сведения о решаемых задачах**

На сегодняшний день разработка ИИ является трудоемким и дорогим процессом. Не смотря на это, множество IT компаний по всему миру готовы внедрить человекоподобное мышление в нашу повседневную жизнь.

Как и другие прикладные науки, наука об искусственном интеллекте представлена теоретической и экспериментальной частями. Практически, «Искусственный интеллект» занимает промежуточное положение между информатикой и вычислительной техникой и такими дисциплинами как когнитивная и поведенческая психология и нейрофизиология. Что касается теоретической основы, ей служит «Философия искусственного интеллекта», но до тех пор, пока нет значимых результатов в данной сфере, теория не имеет самостоятельного значения. Тем не менее, уже сейчас следует различать науку об искусственном интеллекте и другие теоретические дисциплины и методики (робототехнические, алгоритмические, математические, физиологические), которые имеют самостоятельное значение.

## 1.1. Разновидность компьютерных игр

Существуют различные компьютерные игры.

Разновидность игры зависит от поставленных задач перед игроком.

Существует несколько способов различить игры:

- По классу(настольные игры, спортивные игры, военные игры и другие).
- По искусству проектирования(стратегия или приключения).
- По материалу игры(сюжет).
- По характеру(зависит от характеристик, которые требует игра).
- По тематике (свободный и ограниченный мир, игра от первого лица или от третьего лица).

Формат файла определяет то, какая платформа будет поддерживать данную игру. На ПК установщик файл-exe, на Android файл-apk, для платформ iPhone файл-ios, для кнопочных телефонов нового поколения, как Samsung, SonyEricson, Nokia, предназначены файлы формата Javac разрешение .jar;.jad.

## 1.2. Список игровых жанров

Различные игровые жанры формировались спустя года и во главе у них стоят 2 основных жанра, что определяют игру(однопользовательские или многопользовательские), после же нужно опираться на все остальные. Однопользовательские игры, в основном оффлайн, с редким подключением к интернету, когда многопользовательская - онлайн. И в том и в этом случае можно внедрить ИИ внутрь игрового процесса. Например описать поведение противника при игре оффлайн. Или же проводить анализ в реальном времени действий другого игрока для обнаружения подозрительных действий с его стороны.

Ко второстепенным жанрам игры можно определить:

- Аркады
- Викторины
- Головоломки
- Гонки
- Казино
- Казуальные
- Карточные
- Музыка
- Настольные игры
- Обучающие
- Приключения
- Ролевые
- Симуляторы
- Словесные игры
- Спортивные игры
- Стратегии
- Экшен

## **2. Техническое задание**

Тема работы: Разработка приложения для игры в русские шашки.

Необходимо разработать оконную программу на си-подобном языке с использованием Windows Forms для игры - «Русские шашки».

В данной программе должен быть разработан и задействован вражеский ИИ. Противник будет анализировать игровое поле в режиме реального времени и принимать наилучшее решение для собственного выигрыша. Также будет реализована удобная навигация для передвижения по полю.

В соответствии с поставленной целью в работе должны быть решены следующие задачи: проектирование, разработка.

## 2.1. Назначение разработки

Основным назначением приложения является обеспечения игрового процесса на платформе Windows.

## 2.2. Требование к составу выполняемых функций

Разработка приложения должна обеспечивать возможность перечисленных ниже функций:

- Функция Настройки;
- Функция Проигрыш;
- Функция Рестарт;
- Переход к игровой панели;
- Переход к панели с информацией;
- Переход к панели с главным меню;

## 2.3. Требования к техническому обеспечению

Игра должна быть доступна на платформу Windows.

Минимальные требования представлены в таблице 1.

Таблица 1. Характеристики компьютера:

	Требования к ПК
Операционная система	WINDOWS
Процессор	Двухядерный процессор
Свободное место на диске	75 и более Мб

### 3. Выбор средств разработки

Разработано приложение в WinForms на VisualStudio 2019.

Необходимо выбрать язык программирования для разработки игры

Таблица 2. Различные средства разработки

Средство разработки	Содержание
C#	<p>C# — компилируемый, статически типизированный язык программирования общего назначения.</p> <p>C# широко используется для разработки программного обеспечения, являясь одним из самых популярных языков программирования. Область его применения включает создание операционных систем, разнообразных прикладных программ, драйверов устройств, приложений для встраиваемых систем, высокопроизводительных серверов, а также игр[1].</p>
Visual Studio	<p>Интегрированная среда разработки VisualStudio— это стартовая площадка для написания, отладки и сборки кода, а также последующей публикации приложений. Интегрированная среда разработки (IDE) представляет собой многофункциональную программу, которую можно использовать для различных аспектов разработки программного обеспечения. Помимо стандартного редактора и отладчика, которые существуют в большинстве сред IDE, VisualStudio включает в себя компиляторы, средства автозавершения кода, графические</p>

	конструкторы и многие другие функции для упрощения процесса разработки [2].
WinForms	интерфейс программирования приложений (API), отвечающий за графический интерфейс пользователя и являющийся частью Microsoft .NET Framework. Данный интерфейс упрощает доступ к элементам интерфейса Microsoft Windows за счет создания обёртки для существующего Win32 API в управляемом коде. Причём управляемый код — классы, реализующие API для Windows Forms, не зависят от языка разработки. То есть программист одинаково может использовать Windows Forms как при написании ПО на C#, C++, так и на VB.Net, J# и др.

