МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Вятский государственный университет»**

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

Допущено к защите

Руководитель проекта

Преподаватель

\_\_\_\_\_\_\_\_\_\_\_\_ (Клюкин В.Л.)

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_2018г.

Разработка 2D игры “Уголки”

Пояснительная записка курсовой работы по дисциплине

«Компьютерная графика»

ТПЖА.09.03.01.014 ПЗ

Разработал студент группы ИВТ-21 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Седов М. Д./

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ Клюкин В.Л./

Работа защищена с оценкой «\_\_\_\_\_\_\_\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(оценка) (дата)*

Члены комиссии \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_//

(подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ //

(подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ //

(подпись)

Киров 2018

**Реферат**

Седов М.Д. Разработка 2D игры “Уголки”: ТПЖА.090301.014 ПЗ: Курс. работа / ВятГУ, каф. ЭВМ; рук. Клюкин В.Л. - Киров, 2018. – ПЗ 72 с, 12 рис., 2 табл., 21 источник.

СПРАВОЧНИК, АНАЛОГИ, ЗАДАНИЕ, МАТЕМАТИЧЕСКИЙ АППАРАТ, ДИАГРАММЫ КЛАССОВ, ИНТЕРФЕЙС, СТРУКТУРА ИНТЕРФЕЙСА СХЕМА АЛГОРИТМА, МОДУЛЬНАЯ СТРУКТУРА

Объект разработки – 2D игра “Уголки”

Цель курсовой работы – упростить и сделать удобную работу с компьютерной 2D игрой “Уголки” без доступа в сеть интернет, с возможностью изменения цвета шашек для каждой из команд, изменения цвета доски. Также с реализацией поиска наилучшего хода для компьютера при помощи алгоритма MinMax.

Разработанный продукт позволяет пользователям осуществить игру на персональном комьютере.

Разработан программный продукт, реализующий функции, необходимые для перемещения шашек по доске, проверки правильности хода, поиска наилучшего хода, построения дерева всевозможных комбинаций с заданной точностью, эвристический алгоритм для построения весов каждой из клеток поля.

**Оглавление**

[**Введение:** 3](#_Toc532764381)

[**1.** **Аналоги приложения: их плюсы и их минусы** 5](#_Toc532764382)

[**1.1.** **Уголки (online)** 5](#_Toc532764383)

[**1.2.** **Уголки(android)** 5](#_Toc532764384)

[**1.3.** **Игра “Уголки” 1.4** 5](#_Toc532764385)

[**1.4.** **Игра Халма(уголки)** 6](#_Toc532764386)

[**1.5.** **Уголки-русская игра на JavaScript** 6](#_Toc532764387)

[**2.** **Актуальность темы** 8](#_Toc532764388)

[**3.1** **Программный продукт должен обеспечивать** 9](#_Toc532764389)

[**3.2** **Требования к входным и выходным данным** 9](#_Toc532764390)

[**3.3** **Требования к программному обеспечению** 9](#_Toc532764391)

[**3.4** **Требования к интерфейсу** 9](#_Toc532764392)

[**4** **Схема функционального взаимодействия блоков** 10](#_Toc532764393)

[**5** **Математический аппарат** 12](#_Toc532764394)

[**6** **Диаграммы классов** 14](#_Toc532764395)

[**7** **Разработка алгоритмов** 15](#_Toc532764396)

[**7.1** **Функция «Поиск лучшего хода компьютера»** 15](#_Toc532764397)

[**8** **Разработка структуры интерфейса пользователя** 16](#_Toc532764398)

[**9** **Разработка модульной структуры проекта** 17](#_Toc532764399)

[**Заключение** 18](#_Toc532764400)

[**Приложение А** 19](#_Toc532764401)

[**Приложение Б** 20](#_Toc532764402)

[**Приложение В** 21](#_Toc532764403)

[**Приложение Е** 24](#_Toc532764404)

[**Приложение З** 52](#_Toc532764405)

[**Биографический список** 53](#_Toc532764406)

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | | | | | |
|  |  |  |  |  |
| Изм. | Лист | № докум | Подпись | Дата |
| Разраб | | Седов |  |  |  | Литера | | | Лист | Листов |
| Пров | | Клюкин |  |  |  | П |  | 2 | 51 |
|  | |  |  |  | *Кафедра ЭВМ*  *Группа ИВТ-21* | | | | |
| Н. Контр. | |  |  |  |
| Утв | | Страбыкин |  |  |

# **Введение:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 3 |
| Изм. | Лист | № докум | Подпись | Дата |

Все знания, которые человек накапливает в течение своей жизни, не могут вечно храниться в памяти. С древнейших времен люди понимали, что невозможно все сохранить в памяти и стремились перенести важную накопленную информацию на внешних носителях.

Один из первых способов хранения информации на внешних носителях – наскальные надписи. С развитием человечества, появлялись все более новые способы хранения информации, например, древнеегипетское письмо и на глиняных табличках. С появлением алфавитного письма хранение информации начинается на книжных носителях. Но технологии не стоят на месте, и в современном мире информация храниться на многих внешних носителях такие как, оптические носители, жесткие диски, флеш-накопители, сетевые хранилища. С появлением сети Интернет, доступ к огромным объемам информации гораздо упростился, нужно лишь было иметь доступ к нему.

В современном мире у каждого человека есть персональный компьютер с доступом к сети Интернет. С его помощью можно искать информацию в сети Интернет или же в электронных версиях книг, которые предварительно можно загрузить на него и использовать без доступа к сети.

Всё это показывает преимущества электронных устройств над бумажными. Поэтому было принято решение разработать приложение, позволяющее без доступа к сети Интернет, осуществлять игру против реального игрока на одном компьютере, либо игру против компьютера в игру «Уголки».

Приложения, выполняющие подобную функцию, уже существуют, но аналоги, имеющиеся в свободном доступе, имеют свои недоработки. Поэтому разработка собственной игры «Уголки» должна исключать описанные недостатки и иметь новый функционал, что позволяет ему конкурировать на рынке уже с имеющимися продуктами этой же категории.

# **Аналоги приложения: их плюсы и их минусы**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 4 |
| Изм. | Лист | № докум | Подпись | Дата |

## **Уголки (online)**

Плюсы:

1. Простота освоения интерфейса.
2. Просмотр результатов уже проведенных игр.
3. Довольно высокий уровень интеллекта “компьютера”

Минусы:

1. Возможность игры только online.
2. Отсутствие возможности игры двух людей друг с другом
3. Отсутствие игры за шашки разных цветов.
4. Отсутствие какой-либо анимации.
5. Нет возможности сохранить результаты текущей игры

## **Уголки(android)**

Плюсы:

1. Низкие аппаратные требования (OC Android 2x).
2. Возможность игры на мобильном устройстве (высокая мобильность при использовании)
3. Простота интерфейса.

Минусы:

1. Отсутствие возможности игры двух людей друг с другом.
2. Отсутствие какой-либо анимации.
3. Отсутствие игры за шашки разных цветов.
4. Нет возможности сохранить результаты текущей игры.

## **Игра “Уголки” 1.4**

Плюсы:

1. Простота использования интерфейса.
2. Вывод списка пройденных ходов текущей игры.
3. Функция, рассчитывающая примерный процент победы.

Минусы:

1. Отсутствие возможности игры двух людей друг с другом.
2. Отсутствие какой-либо анимации.
3. Шашки сделаны в форме пешек, нет возможности изменения цвета как самих шашек, так и доски.

## **Игра Халма(уголки)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 5 |
| Изм. | Лист | № докум | Подпись | Дата |

Плюсы:

1. Интуитивно понятный интерфейс.
2. Возможность изменения цветов шашек, фона.
3. Возможность сохранения, открытия игр.
4. Демонстрационная игра компьютера против компьютера

Минусы:

1. Отсутствие какой-либо анимации.
2. Отсутствие возможности игры двух людей друг с другом.
3. Нет проверки на наличие своих шашек в “доме” после 40 хода
4. Отсутствие информации о победе какой-либо из сторон.

## **Уголки-русская игра на JavaScript**

Плюсы:

1. Минимализм интерфейса.
2. Сохранение результатов прошлых игр.
3. Минимальные аппаратные требования.
4. Существует проверка на наличие своих шашек в “доме” после 40 хода.

Минусы:

1. Отсутствие какой-либо анимации.
2. Отсутствие возможности игры двух людей друг с другом.
3. Возможность игры только online.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Аналоги  Функции | Уголки  (online) | Уголки  (android) | Уголки  “1.4” | Халма  (Уголки) | Уголки-  Русская игра на  JavaScript |
| Анимация передвижения шашек | **-** | **+** | **-** | **-** | **-** |
| Сохранение(импорт) ключевых свойств проекта | **-** | **-** | **+** | **+** | **+-** |
| Возможность игры как против компьютера, так и против человека | **-** | **-** | **-** | **-** | **-** |
| Возможность игры online | **+** | **-** | **-** | **-** | **+** |
| Разработка приложения под ОС windows | **-** | **-** | **+** | **+** | **-** |

Таблица 1 – Сравнение аналогов

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 6 |
| Изм. | Лист | № докум | Подпись | Дата |

# **Актуальность темы**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 7 |
| Изм. | Лист | № докум | Подпись | Дата |

В настоящее время нет другого проекта, который бы соответствовал всем необходимым требованиям: возможность игры как против компьютера, так и против реального игрока, реализация анимации перемещения шашек, изменение цвета шашек, изменение цвета доски, сохранение настроек проведенной игры, открытие настроек, высокий уровень интеллекта компьютера.

При учете всех вышесказанных критерий, был разработан и осуществлен данный проект, актуальность которого подтверждается.