Microsoft Visual C# — интегрированная среда разработки приложений на языке C#, разработанная корпорацией Microsoft и поставляемая либо как часть комплекта Microsoft Visual Studio, либо отдельно в виде бесплатного функционально ограниченного комплекта Microsoft Visual Studio Community Edition (ранее Visual C# Express Edition). Сменила интегрированную среду разработки Microsoft QuickC.

Visual C# поддерживает перечень приложений как на Managed C# и C#/CLI, так и на обычном C#, и тем самым позволяет генерировать код как для платформы .NET Framework, так и для исполнения в среде «чистой» Windows. В этом отношении Visual C# является уникальным среди других языковых средств, предоставляемых средой Visual Studio, поскольку ни Visual Basic .NET, ни Visual J# неспособны генерировать код для чистого Win32, в отличие от предыдущих версий (Visual Basic и Visual J++ соответственно).

Последняя версия Visual C# 2019 входит в комплект Visual Studio 2019.



## 4. Создание программы

### 4.1. Процесс создания программы

При создании игры была задействована одна из платформ VisualStudio - WinForms. Данная платформа загружается с версией .NET Framework 4.5.1.

В процессе создания интерфейса приложения были задействованы стандартные элементы управления такие как Button, PictureBox, ImageList.

Некоторые элементы управления добавляются на форму в процессе работы приложения.

### 4.2. Описание интерфейса

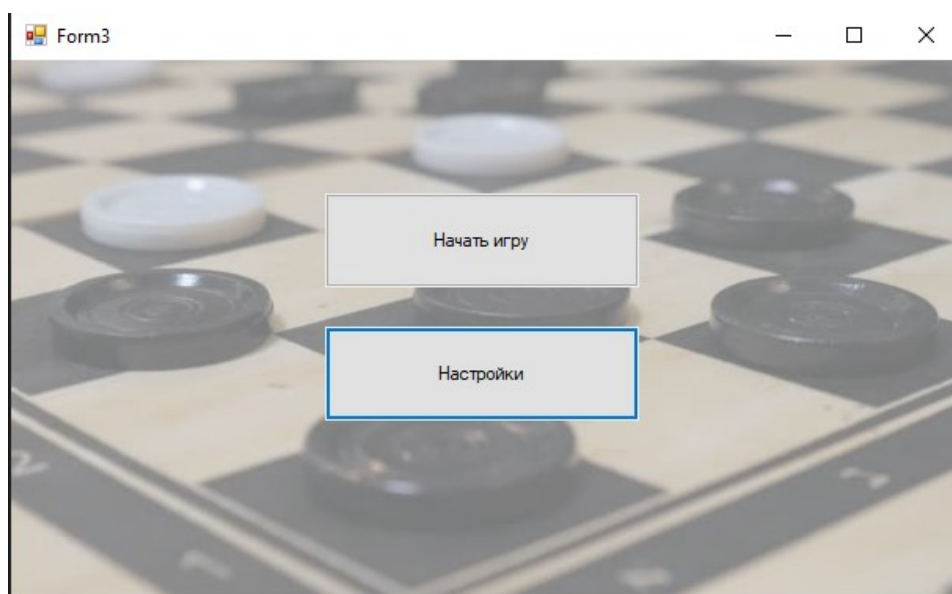


Рисунок 1– Основное меню

Условные обозначения на рисунке:

- 1 –Кнопка «Начать игру»
- 2 –Кнопка «Настройки»

При нажатии кнопки «Играть», вас переносит на игровую панель, где и происходит сам игровой процесс.

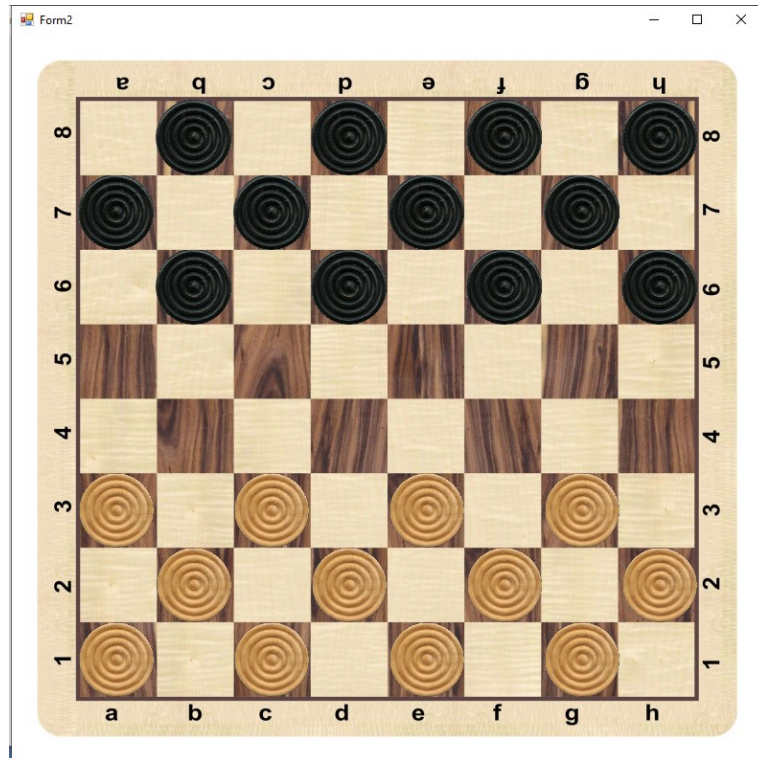


Рисунок 2 –игровая панель

При нажатии на кнопку «Настройки» вы переходите в окно с параметрами, влияющими на процесс игры.

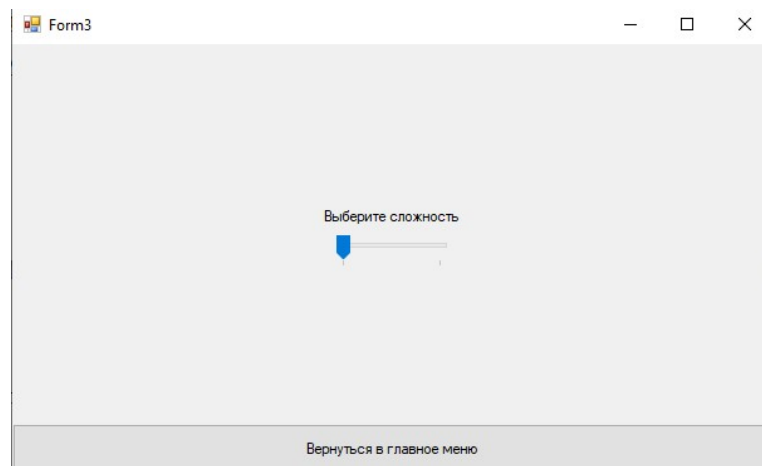


Рисунок 3 - окно с информацией о выполненной работе

В процессе игры игрок может взаимодействовать с фигурами, что представляют из себя шашки.

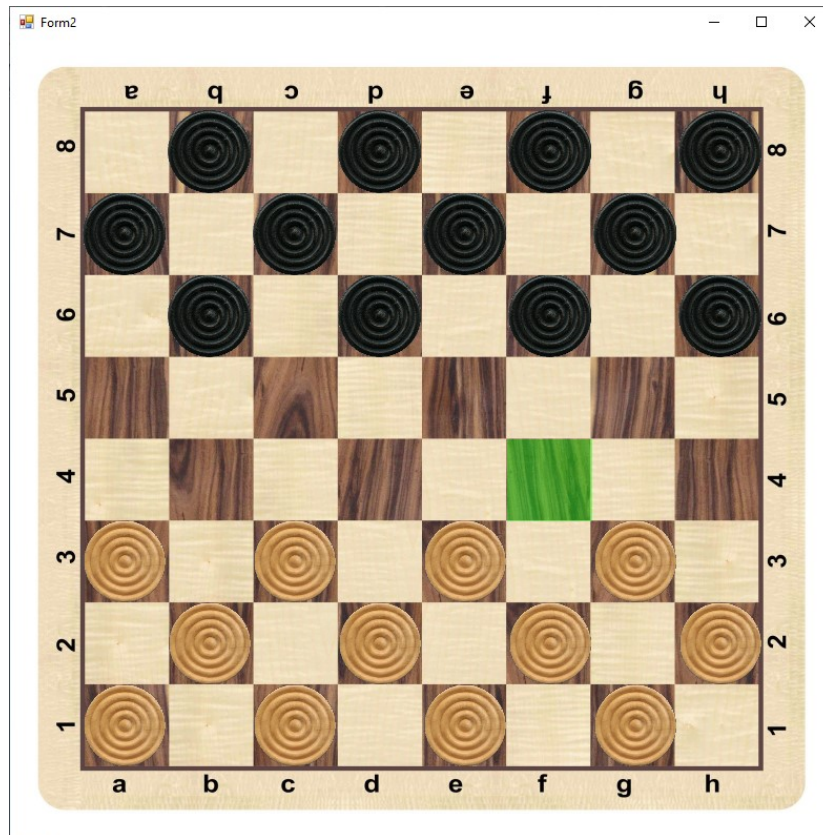


Рисунок 4 – навигация по игровому полю

Из данного окна вы можете перезапустить игру или выйти в главное меню.

### **4.3. Краткие сведения о проделанной работе**

Воссозданная нами игра работает при поддержке платформы WinForms, которая позволяет нам использовать коды, разработанные в Microsoft VisualStudio, непосредственно на элементы нашей игры, прививать им разные значения и правила.

В ходе разработки приложения были изучены такие методы разработки как EventHandler, Task, Async.

### **Заключение**

Проведенная нами работа может показать, что игра не является простейшей программой, она требует к себе большого внимания, так как малейшая ошибка может привести к сбою всей программы.

Нами была разработана игра поддерживающая алгоритмы искусственного интеллекта.

В процессе выполнения работы была освоена технология для разработки оконного приложения WindowsForms, и получены практические навыки в разработке искусственного интеллекта. Были получены навыки по созданию удобных интерфейсов на языке высокого уровня C#.

## Список литературы

1) Бьерн Страуструп. Язык программирования C# = The C# Programming Language / Пер. с англ. — 3-е изд. — СПб.; М.: Невский диалект — Бином, 1999. — 991 с.

2) Ник Рендольф, Дэвид Гарднер, Майкл Минутилло, Крис Андерсон. Visual Studio 2010 для профессионалов = Professional Visual Studio 2010. — М.: «Диалектика», 2011. — 1184 с.