1. **Постановка задачи**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 8 |
| Изм. | Лист | № докум | Подпись | Дата |

Разработать 2D игру “Уголки” для операционной системы Windows. Цель данной игры – создание приложения с простым интерфейсом и управлением, создание игры против компьютера (а следовательно и создание всех алгоритмов, эвристических просчетов для этого), создание игры против реального игрока на одном компьютере, визуализация перемещения шашек, проверка на допустимый ход, индикатор выбранной клетки, проверка на досрочную победу одной из команд.

## **Программный продукт должен обеспечивать**

* Возможность игры как против компьютера, так и против реального игрока.
* Изменения цветов шашек каждой из команд.
* Изменение цвета доски.
* Возможность сохранения (открытия) настроек приложения (цветов, списка прошлой игры).
* Поиск наилучшего хода для компьютера алгоритмом MinMax
* Создание дерева всевозможных комбинаций из ходов компьютера.
* Отображение и определения победителя игры.
* Проверки на досрочную победу (проигрыш) одного из участников.

## **Требования к входным и выходным данным**

* На вход принимаются сигналы нажатия на клавиши мыши с дальнейшей реакции на игровой процесс: перемещение шашек, анимация их движения, подсветка выделенной клетки.

## **Требования к программному обеспечению**

* Операционная система Windows 7 или выше.

## **Требования к интерфейсу**

* Интерфейс должен быть прост в освоении для любого пользователя.
* Взаимодействие с пользователем через нажатия клавиш мыши.

# **Схема функционального взаимодействия блоков**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 9 |
| Изм. | Лист | № докум | Подпись | Дата |

Все функции, реализованные в проекте, можно разбить на следующие блоки:

1. В блоке поиска лучшего хода можно выделить функции:
   1. Функция создания весов каждой клетки матрицы поля для дальнейшей обработки.
   2. Функция MinMax для получения результата лучшего хода.
   3. Функция построения дерева всевозможных ходов для искусственного интеллекта.

Данный блок взаимодействует с блоком игрового взаимодействия.

1. В блоке изменения интерфейса содержится функция:
   1. Изменение цветов шашек каждой из команд, изменение количества ходов партии, изменение цвета заднего фона (доски).
   2. Функция взаимодействует с функциями «Сохранение цветов шашек, доски, сохранение информации проведенной игры в файл» и «Импорт данных о цвете шашек, цвета доски, импорт списка перемещений из файла» из блоков экспорта и импорта.
2. В блоке файлы содержатся функции:
   1. Создание файла.
   2. Чтение из файла

Блок взаимодействует с блоками импорта и экспорта.

1. В блоке игрового взаимодействия выделяются следующие функции:
   1. Анимация перемещения шашек по полю.
   2. Обработка событий нажатия и отжатия клавиши мыши для дальнейших перемещений шашек.
   3. Проверка окончания игры, с дальнейшим определением победителя.
   4. Проверка количества совершенных ходов с дальнейшей проверкой на окончание игры.

Блок взаимодействует с блоком поиска лучшего хода.

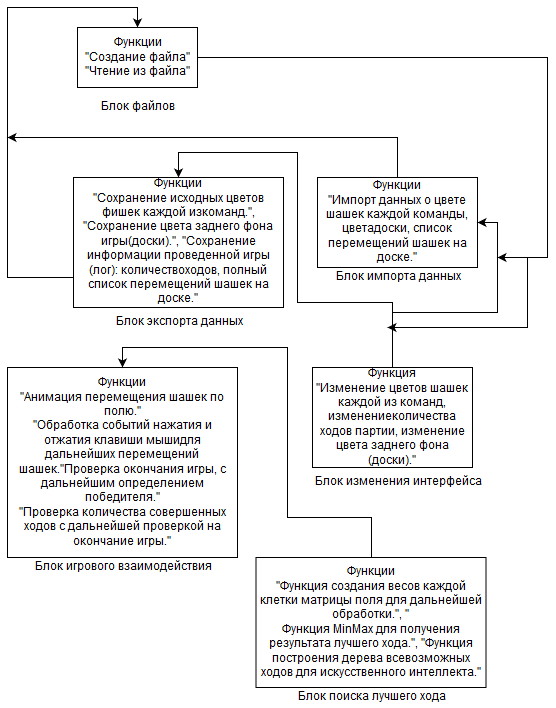
1. В блоке экспорта содержатся следующие функции:
   1. Сохранение исходных цветов фишек каждой из команд.
   2. Сохранение цвета заднего фона игры (доски).
   3. Сохранение информации проведенной игры (лог): количество ходов, полный список перемещений шашек на доске.

Данный блок берет информацию из блока файлов.

1. В блоке импорта выделяется одна функция:
   1. Импорт данных о цвете шашек каждой команды, цвета доски, импорт списка перемещений шашек на доске.

Данная функция берет информацию из блока файлов.

Схема функциональных блоков представлена в приложении А.



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 10 |
| Изм. | Лист | № докум | Подпись | Дата |

Рисунок 1 - Схема взаимодействия функциональных блоков

# **Математический аппарат**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 11 |
| Изм. | Лист | № докум | Подпись | Дата |

В качестве математического аппарата был выбран конечный автомат. Схема конечного автомата представлена в приложении Б.

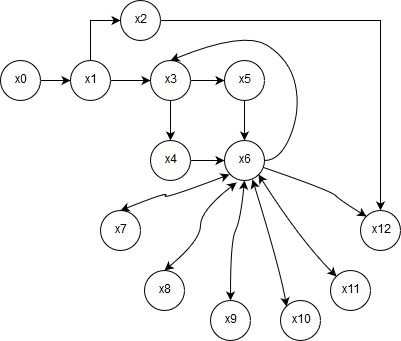


Рисунок 2 - Схема конечного автомата

Зададим состояния автомата множеством внутренних состояний:

X = {x0, x1, x2, x3, x4, x5, x6, x7, x8, x9, x10, x11, x12};

* x0 – начальное состояние, когда приложение еще запускается. Переходит в x1.
* x1 – состояние, когда приложение запустилось (основное окно). Переходит в x3.
* x2 – состояние, когда пользователь нажал “Exit”, переходит в x12.
* x3 – состояние, когда пользователь перешёл в меню выбора режима игры. Переходит в x4 и x5.
* x4 – состояние, когда пользователь перешёл в режим игры против компьютера. Переходит в x6.
* x5 – состояние, когда пользователь перешёл в режим игры против реального игрока. Переходит x6.
* x6 – состояние игрового процесса. Переходит в состояния x3, x7, x8, x9, x10, x11, x12.
* x7 – состояние сохранения настроек. Переходит в состояние x6.
* x8 –состояние открытия и загрузки настроек. Переходит в x6.
* x9 – нажата пауза в игре. Переходит в x6.
* x10 – состояние изменение цвета шашек, переходит в x6.
* x11 – состояние изменения цвета доски, переходит в x6

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 12 |
| Изм. | Лист | № докум | Подпись | Дата |

* x12 – состояние, когда пользователь выходит из приложения.

# **Диаграммы классов**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | *ТПЖА.09.03.01.014 ПЗ* | Лист |
|  |  |  |  |  | 13 |
| Изм. | Лист | № докум | Подпись | Дата |



# **Разработка алгоритмов**

Функция «Поиск лучшего хода компьютера» требует разработки алгоритма, что в дальнейшем упростит написание кода и разработку в целом. Схема алгоритма данной функции рассмотрена в приложении В.

## **Функция «Поиск лучшего хода компьютера»**

Данная функция заключается в поиске наилучшего хода компьютера, который и будет осуществлен. Весь алгоритм состоит из данных частей: эвристическая оценка веса каждой из клеток поля для дальнейшей проверки ходов, построение дерева всевозможных ходов с заданной глубиной для компьютера, проверка наилучшего хода из одной клетки в другую. Первая часть необходима для того, чтобы была информация о весе каждой из клеток. Так как ход на клетку с большим весом имеет и больший процент выигрыша компьютера. Вторая часть состоит из построения дерева из всевозможных ходов, которые и проверяются в дальнейшем на наилучший ход. Создается дерево, вершинами которого является положение всех шашек на доске, а значение каждой вершины равно расстоянию от данной клетки до противоположного угла. Чем больше расстояние – тем меньше и шанс победы. В третьей части описывается алгоритм для проверки наилучшего хода. Рекурсивно на каждом шаге вызывается функция лучшего хода, которая получает значение расстояния от заданной клетки до противоположного угла, которое проверяется с предыдущим. Вычисляется минимальное расстояние при глубине, равной 3. В дальнейшем осуществляется ход именно в ту клетку, расстояние до которой имеет минимальное значение. Первоначально для компьютера значения, где находятся фишки равно максимальному, и не допустимому с той целью, чтобы не рассматривать данные ветки вариаций исхода игры. Веса же каждой из клеток первоначально тоже заданы, и содержатся в вспомогательном файле, из которого и будут браться.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 14 |
| Изм. | Лист | № докум | Подпись | Дата |

# **Разработка структуры интерфейса пользователя**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 15 |
| Изм. | Лист | № докум | Подпись | Дата |

Вся работа с проектом осуществляется посредством нажатия клавиш мыши. При первом открытии приложения на форме находятся две кнопки: Start и Exit, каждая из которых выполняет свои соответствующие функции. Для придания большей красоты, а также для минимизации графической части, было принято убрать стандартную верхнюю панель диалоговых окон, предоставляемую Windows. После того, как пользователь перейдет в меню к старту меню, перед ним появится возможность выбрать один из двух предложенных режимов игры: игры против второго реального игрока, либо игры против компьютера. Как в первом, так и во втором случае, исходит будет один: пользователь будет перенаправлен в главное окно, отвечающее за игровой процесс. Большую часть графического пространства занимает графическая сцена с отображаемыми на ней шашками каждой команды. Немного правее находятся три вспомогательных кнопки для воздействия на игру: Start/Pause, Stop, Exit. В верхней части приложения находятся классические вкладки, в каждой из которых содержится свои ключевые функции приложения. В первой вкладке File находятся вкладки Save и Open, которые нужны для сохранения, либо импорта главных свойств приложения: цвета шашек каждой из команд, цвета доски, полный список проведенной игры. Во второй вкладке Settings находится визуальная настройка цветов шашек и доски, где пользователь может выбрать необходимый цвет, для дальнейшей игры. В третьей вкладке More app находится вспомогательная информация по самой игре, а также информация о самом проекте.

# **Разработка модульной структуры проекта**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 16 |
| Изм. | Лист | № докум | Подпись | Дата |

Данный проект был разбит на модульные структуры, так как это улучшит читаемость кода, ускорит отладку приложения и облегчит поддержание и изменение всего проекта в дальнейшем.

Краткое описание модулей представлено в таблице 2.

Таблица 2 – содержание модулей проекта

|  |  |
| --- | --- |
| Cloud.h | Содержит предварительное описание функций для анимации облаков |
| Cloud.cpp | Содержит реализацию всех методов и функций для анимации облаков |
| Constants.h | Содержит вспомогательные данные для работы внутренней логики игры |
| Custombutton.h | Содержит предварительное описание функций для кастомизированных кнопок |
| Custombutton.cpp | Содержит реализацию всех методов и функций для кнопок |
| Customedit.h | Содержит предварительное описание функций для кастомизированных полей ввода |
| Customedit.cpp | Содержит реализацию всех методов и функций для полей ввода |
| Customlabel.h | Содержит предварительное описание функций для кастомизированных меток |
| Customlabel.cpp | Содержит реализацию всех методов и функций для меток |
| Game.h | Содержит предварительное описание функций для игровой логики |
| Game.cpp | Содержит реализацию всех методов и функций для игровой логики |
| Mainwindow.h | Содержит предварительное описание функций для отображения меню игры |
| Mainwindow.cpp | Содержит реализацию всех методов и функций для отображения меню игры |
| View.h | Содержит предварительное описание функций для визуализации игрового процесса |
| View.cpp | Содержит реализацию всех методов и функций для визуализации игрового процесса |
| Main.cpp | Содержит основные команды для запуска приложения |
| Mainwindow.ui | Содержит описание главной форме, на которой размещены все необходимые виджеты |

# **Заключение**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 17 |
| Изм. | Лист | № докум | Подпись | Дата |

По завершении всех этапов разработки проекта можно сделать следующие выводы:

* Данный продукт полностью соответствует всем требованиям, указанным в постановке задач;
* Разработанное приложение отличается от предложенных аналогов в своей области тем, что имеет возможность игры как с реальным игроком, так и с компьютером, имеет интуитивно-понятный интерфейс, простое управление и перемещение во время игрового процесса, имеет доступ в игре без сети, не нуждается в высоких характеристиках персонального компьютера, а также имеет полностью открытый код, позволяющий редактировать, использовать и распространять каждый человек;
* Приложение выполняет все функции и реагирует на всевозможные команды.

# **Приложение А**

(обязательное)  
**Схема взаимодействия функциональных блоков**

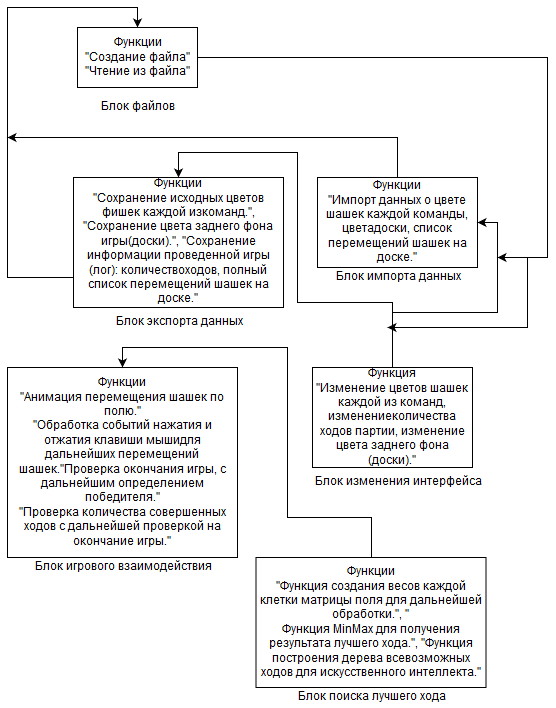
****

Рисунок 3 - Схема взаимодействия функциональных блоков

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 18 |
| Изм. | Лист | № докум | Подпись | Дата |

# **Приложение Б**

(обязательное)  
**Математический аппарат**

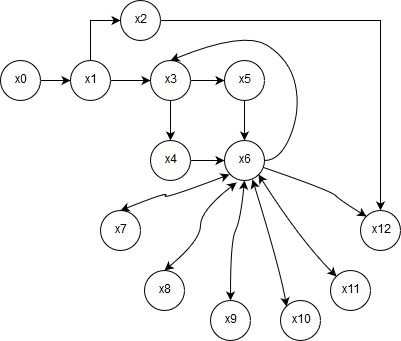
****

Рисунок 4 - Схема конечного автомата

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.110 ПЗ | Лист |
|  |  |  |  |  | 19 |
| Изм. | Лист | № докум | Подпись | Дата |

# **Приложение В**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 20 |
| Изм. | Лист | № докум | Подпись | Дата |

(обязательное)  
**Схемы алгоритмов**

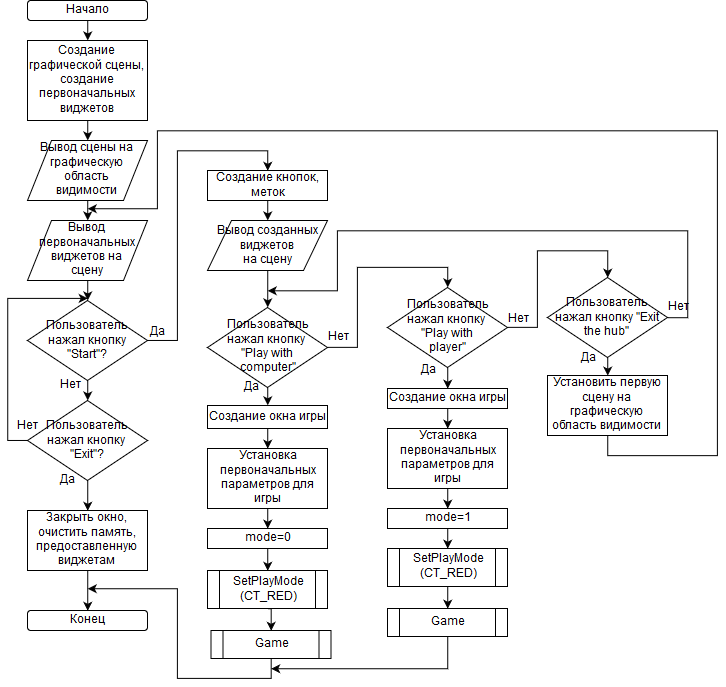
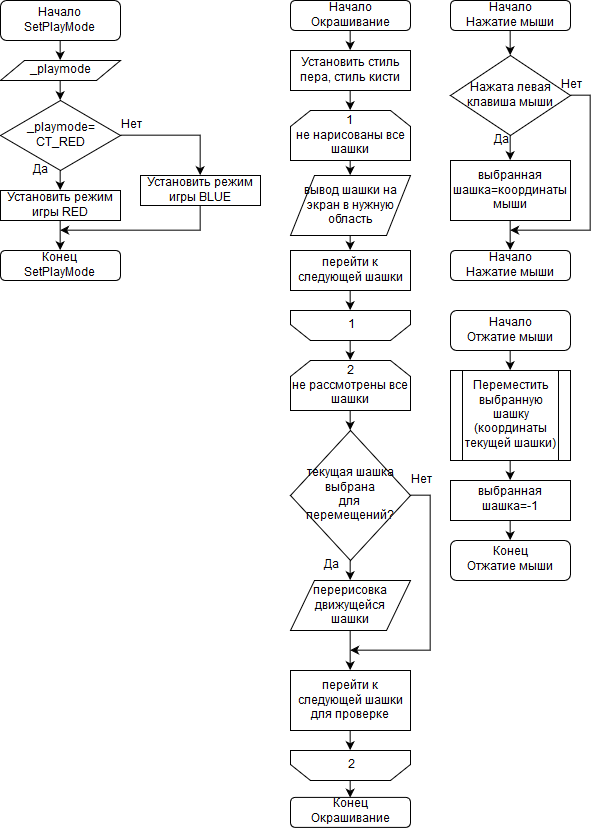
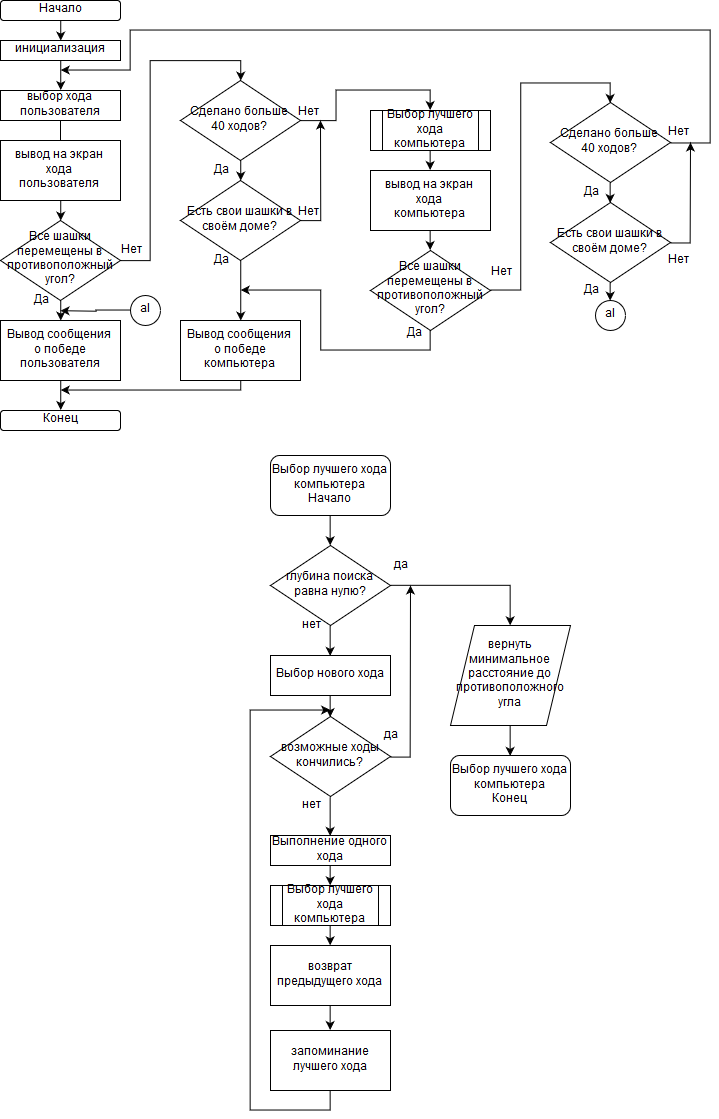