3) Айвор Хортон. Microsoft Visual C# 2005: базовый курс = Beginning Visual C# 2005. — М.: «Диалектика», 2007. — 1152 с.

## ПРИЛОЖЕНИЯ

### Приложение Б. Программный код приложения

```

using ...

namespace CheckersAI
{
    Ссылка: 5
    public partial class Form2 : Form
    {
        public System.Reflection.Assembly assembly = System.Reflection.Assembly.GetExe
        public PictureBox prevBox = new PictureBox();
        public PictureBox prevTipBox = new PictureBox();
        public PictureBox comboAttackBox = new PictureBox();
        public const int N = 7;
        public bool moveBiz = true;
        public int attackCount = 0;
        public short blackCount = 0;
        public short whiteCount = 0;
        public int enemyAttackI = 0;
        public int enemyAttackJ = 0;
        public int enemyAttackIDam = 0;
        public int enemyAttackJDam = 0;
        public int enemyAttackIDamWhiteDel = 0;
        public int enemyAttackJDamWhiteDel = 0;
        public short side = 0;
        public int safeI = 0;
        public int safeJ = 0;
        public int DamI = 0;
        public int DamJ = 0;
        public int globalI = 0;
        public int globalJ = 0;
        public int priorI = 0;
        public int priorJ = 0;
        public int attackPoint = 0;
        public int enemyAttackPoint = -15;
        public int generalPoint = -15;
        public PictureBox choosenBox = new PictureBox();
        public Color tipBoxClr = new Color();
        public int maxWhatWhite = 0;
        public bool isAttack = false;
        public int globalIWhite = 0;
        public int globalJWhite = 0;
        public int[,] RGmap = new int[8, 8]...;
        public int[,] map = new int[8, 8]...;
        public int[] mapPr = new int[12] { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
        public int[] mapPrX = new int[12] { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
        public int[] mapPrXx = new int[12] { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
        public int[] mapPrDam = new int[12] { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
        public int[] mapPrXDam = new int[12] { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
        public int[] mapPrXxDam = new int[12] { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
        ссылка: 1
        public Form2()...
        ссылка: 1
        private void Form2_Load(object sender, EventArgs e)...
        Ссылка: 8
        public int GetEnemyAttackPoint() ...
        public int moveForwardIRight = 0;
        public int moveForwardJRight = 0;
        public int moveForwardILeft = 0;
        public int moveForwardJLeft = 0;
        public int moveForwardI = 0;
        public int moveForwardJ = 0;
        public bool chLeft = false;
    }
}

```



```

public bool chRight = false;
public bool chDam1 = false;
public bool chDam2 = false;
public bool chDam3 = false;
public bool chDam4 = false;
public bool chDam1H = false;
public bool chDam2H = false;
public bool chDam3H = false;
public bool chDam4H = false;
public bool chDam1HR = false;
public bool chDam2HR = false;
public bool chDam3HR = false;
public bool chDam4HR = false;
public int moveForwardI1 = 0;
public int moveForwardJ1 = 0;
public int moveForwardI2 = 0;
public int moveForwardJ2 = 0;
public int moveForwardI3 = 0;
public int moveForwardJ3 = 0;
public int moveForwardI4 = 0;
public int moveForwardJ4 = 0;
Ссылка: 1
public bool MoveForwardRight(int ti, int tj) ...
Ссылка: 1
public bool MoveForwardLeft(int ti, int tj) ...
Ссылка: 2
public bool MoveForwardDam1(int ti, int tj, int qu) ...
Ссылка: 2
public bool MoveForwardDam2(int ti, int tj, int qu) ...
Ссылка: 2
public bool MoveForwardDam3(int ti, int tj, int qu) ...
Ссылка: 2
public bool MoveForwardDam4(int ti, int tj, int qu) ...
Ссылка: 0
public void OutputMapPr(int[] numbers) ...
Ссылка: 0
public void OutputMapPr2(int[, ] numbers) ...
Ссылка: 1
public async Task GenerateMoveEnemyAI() {
    int k02 = 0, k01 = 0;
    int maxWhatStatic = 0;
    int maxWhatStaticDam = 0;
    maxWhat = 0;
    int mode = 0;
    generalPoint = -15;
    int tempChe = 0;
    for (int i = 7; i >= 0; i--)
    {
        for (int j = 7; j >= 0; j--)
        {
            int prem = 0;
            attackCount = 0;
            if (map[i, j] == 2 && !moveBiz)
            {
                for (int q = 0; q < 12; q++) mapPr[q] = 0;
                gI = i;
                gJ = j;
                globalI = i;
                globalJ = j;
                blackCount++;
                maxWhat = 0;
                AttackWays(i, j);
            }
        }
    }
}

```

```

MoveForwardRight(1, j);
MoveForwardLeft(i, j);
for (int q = 0; q < 12; q++) mapPrx[q] = mapPr[q];
if (maxWhat != 0)
{
    int currentI = i, currentJ = j;
    mapPrx[maxWhat / 10] = i * 10 + j;
    for (int il = maxWhat / 10 - 1; il >= 0; il--)
    {
        if (mapPrx[il] != 0)
        {
            map[mapPrx[il] / 10, mapPrx[il] % 10] = 2;
            map[currentI, currentJ] = 0;
            tempChe = map[(currentI + mapPrx[il] / 10) / 2, (currentJ + mapPrx[il] % 10) / 2];
            map[(currentI + mapPrx[il] / 10) / 2, (currentJ + mapPrx[il] % 10) / 2] = 0;
            currentI = mapPrx[il] / 10;
            currentJ = mapPrx[il] % 10;
        }
    }
    if (currentI == 7) prem = prem + 1;
    enemyAttackPoint = GetEnemyAttackPoint();
    for (int il = 1; il <= maxWhat / 10; il++)
    {
        if (mapPrx[il] != 0)
        {
            map[mapPrx[il] / 10, mapPrx[il] % 10] = 2;
            map[currentI, currentJ] = 0;
            map[(currentI + mapPrx[il] / 10) / 2, (currentJ + mapPrx[il] % 10) / 2] = tempChe;
            currentI = mapPrx[il] / 10;
            currentJ = mapPrx[il] % 10;
        }
    }
    attackPoint = maxWhat / 10 + prem;
    if (attackPoint - enemyAttackPoint >= generalPoint)
    {
        for (int q = 0; q < 12; q++) mapPrx[q] = mapPr[q];
        generalPoint = attackPoint - enemyAttackPoint;
        priorI = i;
        priorJ = j;
        maxWhatStatic = maxWhat;
        mode = 1;
    }
}
if (chLeft || chRight)
{
    int tempL = -15;
    int tempR = -15;
    int pary = 0;
    if (chRight)
    {
        map[i, j] = 0;
        map[moveForwardIRight, moveForwardJRight] = 2;
        tempR = -GetEnemyAttackPoint();
        map[i, j] = 2;
        map[moveForwardIRight, moveForwardJRight] = 0;
        if (moveForwardIRight == 7) tempR++;
    }
    if (chLeft)
    {
        map[i, j] = 0;

```



```

        map[moveForwardILeft, moveForwardJLeft] = 2;
        tempL = -GetEnemyAttackPoint();
        map[i, j] = 2;
        map[moveForwardILeft, moveForwardJLeft] = 0;
        if (moveForwardILeft == 7) tempL++;
    }
    if (tempL == tempR)
    {
        Random rnd = new Random();
        pary = rnd.Next(1, 3);
    }
    if (tempL >= generalPoint && (tempL > tempR || pary == 1) && chLeft)
    {
        generalPoint = tempL;
        priorI = i;
        priorJ = j;
        moveForwardI = moveForwardILeft;
        moveForwardJ = moveForwardJLeft;
        mode = 3;
    }
    if (tempR >= generalPoint && (tempR > tempL || pary == 2) && chRight)
    {
        generalPoint = tempR;
        priorI = i;
        priorJ = j;
        moveForwardI = moveForwardIRight;
        moveForwardJ = moveForwardJRight;
        mode = 3;
    }
}
}
else if (!moveBiz && map[i, j] == 22)
{
    blackCount++;
    chDam1H = false;
    chDam2H = false;
    chDam3H = false;
    chDam4H = false;
    maxWhatDam = 0;
    AttackWaysDam(i, j);
    for (int q = 0; q < 12; q++) mapPrxDam[q] = mapPrDam[q];
    if (maxWhatDam != 0)
    {
        int currentI = i, currentJ = j;
        mapPrxDam[maxWhatDam] = i * 10 + j;
        for (int il = maxWhatDam - 1; il >= 0; il--)
        {
            if (mapPrxDam[il] != 0)
            {
                if (mapPrxDam[il] / 10 - currentI > 0) ko1 = 1;
                else ko1 = -1;
                if (mapPrxDam[il] % 10 - currentJ > 0) ko2 = 1;
                else ko2 = -1;
                map[mapPrxDam[il] / 10, mapPrxDam[il] % 10] = 22;
                map[currentI, currentJ] = 0;
                tempChe = map[mapPrxDam[il] / 10 - ko1, mapPrxDam[il] % 10 - ko2];
                map[mapPrxDam[il] / 10 - ko1, mapPrxDam[il] % 10 - ko2] = 0;
                currentI = mapPrxDam[il] / 10;
                currentJ = mapPrxDam[il] % 10;
            }
        }
    }
}

```

```

}
enemyAttackPoint = GetEnemyAttackPoint();
for (int il = 1; il <= maxWhatDam; il++)
{
    if (mapPrXDam[il] != 0)
    {
        if (mapPrXDam[il] / 10 - currentI > 0) ko1 = 1;
        else ko1 = -1;
        if (mapPrXDam[il] % 10 - currentJ > 0) ko2 = 1;
        else ko2 = -1;
        map[mapPrXDam[il] / 10, mapPrXDam[il] % 10] = 22;
        map[currentI, currentJ] = 0;
        map[currentI + ko1, currentJ + ko2] = tempChe;
        currentI = mapPrXDam[il] / 10;
        currentJ = mapPrXDam[il] % 10;
    }
}
attackPoint = maxWhatDam;
if (attackPoint - enemyAttackPoint >= generalPoint)
{
    for (int q = 0; q < 12; q++) mapPrXDam[q] = mapPrXDam[q];
    generalPoint = attackPoint - enemyAttackPoint;
    priorI = i;
    priorJ = j;
    maxWhatStaticDam = maxWhatDam;
    mode = 4;
}
}
for (int chcker = 1; chcker < 7; chcker++) {
    MoveForwardDam1(i, j, chcker);
    MoveForwardDam2(i, j, chcker);
    MoveForwardDam3(i, j, chcker);
    MoveForwardDam4(i, j, chcker);
}
chDam1HR = true;
chDam2HR = true;
chDam3HR = true;
chDam4HR = true;
if (chDam1H || chDam2H || chDam3H || chDam4H)
{
    int temp1 = -15;
    int temp2 = -15;
    int temp3 = -15;
    int temp4 = -15;
    int pary = 0;
    for (int chcker = 1; chcker < 7; chcker++)
    {
        MoveForwardDam1(i, j, chcker);
        MoveForwardDam2(i, j, chcker);
        MoveForwardDam3(i, j, chcker);
        MoveForwardDam4(i, j, chcker);
        if (chDam1 && chDam1HR)
        {
            map[i, j] = 0;
            map[moveForwardI1, moveForwardJ1] = 22;
            temp1 = -GetEnemyAttackPoint();
            map[i, j] = 22;
            map[moveForwardI1, moveForwardJ1] = 0;
        }
    }
}