Рисунок 5 – Схема алгоритма начального меню выбора



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 21 |
| Изм. | Лист | № докум | Подпись | Дата |

Рисунок 6 – Схема алгоритмов логики игрового процесса



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.110 ПЗ | Лист |
|  |  |  |  |  | 22 |
| Изм. | Лист | № докум | Подпись | Дата |

Рисунок 7 – Схема алгоритма поиска лучшего хода

# **Приложение Е**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.110 ПЗ | Лист |
|  |  |  |  |  | 23 |
| Изм. | Лист | № докум | Подпись | Дата |

(обязательное)

**Экранные формы**

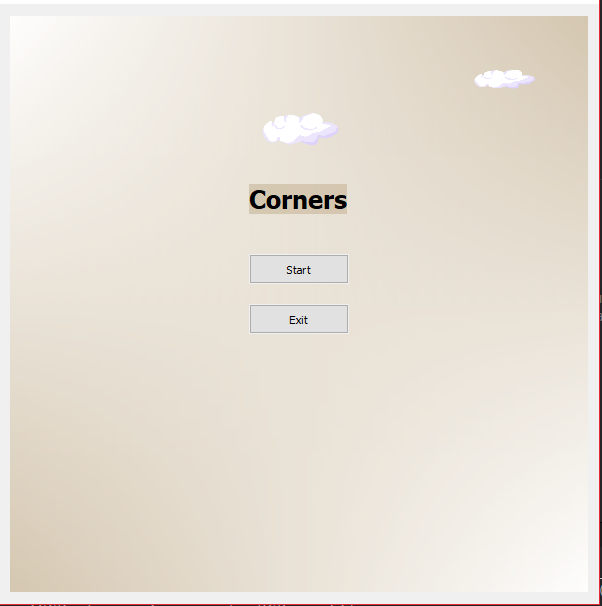


Рисунок 8 – Окно при старте программы

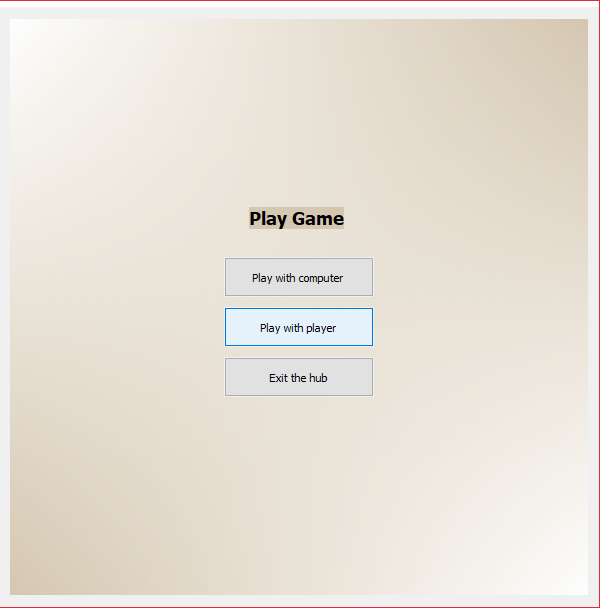


Рисунок 9 – Окно выбора режима игры

Рисунок 29 – Вкладка «Рецепты и калькуляторы» основного меню

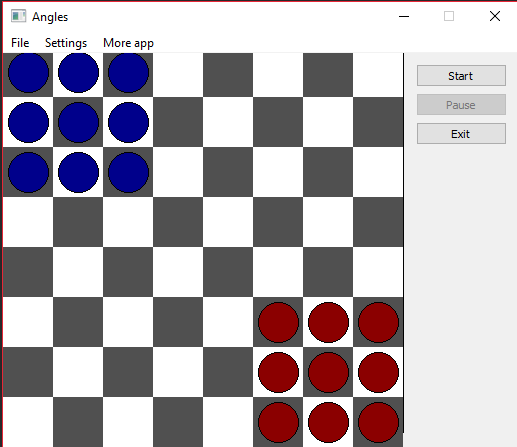


Рисунок 10 – Окно главного игрового процесса

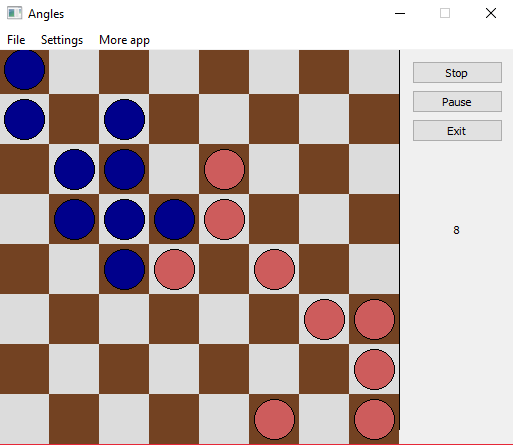


Рисунок 11 – Окно с активной игрой

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.110 ПЗ | Лист |
|  |  |  |  |  | 24 |
| Изм. | Лист | № докум | Подпись | Дата |

23

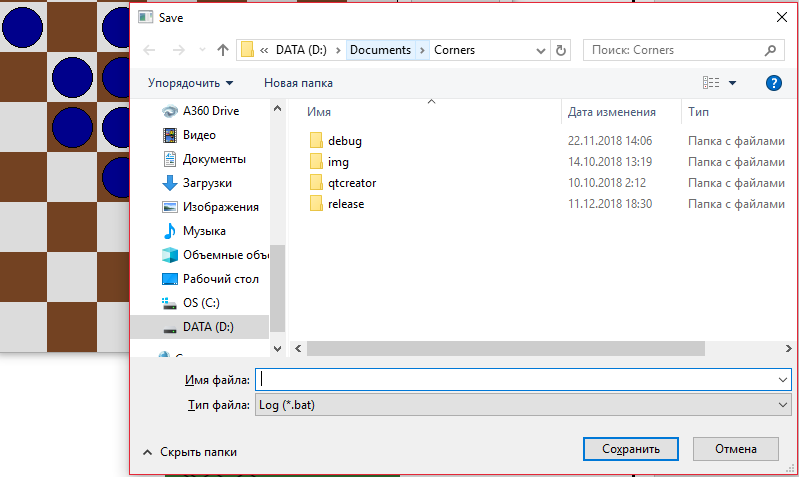


Рисунок 12 – Окно с сохранением информации об игре

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.110 ПЗ | Лист |
|  |  |  |  |  |  |
| Изм. | Лист | № докум | Подпись | Дата |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.110 ПЗ | Лист |
|  |  |  |  |  | 25 |
| Изм. | Лист | № докум | Подпись | Дата |

**Приложение Ж**

Рисунок 37 – Раздел «Информация о витамине, минерале или рецепте»

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 26 |
| Изм. | Лист | № докум | Подпись | Дата |

Рисунок 39 – Раздел «Рассчитать состав блюда или продукта»

(обязательное)

**Листинг основных функций программного продукта**

**”Cloud.h”**

#ifndef CLOUD\_H

#define CLOUD\_H

#include <QObject>

#include <QtWidgets>

class Cloud : public QObject, public QGraphicsItem

{

Q\_OBJECT

public:

Cloud(int w, int h);

Cloud(int w, int h, int sp);

QRectF boundingRect() const;

void paint(QPainter \*painter, const QStyleOptionGraphicsItem \*option, QWidget \*widget);

void setSpeed(qreal speed) { dx = speed; }

private:

int width;

int height;

qreal dx;

QPixmap PixMap;

QTimer time;

private slots:

void move();

};

#endif // CLOUD\_H

**”Cloud.cpp”**

#include "cloud.h"

Cloud::Cloud(int w, int h) : height(h), width(w) {

}

Cloud::Cloud(int w, int h, int sp) : width(w), height(h), dx(sp){

PixMap.load(":/img/img/Cloud.png");

PixMap = PixMap.scaled(width, height);

connect(&time, &QTimer::timeout, this, &Cloud::move);

time.start(50);

}

QRectF Cloud::boundingRect() const {

return QRectF(0, 0, width, height);

}

void Cloud::paint(QPainter \*painter, const QStyleOptionGraphicsItem \*option, QWidget \*widget) {

painter->drawPixmap(0, 0, PixMap, 0, 0, width, height);