```

```

if (chDam2 && chDam2HR)
{
    map[i, j] = 0;
    map[moveForwardI2, moveForwardJ2] = 22;
    temp2 = -GetEnemyAttackPoint();
    map[i, j] = 22;
    map[moveForwardI2, moveForwardJ2] = 0;
}
if (chDam3 && chDam3HR)
{
    map[i, j] = 0;
    map[moveForwardI3, moveForwardJ3] = 22;
    temp3 = -GetEnemyAttackPoint();
    map[i, j] = 22;
    map[moveForwardI3, moveForwardJ3] = 0;
}
if (chDam4 && chDam4HR)
{
    map[i, j] = 0;
    map[moveForwardI4, moveForwardJ4] = 22;
    temp4 = -GetEnemyAttackPoint();
    map[i, j] = 22;
    map[moveForwardI4, moveForwardJ4] = 0;
}
if (temp1 == temp2 && temp3 == temp4 && temp2 == temp3)
{
    Random rnd = new Random();
    pary = rnd.Next(1, 5);
}
if (temp1 >= generalPoint /* && (temp1 > temp2 && temp1 > temp3 && temp1 > temp4 || pary == 1)*/ && chDam1HR)
{
    generalPoint = temp1;
    priorI = i;
    priorJ = j;
    moveForwardI = moveForwardI1;
    moveForwardJ = moveForwardJ1;
    mode = 5;
}
if (temp2 >= generalPoint /* && (temp2 > temp1 && temp2 > temp3 && temp2 > temp4 || pary == 2)*/ && chDam2HR)
{
    generalPoint = temp2;
    priorI = i;
    priorJ = j;
    moveForwardI = moveForwardI2;
    moveForwardJ = moveForwardJ2;
    mode = 5;
}
if (temp3 >= generalPoint /*&& (temp3 > temp2 && temp3 > temp1 && temp3 > temp4 || pary == 3)*/ && chDam3HR)
{
    generalPoint = temp3;
    priorI = i;
    priorJ = j;
    moveForwardI = moveForwardI3;
    moveForwardJ = moveForwardJ3;
    mode = 5;
}
if (temp4 >= generalPoint /*&& (temp4 > temp2 && temp4 > temp3 && temp4 > temp1 || pary == 4)*/ && chDam4HR)
{
    generalPoint = temp4;
    priorI = i;
}

```





```

    {
        if (mapPrxxDam[i1] != 0)
        {
            if (mapPrxxDam[i1] / 10 - currentIDam > 0) ko1 = 1;
            else ko1 = -1;
            if (mapPrxxDam[i1] % 10 - currentJDam > 0) ko2 = 1;
            else ko2 = -1;
            map[mapPrxxDam[i1] / 10, mapPrxxDam[i1] % 10] = 22;
            map[currentIDam, currentJDam] = 0;
            map[mapPrxxDam[i1] / 10 - ko1, mapPrxxDam[i1] % 10 - ko2] = 0;
            currentIDam = mapPrxxDam[i1] / 10;
            currentJDam = mapPrxxDam[i1] % 10;
            System.IO.Stream resourceStream1 =
                assembly.GetManifestResourceStream(@"CheckersAI.game_piece_movement_09.wav");
            SoundPlayer sndpl = new SoundPlayer(resourceStream1);
            await Task.Delay(500);
            this.Controls.Clear();
            RefreshMapImage();
            sndpl.Play();
        }
    }
    moveBiz = true;
    break;
case 5:
    System.IO.Stream resourceStream4 =
        assembly.GetManifestResourceStream(@"CheckersAI.movement_01 (online-audio-converter.com).wav");
    SoundPlayer player4 = new SoundPlayer(resourceStream4);
    await Task.Delay(500);
    map[priorI, priorJ] = 0;
    map[moveForwardI, moveForwardJ] = 22;
    moveBiz = true;
    player4.Play();
    break;
default:
    System.IO.Stream resourceStream2 =
        assembly.GetManifestResourceStream(@"CheckersAI.Victory.wav");
    SoundPlayer player2 = new SoundPlayer(resourceStream2);
    moveBiz = true;
    player2.Play();
    MessageBox.Show("Противник сдается");
    break;
}

this.Controls.Clear();
RefreshMapImage();
WhiteCountCheckers();
}

public int attackCounter = 0;
public int maxWhatDam = 0;
public int currentPosDam = 0;
public int mapPrCountDam = 0;
public bool isEndOfAttcakDam= false;
Ссылка: 5
public bool AttackWaysDam(int ti, int tj) {
    bool mdx = false , mdy = false;
    if (ti > 0 && tj > 0 && stageway != 4)
        for (int attackedPostI = ti - 1, attackedPostJ = tj - 1 ;; attackedPostI--, attackedPostJ--) {
            if (attackedPostI - 1 < 0 && attackedPostJ - 1 < 0 || attackedPostI - 1 < 0 || attackedPostJ - 1 < 0) break;
            if (map[attackedPostI, attackedPostJ] == 2 || map[attackedPostI, attackedPostJ] == 22) break;
            if ((map[attackedPostI, attackedPostJ] == 1 || map[attackedPostI, attackedPostJ] == 11) && (map[attackedPostI + 1, attackedPostJ + 1] == 0

```

```

if (map[attackedPostI, attackedPostJ] == 11) map[attackedPostI, attackedPostJ] = -11;
int nextAtkI = attackedPostI - 1, nextAtkJ = attackedPostJ - 1;
if (!AttackWaysDam(nextAtkI, nextAtkJ))
{
    if (attackCounter > maxWhatDam)
    {
        currentPosDam = attackCounter;
        maxWhatDam = attackCounter;
        isEndOfAttcakDam = true;
        mapPrCountDam = 0;
    }
}
if (isEndOfAttcakDam && currentPosDam == attackCounter)
{
    mapPrDam[mapPrCountDam] = 10 * (attackedPostI - 1) + (attackedPostJ - 1);
    mapPrCountDam++;
    currentPosDam--;
}
if (maxWhatDam - mapPrCountDam == 0)
{
    isEndOfAttcakDam = false;
    mapPrCountDam = 0;
}
if (map[attackedPostI, attackedPostJ] == -1) map[attackedPostI, attackedPostJ] = 1;
if (map[attackedPostI, attackedPostJ] == -11) map[attackedPostI, attackedPostJ] = 11;
attackCounter--;
}
}
if (ti > 0 && tj < 7 && stageWay != 3)
for (int attackedPostI = ti - 1, attackedPostJ = tj + 1; ; attackedPostI--, attackedPostJ++)
{
    if (attackedPostI - 1 < 0 && attackedPostJ + 1 > 7 || attackedPostI - 1 < 0 || attackedPostJ + 1 > 7) break;
    if (map[attackedPostI, attackedPostJ] == 2 || map[attackedPostI, attackedPostJ] == 22) break;
    if ((map[attackedPostI, attackedPostJ] == 1 || map[attackedPostI, attackedPostJ] == 11) && map[attackedPostI - 1, attackedPostJ + 1] == 0
        && (map[attackedPostI + 1, attackedPostJ - 1] == 0 || map[attackedPostI + 1, attackedPostJ - 1] == 22))
    {
        stageWay = 2;
        attackCounter++;
        mdx = true;
        if (map[attackedPostI, attackedPostJ] == 1) map[attackedPostI, attackedPostJ] = -1;
        if (map[attackedPostI, attackedPostJ] == 11) map[attackedPostI, attackedPostJ] = -11;
        int nextAtkI = attackedPostI - 1, nextAtkJ = attackedPostJ + 1;
        if (!AttackWaysDam(nextAtkI, nextAtkJ))
        {
            if (attackCounter > maxWhatDam)
            {
                currentPosDam = attackCounter;
                maxWhatDam = attackCounter;
                isEndOfAttcakDam = true;
                mapPrCountDam = 0;
            }
        }
    }
    if (isEndOfAttcakDam && currentPosDam == attackCounter)
    {
        mapPrDam[mapPrCountDam] = 10 * (attackedPostI - 1) + (attackedPostJ + 1);
        mapPrCountDam++;
        currentPosDam--;
    }
    if (maxWhatDam - mapPrCountDam == 0)
    {