}

void Cloud::move() {

if (dx > 0 && this->pos().x() + dx > 600)

setPos(-100, this->pos().y());

if (dx < 0 && this->pos().x() + dx < -100)

setPos(650, this->pos().y());

else

setPos(this->pos().x() + dx, this->pos().y());

}

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 27 |
| Изм. | Лист | № докум | Подпись | Дата |

**”Constants.h”**

#ifndef CONSTANTS\_H

#define CONSTANTS\_H

int StTab [8][8] = {

1, 2, 3, 4, 5, 6, 7, 8,

9, 10, 11, 12, 13, 14, 15, 16,

10, 17, 18, 19, 20, 21, 22, 23,

11, 18, 24, 25, 26, 27, 28, 29,

12, 19, 25, 30, 31, 32, 33, 34,

13, 20, 26, 31, 32, 402, 403, 404,

14, 21, 27, 32, 33, 403, 405, 406,

15, 22, 28, 33, 34, 405, 407, 408};

#endif // CONSTANTS\_H

**”Custombutton.h”**

#ifndef CUSTOMBUTTON\_H

#define CUSTOMBUTTON\_H

#include <QtWidgets>

class CustomButton : public QObject, public QGraphicsItem

{

Q\_OBJECT

public:

CustomButton();

CustomButton(int w, int h, int wx, int wh);

void setImage(QString);

void setText(QString);

QRectF boundingRect() const;

void paint(QPainter \*painter, const QStyleOptionGraphicsItem \*option, QWidget \*widget);

private:

int width;

int height;

int window\_x;

int window\_height;

QPixmap PixmMap;

protected:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 28 |
| Изм. | Лист | № докум | Подпись | Дата |

void mousePressEvent(QGraphicsSceneMouseEvent \*event);

signals:

void onPressButton();

};

#endif // CUSTOMBUTTON\_H

**”Custombutton.cpp”**

#include "custombutton.h"

CustomButton::CustomButton() {

}

CustomButton::CustomButton(int w, int h, int wx, int wh) : width(w), height(h) {

window\_x = wx;

window\_height = wh;

PixmMap.load(":/img/img/button.png");

PixmMap = PixmMap.scaled(width, height);

setPos(wx, wh);

}

void CustomButton::setImage(QString s) {

PixmMap.load(s);

PixmMap = PixmMap.scaled(width, height);

}

void CustomButton::setText(QString s) {

QPainter \*p = new QPainter();

p->drawText(window\_x, window\_height, s);

}

QRectF CustomButton::boundingRect() const {

return QRectF(0, 0, width, height);

}

void CustomButton::paint(QPainter \*painter, const QStyleOptionGraphicsItem \*option, QWidget \*widget) {

painter->drawPixmap(0, 0, PixmMap, 0, 0, width, height);

}

void CustomButton::mousePressEvent(QGraphicsSceneMouseEvent \*event) {

emit onPressButton();

}

**”Customedit.h”**

#ifndef CUSTOMEDIT\_H

#define CUSTOMEDIT\_H

#include <QtWidgets>

class CustomEdit : public QLineEdit

{

public:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 29 |
| Изм. | Лист | № докум | Подпись | Дата |

CustomEdit();

CustomEdit(int w, int h, int f, int wx, int wh);

private:

int width;

int height;

};

#endif // CUSTOMEDIT\_H

**”Customedit.cpp”**

#include "customedit.h"

CustomEdit::CustomEdit()

{

}

CustomEdit::CustomEdit(int w, int h, int size, int wx, int wh) {

setFixedSize(w, h);

setMaxLength(10);

QFont f;

f.setPixelSize(size);

f.setBold(true);

setFont(f);

setStyleSheet("QLineEdit{background-color: rgba(195, 176, 145, 80);}");

move(wx, wh);

}

**”Customlabel.h”**

#ifndef CUSTOMLABEL\_H

#define CUSTOMLABEL\_H

#include <QLabel>

#include <QtWidgets>

class CustomLabel : public QLabel

{

Q\_OBJECT

public:

CustomLabel();

CustomLabel(int f, int wx, int wh);

signals:

void on\_press\_label();

protected:

void keyPressEvent(QKeyEvent \*event);

};

#endif // CUSTOMLABEL\_H

**”Customlabel.cpp”**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 30 |
| Изм. | Лист | № докум | Подпись | Дата |

#include "customlabel.h"

CustomLabel::CustomLabel()

{

}

CustomLabel::CustomLabel(int size, int wx, int wh) {

QFont f;

f.setPixelSize(size);

f.setBold(true);

setFont(f);

setStyleSheet("QLabel{background-color: rgba(195, 176, 145, 80);}");

move(wx, wh);

}

void CustomLabel::keyPressEvent(QKeyEvent \*event) {

emit on\_press\_label();

}

**”game.h”**

#ifndef GAME\_H

#define GAME\_H

#include <QtWidgets>

#define MIN\_VALUE 0

#define MAX\_VALUE 255

#define EPIC\_BIG\_VALUE 500

class Game

{

public:

Game();

enum ChekerType {

CT\_NO\_ONE = 0,

CT\_RED = 1,

CT\_BLUE = 2

};

void setStep(int \_step) { step = \_step; }

void setMode(int \_mode) { mode = \_mode; }

int getMode() { return mode; }

int getStep() { return step; }

void step\_plus() { step++; }

void setPlayMode(ChekerType \_gameMode) { gamemode = \_gameMode; }

ChekerType getPlayMode() { return gamemode; }

void setActive(bool active) { this->active = active; }

void setAILevel(int AILevel) { this->AILevel = AILevel; }

bool isActive() { return active; }

int getMonsterCount() { return 18; }

void resset(); //start, restart game

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 31 |
| Изм. | Лист | № докум | Подпись | Дата |

QPoint& getRabbitPosition(int chekerIndex) { return reds[chekerIndex]; }

QPoint& getWolfPosition(int chekerIndex)

{

Q\_ASSERT(chekerIndex < 10 && chekerIndex >= 0);

return blues[chekerIndex];

}

QPoint& getMonsterPosition(int chekerIndex)

{

if (chekerIndex < 9)

return getRabbitPosition(chekerIndex);

else

return getWolfPosition(chekerIndex - 9);

}

ChekerType getChekerType(int chekerIndex)

{

if (chekerIndex < 9)

return CT\_RED;

else

return CT\_BLUE;

}

int isChekerTurn() { return chekerTurn; }

int getChekerIndexPosition(const QPoint &pos);

int getSelectedChekerIndex() { return selectedCheker; }

void setSelectedChekerIndex(int chekerindex) { selectedCheker = chekerindex; }

bool canMoveToPosition(int chekerIndex, const QPoint &pos);

bool canBigMoveToPosition(int chekerIndex, const QPoint &pos);

bool moveSelectedChekerToPosition(const QPoint &pos);

bool checkRange(int x, int y) { return (x >= 0 && x <= 7 && y >= 0 && y <= 7); }

bool checkRange(const QPoint &point) {return checkRange(point.x(), point.y()); }

bool isGameOver(ChekerType& winner);

bool checkHomes(ChekerType cheker);

private:

static const int NOT\_INITIALIZED = 255;

static const int EMPTY = 0;

static const int RED = 1;

static const int BLUE = 255;

QPoint blues[9]; //blues cheker (3x3)

QPoint reds[9]; //red cheker (3x3)

ChekerType gamemode; // state radiobutton(chose color)

bool active; //state game

bool chekerTurn;

int AILevel;

int step;

int mode;

int selectedCheker;

int map[8][8]; // main array

QQueue <QPoint> searchWay;

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 32 |
| Изм. | Лист | № докум | Подпись | Дата |

QPoint possibleMoves[8];

bool canMove(int x, int y);

bool canMove(const QPoint &point) { return canMove(point.x(), point.y()); }

bool canMove(const QPoint &currpoint, int x, int y);

bool canMove(const QPoint &currpoint, const QPoint &point) { return canMove(currpoint, point.x(), point.y()); }

int getHeuristicEvulation(int chekerindex);

int runMinMax(ChekerType cheker, int recursiveLevel, int alpha, int beta);

void temporaryChekerMovement(int chekerindex, int x, int y);

void temporaryChekerMovement(int chekerindex, const QPoint &point) { temporaryChekerMovement(chekerindex, point.x(), point.y()); }

void prepareMap();

void initialize();

//signals:

// void isTurnCheker(QPoint prevPos, QPoint newPos);

};

#endif // GAME\_H

**”game.cpp”**

#include "game.h"

#include <constants.h>

#include "qmath.h"

Game::Game() : active(false), selectedCheker(-1)

{

possibleMoves[0] = QPoint(-1, 0);

possibleMoves[1] = QPoint(1, 0);

possibleMoves[2] = QPoint(0, 1);

possibleMoves[3] = QPoint(0, -1);

possibleMoves[4] = QPoint(-2, 0);

possibleMoves[5] = QPoint(2, 0);

possibleMoves[6] = QPoint(0, 2);

possibleMoves[7] = QPoint(0, -2);

initialize();

step = 0;

}

void Game::resset()

{

active = true;

initialize();

if (gamemode != CT\_RED) {

}

chekerTurn = true;

}

int Game::getChekerIndexPosition(const QPoint &pos)

{

for (int i = 0; i < 9; i++) if (reds[i] == pos) return i;

for (int i = 0; i < 9; i++) if (blues[i] == pos) return i + 9;

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 33 |
| Изм. | Лист | № докум | Подпись | Дата |

return -1;

}

void Game::initialize()

{

for (int i = 0; i < 3; i++)

for (int j = 0; j < 3; j++)

this->blues[i \* 3 + j] = QPoint(i, j);

for (int i = 0; i < 3; i++)

for (int j = 0; j < 3; j++)

this->reds[i \* 3 + j] = QPoint(i + 5, j + 5);

}

bool Game::canMoveToPosition(int chekerIndex, const QPoint &pos) {

Q\_ASSERT(chekerIndex >= 0 || chekerIndex <= 18);

if (!checkRange(pos)) return false;