```

```

    }
    if (maxiwhatDam - mapPrCountDam == 0)
    {
        isEndOfAttcakDam = false;
        mapPrCountDam = 0;
    }
    if (map[attackedPostI, attackedPostJ] == -1) map[attackedPostI, attackedPostJ] = 1;
    if (map[attackedPostI, attackedPostJ] == -11) map[attackedPostI, attackedPostJ] = 11;
    attackCounter--;
}
}
if (ti < 7 && tj > 0 && stageway != 2)
for (int attackedPostI = ti + 1, attackedPostJ = tj - 1; ; attackedPostI++, attackedPostJ--)
{
    if (attackedPostI + 1 > 7 && attackedPostJ - 1 < 0 || attackedPostI + 1 > 7 || attackedPostJ - 1 < 0) break;
    if (map[attackedPostI, attackedPostJ] == 2 || map[attackedPostI, attackedPostJ] == 22) break;
    if ((map[attackedPostI, attackedPostJ] == 1 || map[attackedPostI, attackedPostJ] == 11) && map[attackedPostI + 1, attackedPostJ - 1] == 0
        && (map[attackedPostI - 1, attackedPostJ + 1] == 0 || map[attackedPostI - 1, attackedPostJ + 1] == 22))
    {
        stageway = 3;
        attackCounter++;
        mdx = true;
        if (map[attackedPostI, attackedPostJ] == 1) map[attackedPostI, attackedPostJ] = -1;
        if (map[attackedPostI, attackedPostJ] == 11) map[attackedPostI, attackedPostJ] = -11;
        int nextAtkI = attackedPostI + 1, nextAtkJ = attackedPostJ - 1;
        if (!AttackwaysDam(nextAtkI, nextAtkJ))
        {
            if (attackCounter > maxiwhatDam)
            {
                currentPosDam = attackCounter;
                maxiwhatDam = attackCounter;
                isEndOfAttcakDam = true;
                mapPrCountDam = 0;
            }
        }
        if (isEndOfAttcakDam && currentPosDam == attackCounter)
        {
            mapPrDam[mapPrCountDam] = 10 * (attackedPostI + 1) + (attackedPostJ - 1);
            mapPrCountDam++;
            currentPosDam--;
        }
        if (maxiwhatDam - mapPrCountDam == 0)
        {
            isEndOfAttcakDam = false;
            mapPrCountDam = 0;
        }
        if (map[attackedPostI, attackedPostJ] == -1) map[attackedPostI, attackedPostJ] = 1;
        if (map[attackedPostI, attackedPostJ] == -11) map[attackedPostI, attackedPostJ] = 11;
        attackCounter--;
    }
}
if (ti < 7 && tj < 7 && stageway != 1)
for (int attackedPostI = ti + 1, attackedPostJ = tj + 1; ; attackedPostI++, attackedPostJ++)
{
    if (attackedPostI + 1 > 7 && attackedPostJ + 1 > 7 || attackedPostI + 1 > 7 || attackedPostJ + 1 > 7) break;
    if (map[attackedPostI, attackedPostJ] == 2 || map[attackedPostI, attackedPostJ] == 22) break;
    if ((map[attackedPostI, attackedPostJ] == 1 || map[attackedPostI, attackedPostJ] == 11) && map[attackedPostI + 1, attackedPostJ + 1] == 0
        && (map[attackedPostI - 1, attackedPostJ - 1] == 0 || map[attackedPostI - 1, attackedPostJ - 1] == 22))
    {
        stageway = 4;
        public void isATTACKEDNOW() ...
        public bool isWayCheckDam(int ti, int tj, int chI, int chJ) ...
        public int DeleteBlackCheckersOnDamWay() ...
        public bool isAttackCheck(PictureBox tempBox) ...
        public void RefreshField() ...
        public void RefreshMapImage() ...
        public bool cAttackNOW = false;
        public void ClickOnCell(object sender, EventArgs e) ...
        public void MouseEnterInCell(object sender, EventArgs e) ...
        public static int CordXToIndex(int x) ...
        public static int CordYToIndex(int y) ...
        private void textBox1_TextChanged(object sender, EventArgs e) ...
        private void button1_Click(object sender, EventArgs e) ...
        private void textBox1_MouseClick(object sender, MouseEventArgs e) ...
    }
}
}

```

```

        attackCounter++;
        mdx = true;
        if (map[attackedPostI, attackedPostJ] == 1) map[attackedPostI, attackedPostJ] = -1;
        if (map[attackedPostI, attackedPostJ] == 11) map[attackedPostI, attackedPostJ] = -11;
        int nextAtkI = attackedPostI + 1, nextAtkJ = attackedPostJ + 1;
        if (!AttackWaysDam(nextAtkI, nextAtkJ))
        {
            if (attackCounter > maxWhatDam)
            {
                currentPosDam = attackCounter;
                maxWhatDam = attackCounter;
                isEndOfAttcakDam = true;
                mapPrCountDam = 0;
            }
        }
        if (isEndOfAttcakDam && currentPosDam == attackCounter)
        {
            mapPrDam[mapPrCountDam] = 10 * (attackedPostI + 1) + (attackedPostJ + 1);
            mapPrCountDam++;
            currentPosDam--;
        }
        if (maxWhatDam - mapPrCountDam == 0)
        {
            isEndOfAttcakDam = false;
            mapPrCountDam = 0;
        }
        if (map[attackedPostI, attackedPostJ] == -1) map[attackedPostI, attackedPostJ] = 1;
        if (map[attackedPostI, attackedPostJ] == -11) map[attackedPostI, attackedPostJ] = 11;
        attackCounter--;
    }
}

return mdx;
}

public int willAttackedWhiteCheckerCount = 0;
public byte stageway = 0;
public bool isEndOfAttackWay = false;
public int maxWhat = 0;
public int mapPrCount = 0;
public int currentpos = 0;
Ссылка: 5
public bool AttackWays(int ti, int tj) ...
public int gI = 0;
public int gJ = 0;
public int willAttackedWhiteCheckerCountWhite = 0;
Ссылка: 6
public bool AttackWaysWhite(int ti, int tj) ...
Ссылка: 1
public bool WhiteCountCheckers() ...
Ссылка: 1
public bool AttackContinuesDam(int ti, int tj) ...

Ссылка: 1
public bool AttackContinues(int ti, int tj) ...
Ссылка: 1
public void isAttackedNow() ...
Ссылка: 1
public bool isWayCheckDam(int ti, int tj, int chI, int chJ) ...
Ссылка: 1
public int DeleteBlackCheckersOnDamWay() ...
Ссылка: 2
public bool isAttackCheck(PictureBox tempBox) ...
Ссылка: 2
public void RefreshField() ...

```