QPoint oldPosition = getMonsterPosition(chekerIndex);

if (pos.x() != oldPosition.x() && pos.y() != oldPosition.y()) return false;

QPoint diff = oldPosition - pos;

QPoint sum = oldPosition + pos;

//for (int i = 0; i < )

if (abs(diff.x()) == 2 || abs(diff.y()) == 2)

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

if (getChekerIndexPosition(pos) == -1 &&

getChekerIndexPosition(QPoint(sum.x() / 2, sum.y() / 2))!= -1)

return true;

if (abs(diff.x()) > 1 || abs(diff.y()) > 1) return false;

if (oldPosition == pos) return false;

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

if (i != chekerIndex && getMonsterPosition(i) == pos)

return false;

return true;

}

bool Game::canBigMoveToPosition(int chekerIndex, const QPoint &pos) {

prepareMap();

searchWay.clear();

QPoint buff = getMonsterPosition(chekerIndex);

if (buff == pos) return false;

searchWay.enqueue(buff);

while (!searchWay.isEmpty()) {

QPoint currentPosition = searchWay.dequeue();

for (int i = 4; i < 8; i++)

if (canMove(currentPosition, currentPosition + possibleMoves[i])) {

QPoint newPosition = currentPosition + possibleMoves[i];

map[newPosition.y()][newPosition.x()] = map[currentPosition.y()][currentPosition.x()] + 1;

if (newPosition == pos) return true;

searchWay.enqueue(newPosition);

}

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 34 |
| Изм. | Лист | № докум | Подпись | Дата |

}

return false;

}

bool Game::moveSelectedChekerToPosition(const QPoint &pos) {

if (selectedCheker < 0) return false;

if (canMoveToPosition(selectedCheker, pos) ||

canBigMoveToPosition(selectedCheker, pos)

)

if (selectedCheker < 9) {

//emit isTurnCheker(reds[selectedCheker], pos);

reds[selectedCheker] = pos;

} else {

//emit isTurnCheker(blues[selectedCheker - 9], pos);

blues[selectedCheker - 9] = pos;

}

else

return false;

//if (isGameOver(CT\_RED)) return true;

if (mode == 0) {

chekerTurn = !chekerTurn;

prepareMap();

runMinMax(gamemode == CT\_RED ? CT\_BLUE : CT\_RED, AILevel, -EPIC\_BIG\_VALUE, EPIC\_BIG\_VALUE);

ChekerType winner;

if (isGameOver(winner))

QMessageBox::information(0, "Attention", winner == CT\_RED ? "You win" : "You lost");

chekerTurn = !chekerTurn;

}

/\*

QString s;

s.setNum(test);

QMessageBox::information(0, "1", s);

\*/

//isGameOver(CT\_RED);

step++;

return true;

}

bool Game::isGameOver(ChekerType &winner) {

winner = CT\_NO\_ONE;

if (getStep() > 40)

for (int i = 5; i < 8; i++)

for (int j = 5; j < 8; j++)

for (int k = 0; k < getMonsterCount() / 2; k++)

if (getRabbitPosition(k) == QPoint(i, j))

{

QMessageBox::information(0, "Attention", "You lose");

active = false;

return 0;

}

int count = 0;

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 35 |
| Изм. | Лист | № докум | Подпись | Дата |

for (int i = 0; i < 3; i++)

for (int j = 0; j < 3; j++)

for (int k = 0; k < getMonsterCount() / 2; k++)

if (getRabbitPosition(k) == QPoint(i, j))

count++;

if (count == 9) winner = CT\_RED;

count = 0;

for (int i = 5; i < 8; i++)

for (int j = 5; j < 8; j++)

for (int k = 0; k < getMonsterCount() / 2; k++)

if (getWolfPosition(k) == QPoint(i, j))

count++;

if (count == 9) winner = CT\_BLUE;

if (winner != CT\_NO\_ONE) {

active = false;

return true;

}

return false;

}

bool Game::checkHomes(Game::ChekerType cheker) {

/\*if (cheker == CT\_RED)

for (int i = 5; i < 7; i++)

for (int j = 5; j < 7; j++)

//if (getChekerIndexPosition(QPoint(i, j)) )

\*/

}

bool Game::canMove(int x, int y) {

if (!checkRange(x, y))

return false;

if (map[y][x] != 0)

return false;

return true;

}

bool Game::canMove(const QPoint &currpoint, int x, int y) {

if (!checkRange(x, y))

return false;

QPoint pos(x, y);

if (map[y][x] == 0 && (

map[((currpoint + pos) / 2).y()][((currpoint + pos) / 2).x()] == BLUE ||

map[((currpoint + pos) / 2).y()][((currpoint + pos) / 2).x()] == RED

)) return true;

return false;

}

int Game::getHeuristicEvulation(int chekerindex) {

int max = 0, count = 0;

/\*int minR = 0, minB = 0;

if (cheker == CT\_RED) {

searchWay.clear();

for (int k = 0; k < 9; k++) {

//if (reds[i].y() >= 0 && reds[i].x() >= 0 && reds[i].y() <= 2 && reds[i].x() <= 2) return 0;

searchWay.enqueue(reds[k]);

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 36 |
| Изм. | Лист | № докум | Подпись | Дата |

while (!searchWay.empty()) {

QPoint currentPosition = searchWay.dequeue();

for (int i = 0; i < 4; i++)

if (canMove(currentPosition, currentPosition + possibleMoves[i])) {

QPoint newPosition = currentPosition + possibleMoves[i];

map[newPosition.y()][newPosition.x()] = map[currentPosition.y()][currentPosition.x()] + 1;

if (minR < StTab[newPosition.y()][newPosition.x()] - StTab[currentPosition.y()][currentPosition.x()])

minR = StTab[newPosition.y()][newPosition.x()] - StTab[currentPosition.y()][currentPosition.x()];

searchWay.enqueue(newPosition);

}

}

}

} else {\*/

searchWay.clear();

searchWay.enqueue(blues[chekerindex - 9]);

while (!searchWay.empty()) {

QPoint currentPosition = searchWay.dequeue();

for (int i = 0; i < 8; i++)

if (canMove(currentPosition, currentPosition + possibleMoves[i]) && count < AILevel) {

QPoint newPosition = currentPosition + possibleMoves[i];

map[newPosition.y()][newPosition.x()] = map[currentPosition.y()][currentPosition.x()] + 1;

if (StTab[(newPosition).y()][(newPosition).x()] -

StTab[(currentPosition).y()][(currentPosition).x()] > max)

max = StTab[(newPosition).y()][(newPosition).x()] -

StTab[(currentPosition).y()][(currentPosition).x()];

++count;

searchWay.enqueue(newPosition);

}

}

return max;

}

int Game::runMinMax(Game::ChekerType cheker, int recursiveLevel, int alpha, int beta) {

//if (recursiveLevel == 0) return getHeuristicEvulation(cheker);

int test = 0;

prepareMap();

int bestMove = NOT\_INITIALIZED;

bool isRed = (cheker == CT\_RED);

for (int i = (isRed ? 0 : 72); i < (isRed ? 72 : 144); i++)

{

int curCheker = i / 8;

QPoint curChekerPos = curCheker < 9 ? reds[curCheker] : blues[curCheker - 9];

QPoint curMove = possibleMoves[i % 8];

if (canMoveToPosition(curCheker, curChekerPos + curMove)) {

if (StTab[(curChekerPos + curMove).y()][(curChekerPos + curMove).x()] -

StTab[(curChekerPos).y()][(curChekerPos).x()] > test)

{

test = StTab[(curChekerPos + curMove).y()][(curChekerPos + curMove).x()] -

StTab[(curChekerPos).y()][(curChekerPos).x()];

bestMove = i;

}

/\*temporaryChekerMovement(curCheker, curMove);

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 37 |
| Изм. | Лист | № докум | Подпись | Дата |

test = -runMinMax(isRed ? CT\_BLUE : CT\_RED, recursiveLevel - 1, -beta, -alpha);

QString s;

s.setNum(test);

QMessageBox::information(0, "1", s);

temporaryChekerMovement(curCheker, -curMove);

if (test > alpha) {

alpha = test;

bestMove = i;

}\*/

}

}

//if (recursiveLevel == AILevel && bestMove != NOT\_INITIALIZED) {

if (cheker == CT\_BLUE)

blues[bestMove / 8 - 9] += possibleMoves[bestMove % 8];

else

reds[bestMove / 8] += possibleMoves[bestMove % 8];

//}

return alpha;

}

void Game::temporaryChekerMovement(int chekerindex, int x, int y) {

if (chekerindex < 9) {

map[reds[chekerindex].y()][reds[chekerindex].x()] = EMPTY;

map[reds[chekerindex].y() + y][reds[chekerindex].x() + x] = RED;

reds[chekerindex] += QPoint(x, y);

} else {

map[blues[chekerindex - 9].y()][blues[chekerindex - 9].x()] = EMPTY;

map[blues[chekerindex - 9].y() + y][blues[chekerindex - 9].x() + x] = BLUE;

blues[chekerindex - 9] += QPoint(x, y);

}

}

void Game::prepareMap() {

/\*for (int i = 0; i < 8; i++)

for (int j = 0; j < 8; j++)

map[i][j] = EMPTY;\*/

for (int i = 0; i < 8; i++)

memset(map[i],0, 8 \* sizeof(int));

for (int i = 0; i < 9; i++) {

map[reds[i].y()][reds[i].x()] = RED;

map[blues[i].y()][blues[i].x()] = BLUE;

}

}

**”mainwindow.h”**

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

#include <QtWidgets>

#include <cloud.h>

#include <custombutton.h>

#include <customlabel.h>

#include <customedit.h>

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 38 |
| Изм. | Лист | № докум | Подпись | Дата |

#include <view.h>

#include <game.h>

namespace Ui {

class MainWindow;

}

class MainWindow : public QMainWindow

{

Q\_OBJECT

public:

explicit MainWindow(QWidget \*parent = 0);

~MainWindow();

private:

Ui::MainWindow \*ui;

QGraphicsScene \*scene;

QGraphicsScene \*second\_menu;

QGraphicsScene \*register\_scene;

Cloud \*cloudA;

Cloud \*cloudB;

QPushButton \*startButton;

QPushButton \*exitButton;

QPushButton \*AIButton;

QPushButton \*player\_Button;

QPushButton \*back\_Button;

CustomLabel \*login;

CustomLabel \*password;

CustomLabel \*nameGame;

CustomLabel \*playGame;

CustomLabel \*ExitGame;

CustomEdit \*login\_edit;

CustomEdit \*password\_edit;

View \*view;

Game \*game;

private:

void init\_items();

void init\_view();

private slots:

void on\_startGame();

void on\_AI\_menu();

void on\_player\_menu();

void on\_back\_menu();

void on\_register\_menu();

void on\_Show\_main\_menu();

protected:

void keyPressEvent(QKeyEvent \*event);

};

#endif // MAINWINDOW\_H

**”mainwindow.cpp”**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 39 |
| Изм. | Лист | № докум | Подпись | Дата |

#include "mainwindow.h"

#include "ui\_mainwindow.h"

MainWindow::MainWindow(QWidget \*parent) :

QMainWindow(parent),

ui(new Ui::MainWindow)

{

this->setWindowFlags(Qt::CustomizeWindowHint);

ui->setupUi(this);

init\_view();

init\_items();

game = new Game;

view = new View;

connect(startButton, SIGNAL(clicked(bool)), this, SLOT(on\_startGame()));

connect(exitButton, SIGNAL(clicked(bool)), this, SLOT(close()));

connect(view, SIGNAL(on\_Exit\_Game()), this, SLOT(on\_Show\_main\_menu()));

}

MainWindow::~MainWindow()

{

delete ui;

}

void MainWindow::init\_items() {

QFont font;

font.setPixelSize(15);

font.setBold(true);

nameGame = new CustomLabel(25, 250, 180);

nameGame->setText("Corners");

// login = new CustomLabel(15, 150, 250);

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 40 |
| Изм. | Лист | № докум | Подпись | Дата |

// login->setText("Login");

// password = new CustomLabel(15, 150, 280);

// password->setText("Password");

// login\_edit = new CustomEdit(100, 20, 15, 250, 250);

// password\_edit = new CustomEdit(100, 20, 15, 250, 280);

cloudA = new Cloud(100, 50, 3.0);

cloudA->setPos(150, 100);

cloudB = new Cloud(80, 30, -1.0);

cloudB->setPos(500, 60);

//startButton = new CustomButton(80, 30, 200, 350);

//startButton->setText("Hello");

//registerButton = new CustomButton(80, 30, 300, 350);

startButton = new QPushButton("Start");

startButton->setGeometry(250, 250, 100, 30);

exitButton = new QPushButton("Exit");

exitButton->setGeometry(250, 300, 100, 30);

scene->addItem(cloudA);

scene->addItem(cloudB);

scene->addWidget(startButton);

scene->addWidget(exitButton);

//scene->addItem(startButton);

//scene->addItem(registerButton);

// scene->addWidget(login);

// scene->addWidget(password);

// scene->addWidget(login\_edit);

// scene->addWidget(password\_edit);

scene->addWidget(nameGame);

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 41 |
| Изм. | Лист | № докум | Подпись | Дата |

}

void MainWindow::init\_view() {

scene = new QGraphicsScene(this);

ui->graphicsView->setRenderHint(QPainter::Antialiasing);

ui->graphicsView->setCacheMode(QGraphicsView::CacheNone);

ui->graphicsView->setVerticalScrollBarPolicy(Qt::ScrollBarAlwaysOff);

ui->graphicsView->setHorizontalScrollBarPolicy(Qt::ScrollBarAlwaysOff);

ui->graphicsView->setFrameStyle(0);

ui->graphicsView->setSceneRect(0, 0, 600, 600);

ui->graphicsView->setSizePolicy(QSizePolicy(QSizePolicy::Fixed, QSizePolicy::Fixed));

setFixedSize(600, 600);

setWindowTitle("Corners");

ui->graphicsView->setScene(scene);

QBrush br;

QImage pic;

pic.load(":/img/img/background\_1.png");

br.setTextureImage(pic);

ui->graphicsView->setBackgroundBrush(br);

//ui->graphicsView->setBackgroundBrush(QBrush(QColor(195, 176, 145)));

}

void MainWindow::on\_startGame() {

second\_menu = new QGraphicsScene(this);

ui->graphicsView->setScene(second\_menu);

playGame = new CustomLabel(18, 250, 200);

playGame->setText("Play Game");

AIButton = new QPushButton("Play with computer");

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 42 |
| Изм. | Лист | № докум | Подпись | Дата |

AIButton->setGeometry(225, 250, 150, 40);

player\_Button = new QPushButton("Play with player");

player\_Button->setGeometry(225, 300, 150, 40);

back\_Button = new QPushButton("Exit the hub");

back\_Button->setGeometry(225, 350, 150, 40);

connect(AIButton, SIGNAL(clicked(bool)), this, SLOT(on\_AI\_menu()));

connect(player\_Button, SIGNAL(clicked(bool)), this, SLOT(on\_player\_menu()));

connect(back\_Button, SIGNAL(clicked(bool)), this, SLOT(on\_back\_menu()));

second\_menu->addWidget(playGame);

second\_menu->addWidget(AIButton);

second\_menu->addWidget(player\_Button);

second\_menu->addWidget(back\_Button);

}

void MainWindow::on\_AI\_menu() {

view->setGame(game);

view->show();

game->setMode(0);

game->setPlayMode(Game::CT\_RED);

this->hide();

}

void MainWindow::on\_player\_menu() {

view->setGame(game);

view->show();

game->setMode(1);

game->setPlayMode(Game::CT\_RED);

this->hide();

}

void MainWindow::on\_back\_menu() {

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 43 |
| Изм. | Лист | № докум | Подпись | Дата |

ui->graphicsView->setScene(scene);

}

void MainWindow::on\_register\_menu() {

register\_scene = new QGraphicsScene(this);

}

void MainWindow::on\_Show\_main\_menu() {

this->show();

}

void MainWindow::keyPressEvent(QKeyEvent \*event) {

switch (event->key()) {

case Qt::Key\_Escape:

close();

break;

default:

break;

}

}

**”view.h”**

#ifndef VIEW\_H

#define VIEW\_H

#include <QtWidgets>

#include "game.h"

namespace Ui {

class View;

}

class View : public QMainWindow

{

Q\_OBJECT

public:

explicit View(QWidget \*parent = 0);

~View();

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 44 |
| Изм. | Лист | № докум | Подпись | Дата |

void setGame(Game\* game);

private:

Ui::View \*ui;

Game\* game;

QPoint mousePosition;

QPoint realMousePosition;

QColor backGroundcolorOne;

QColor backGroundcolorTwo;

QColor chekercolorOne;

QColor chekercolorTwo;

protected:

void paintEvent(QPaintEvent \*event);

private slots:

void on\_pbPlay\_clicked();

void on\_actionSave\_triggered();

void on\_actionOpen\_triggered();

void on\_action1\_triggered();

void on\_action2\_triggered();

void on\_action3\_triggered();

void on\_actionDefault\_triggered();

void on\_action1\_2\_triggered();

void on\_action2\_2\_triggered();

void on\_action3\_2\_triggered();

void on\_action1\_3\_triggered();

void on\_action2\_3\_triggered();

void on\_action3\_3\_triggered();

void on\_actionInfo\_triggered();

void on\_pbStop\_clicked();

void on\_pushButton\_clicked();

void on\_actionExit\_triggered();

void hideTextLabel();

void on\_actionApp\_triggered();

protected:

void mousePressEvent(QMouseEvent \*event);

void mouseMoveEvent(QMouseEvent \*event);

void mouseReleaseEvent(QMouseEvent \*event);

void closeEvent(QCloseEvent \*event);

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 45 |
| Изм. | Лист | № докум | Подпись | Дата |

signals:

void on\_Exit\_Game();

};

#endif // VIEW\_H

**”view.cpp”**

#include "view.h"

#include "ui\_view.h"

View::View(QWidget \*parent) :

QMainWindow(parent),

ui(new Ui::View)

{

ui->setupUi(this);

this->setWindowTitle("Angles");

QTimer\* updater = new QTimer(this);

updater->start(30);

connect(updater, SIGNAL(timeout()), SLOT(update()));

backGroundcolorOne = QColor(80, 80, 80);

backGroundcolorTwo = QColor(255, 255, 255);

chekercolorOne = QColor(0, 0, 139);

chekercolorTwo = QColor(139, 0, 0);

ui->pbStop->setEnabled(false);

}

void View::setGame(Game\* game)

{

this->game = game;

}

View::~View()

{

delete ui;

}

void View::paintEvent(QPaintEvent \*)

{

QPainter p(this);

p.setPen(Qt::NoPen);

//draw grid

p.setBrush(QBrush(backGroundcolorOne));

for (int i = 0; i < 8; i++)

for (int k = 0; k < 8; k++)

if ((i + k) % 2 == 0) p.drawRect(i \* 50, k \* 50 + 15, 50, 50);

p.setBrush(QBrush(backGroundcolorTwo));

for (int i = 0; i < 8; i++)

for (int k = 0; k < 8; k++)

if ((i + k) % 2 != 0) p.drawRect(i \* 50, k \* 50 + 15, 50, 50);

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 46 |
| Изм. | Лист | № докум | Подпись | Дата |

p.setPen(Qt::SolidLine);

p.drawLine(400, 0, 400, 400);

//draw chekers

for (int i = 0; i < this->game->getMonsterCount(); i++) {

if (i != game->getSelectedChekerIndex()) {

p.setBrush(QBrush(this->game->getChekerType(i) == Game::CT\_BLUE ? chekercolorOne : chekercolorTwo));

p.drawEllipse(this->game->getMonsterPosition(i) \* 50 + QPoint(25,25) + QPoint(0, 15), 20, 20);

}

}

//draw moveing chekers

if (game->getSelectedChekerIndex() >= 0) {

if (!(mousePosition.x() < 0 || mousePosition.y() < 0

|| mousePosition.x() > 7 || mousePosition.y() > 7))

if (game->canMoveToPosition(game->getSelectedChekerIndex(), mousePosition) ||

game->canBigMoveToPosition(game->getSelectedChekerIndex(), mousePosition))

{

p.setPen(QPen(QBrush(QColor(148, 51, 145)), 5));

p.setBrush(QBrush(QColor(mousePosition.x() + mousePosition.y() % 2 == 0 ? backGroundcolorOne : backGroundcolorTwo), Qt::NoBrush));

p.drawRect(QRect(mousePosition \* 50 + QPoint(0, 15), QSize(50, 50)));

}

}

if (game->getSelectedChekerIndex() >= 0) {

p.setPen(QPen(QBrush(QColor(255, 165, 0)), 3));

p.setBrush(QBrush(QColor(game->getChekerType(game->getSelectedChekerIndex()) == Game::CT\_BLUE ? chekercolorOne : chekercolorTwo)));

p.drawEllipse(realMousePosition, 20, 20);

}

}

void View::mousePressEvent(QMouseEvent \*event) {

if (event->button() != Qt::LeftButton) return;

if (!game->isActive() || !game->isChekerTurn()) return;

realMousePosition = event->pos();

int chekerIndex = game->getChekerIndexPosition((event->pos() - QPoint(25, 25)) / 50);

if (chekerIndex < 0) return;

if (game->getChekerType(chekerIndex) != game->getPlayMode()) return;

game->setSelectedChekerIndex(chekerIndex);

}

void View::mouseMoveEvent(QMouseEvent \*event) {

realMousePosition = event->pos();

mousePosition = (event->pos() - QPoint(25, 25)) / 50;

}

void View::mouseReleaseEvent(QMouseEvent \*event) {

if (event->button() != Qt::LeftButton) return;

if (!game->isActive()) return;

realMousePosition = event->pos();

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 47 |
| Изм. | Лист | № докум | Подпись | Дата |

if (!game->moveSelectedChekerToPosition((event->pos() - QPoint(25, 25)) / 50)) {

ui->label->setText("ERROR! Try move!");

QTimer \*t = new QTimer(this);

t->start(2000);

connect(t, SIGNAL(timeout()), this, SLOT(hideTextLabel()));

}

QString s1, s2;

s1.setNum(game->getStep());

ui->label\_2->setText(s1);

if (game->getStep() >= 37 && game->getStep() <= 40) {

s2.setNum(40 - game->getStep());

s1.clear();

s1.append("Left ");

s1.append(s2);

ui->label\_3->setText(s1);

}

game->setSelectedChekerIndex(-1);

if (game->getMode() == 1)

if (game->getPlayMode() == Game::CT\_RED)

game->setPlayMode(Game::CT\_BLUE);

else

game->setPlayMode(Game::CT\_RED);

Game::ChekerType winner;

if (game->isGameOver(winner))

QMessageBox::information(0, "Attention", winner == game->getPlayMode() ? "You win" : "You lost");

}

void View::closeEvent(QCloseEvent \*event) {

emit on\_Exit\_Game();

}

void View::on\_actionSave\_triggered() {

QFile file(QFileDialog::getSaveFileName(this, QString::fromUtf8("Save"),

QDir::currentPath(),

"Log (\*.bat)"));

if (file.open(QIODevice::WriteOnly)) {

QString s;

s = backGroundcolorOne.name();

file.write(s.toUtf8());

file.write("\n");

s = backGroundcolorTwo.name();

file.write(s.toUtf8());

file.write("\n");

s = chekercolorOne.name();

file.write(s.toUtf8());

file.write("\n");

s = chekercolorTwo.name();

file.write(s.toUtf8());

file.write("\n");

file.close();

}

}

void View::on\_actionOpen\_triggered() {

QFile file(QFileDialog::getOpenFileName(this, QString::fromUtf8("Open"),

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 48 |
| Изм. | Лист | № докум | Подпись | Дата |

QDir::currentPath(),

"Log (\*.bat)"));

if (file.open(QIODevice::ReadOnly)) {

QQueue <QColor> colors;

while (!file.atEnd()) {

QString str = file.readLine();

//QMessageBox::information(0, "1", str);

str.remove(str.length() - 1, 1);

QColor temp;

temp.setNamedColor(str);

colors.enqueue(temp);

}

backGroundcolorOne = colors.dequeue();

backGroundcolorTwo = colors.dequeue();

chekercolorOne = colors.dequeue();

chekercolorTwo = colors.dequeue();

file.close();

}

}

void View::on\_action1\_triggered() {

if (!game->isActive()) chekercolorTwo = QColor(220, 20, 60);

}

void View::on\_action2\_triggered() {

if (!game->isActive()) chekercolorTwo = QColor(205, 92, 92);

}

void View::on\_action3\_triggered() {

if (!game->isActive()) chekercolorTwo = QColor(178, 34, 34);

}

void View::on\_actionDefault\_triggered() {

if (!game->isActive()) {

backGroundcolorOne = Qt::darkGray;

backGroundcolorTwo = Qt::white;

chekercolorOne = QColor(0, 0, 139);

chekercolorTwo = QColor(139, 0, 0);

}

}

void View::on\_action1\_2\_triggered() {

if (!game->isActive()) chekercolorOne = QColor(102, 205, 170);

}

void View::on\_action2\_2\_triggered() {

if (!game->isActive()) chekercolorOne = QColor(0, 139, 139);

}

void View::on\_action3\_2\_triggered() {

if (!game->isActive()) chekercolorOne = QColor(0, 255, 255);

}

void View::on\_action1\_3\_triggered() {

if (!game->isActive()) {

backGroundcolorOne = QColor(236, 235, 189);

backGroundcolorTwo = QColor(62, 95, 138);

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 49 |
| Изм. | Лист | № докум | Подпись | Дата |

}

}

void View::on\_action2\_3\_triggered() {

if (!game->isActive()) {

backGroundcolorOne = QColor(115, 66, 34);

backGroundcolorTwo = QColor(220, 220, 220);

}

}

void View::on\_action3\_3\_triggered() {

if (!game->isActive()) {

backGroundcolorOne = QColor(24, 21, 19);

backGroundcolorTwo = QColor(250, 223, 173);

}

}

void View::on\_actionInfo\_triggered() {

QMessageBox::information(0, "Information", "Created app by Maxim Sedov");

}

void View::on\_pbStop\_clicked() {

if (ui->pbStop->text() == "Pause") {

ui->pbStop->setText("Resume");

game->setActive(false);

} else {

ui->pbStop->setText("Pause");

game->setActive(true);

}

}

void View::on\_pbPlay\_clicked() {

if (ui->pbPlay->text() == "Start") {

ui->pbPlay->setText("Stop");

game->setPlayMode(Game::CT\_RED);

game->resset();

ui->pbStop->setEnabled(true);

ui->label\_3->clear();

} else {

ui->pbPlay->setText("Start");

game->setActive(false);

}

}

void View::on\_pushButton\_clicked() {

game->resset();

game->setStep(0);

ui->label\_2->clear();

this->close();

}

void View::on\_actionExit\_triggered() {

this->close();

}

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 50 |
| Изм. | Лист | № докум | Подпись | Дата |

void View::hideTextLabel() {

ui->label->setText("");

}

void View::on\_actionApp\_triggered() {

QString s;

s.append("The game ends if one of the following conditions is met:\n");

s.append("One of the players moved all his checkers into the opponent's house. This player won the game.\n");

s.append("One of the players still has several of his checkers in his house and at the same time made more than 40 moves. This player loses the game.");

QMessageBox::information(0, "Info app", s);

}

**“main.cpp”**

#include "mainwindow.h"

#include <QApplication>

int main(int argc, char \*argv[])

{

QApplication a(argc, argv);

MainWindow w;

w.show();

return a.exec();

}

# **Приложение З**

(справочное)

**Список использованных источников и материалов**

* <http://www.min2win.ru/gm.php?id=7605>
* <https://freesoft.ru/android/ugolki_android>
* <http://www.softportal.com/software-1919-ugolki.html>
* <http://www.gambiter.ru/ugolki/item/100-ugolki-skachat.html>
* <https://habr.com/post/146088/>
* <https://habr.com/post/224653/>
* <https://ru.wikipedia.org/wiki/%D0%9C%D0%B8%D0%BD%D0%B8%D0%BC%D0%B0%D0%BA%D1%81>
* <https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D0%B8%D0%B3%D1%80>

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 51 |
| Изм. | Лист | № докум | Подпись | Дата |

# **Биографический список**

1. Qt 5.10. Профессиональное программирование на C++. / Макс Шлее. – БХВ-Петербург. – 2018. – 1072 с.
2. Qt 5.3. Профессиональное программирование на C++. / Макс Шлее. – БХВ-Петербург. – 2015. – 928 с.
3. Программирование шахмат и других логических игр. / Евгений Корнилов. – БХВ-Петербург. – 2005. – 267 с.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.014 ПЗ | Лист |
|  |  |  |  |  | 52 |
| Изм. | Лист | № докум | Подпись | Дата |