

УТВЕРЖДАЮ

Руководитель практики от Университета

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(должность, ФИО)*

МП \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(Подпись)*

Дата « \_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2023 год

**ОТЧЕТ**

**ПО УЧЕБНОЙ ПРАКТИКЕ**

ПМ.03 Ревьюирование программных продуктов

|  |  |
| --- | --- |
| Манылова Полина Максимовна | |
| *(Ф.И.О. обучающегося)* | |
| 09.02.07 Информационные системы и программирование | |
| *(специальность)* | |
|  | |
| Учебная группа | ИСПк-403-52-00 |
|  |  |
| Место прохождения практики | ФГБОУ ВО «Вятский государственный университет», |
| Колледж ВятГУ | *(наименование организации, структурного подразделения организации)* |
|  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Итоговая оценка: |  | | | | |
| Руководитель  практики от университета |  |  |  |  |  |
|  | *(дата)* |  | *(подпись)* |  | *(Ф.И.О.)* |

Киров, 2023 г.



ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ НА ПРАКТИКУ

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Ф.И.О. обучающегося | | | Манылова Полина Максимовна | | | | | | |
| Специальность | | | 09.02.07 Информационные системы и программирование | | | | | | |
| Учебная группа | | | ИСПк-403-52-00 | | | | | | |
| Вид практики | | | учебная практика | | | | | | |
| Сроки прохождения практики с | | | |  | по |  | | | |
| Место прохождения практики | | | | ФГБОУ ВО «Вятский государственный университет»,  Колледж ВятГУ | | | | | |
|  | | (наименование организации, структурного подразделения организации) | | | | | | |
| № | Виды работ, выполняемых обучающимися во время практики | | | | | | Объем работ (час) | Формируемые компетенции | | |
| 1 | Пройти инструктаж по ознакомлению с правилами внутреннего трудового распорядка, охраны труда, техники безопасности, противопожарной безопасности, санитарно-эпидемиологическими правилами и гигиеническими нормативами, а также вводный инструктаж и инструктаж на рабочем месте | | | | | | 2 | ОК-7 | | |
| 2 | Определение требований к программному продукту и его функциональных характеристик, поиск и анализ готовых технических решений | | | | | | 4 | ОК-1-4 | | |
| 3 | Разработка технической документации на программный продукта | | | | | | 10 | ОК-1, ОК-2, ОК-4, ПК-3.3 ОК-9-11, ПК-3.2, ПК-3.3 | | |
| 4 | Ревьюирование программного кода. Создание репозитория | | | | | | 4 | ОК-1, ОК-4, ОК-8, ОК-10, ПК-3.1, ПК-3.4 | | |
| 6 | Разработка эксплуатационной документации | | | | | | 4 | ОК-10, ПК-3.4, ПК-3.5 | | |
| 7 | Подготовка презентации для представления программного продукта потенциальному потребителю. | | | | | | 4 | ОК-5 | | |
| 8 | Подготовка к сдаче и сдача промежуточной аттестации | | | | | | 2 | ОК-6 | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Индивидуальное задание на практику разработано в соответствии с рабочей программой практики. | | | | | |
| Руководитель  практики от университета |  |  |  |  |  |
|  | (дата) |  | (подпись) |  | (Ф.И.О.) |

|  |  |
| --- | --- |
| С индивидуальным заданием ознакомлен(а) |  |
|  | (дата, подпись обучающегося) |

**ОТЗЫВ РУКОВОДИТЕЛЯ ПРАКТИКИ**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Ф.И.О. обучающегося | | Манылова Полина Максимовна | | | |
| Специальность | | 09.02.07 Информационные системы и программирование | | | |
| Учебная группа | | ИСПк-403-52-00 | | | |
| Вид практики | | учебная практика | | | |
| Сроки прохождения практики с | | | 15.09.2023 | по | 20.11.2023 |
| Место прохождения практики | | | ФГБОУ ВО «Вятский государственный университет», Колледж ВятГУ | | |
|  | (наименование организации, структурного подразделения организации) | | | | |

ВИДЫ И КАЧЕСТВО ВЫПОЛНЕНИЯ РАБОТ

|  |  |  |  |
| --- | --- | --- | --- |
| Вид работ | Критерий выполнения работ | | |
| Выполнены полностью самостоятельно | Выполнены с незначительной помощью наставника | Выполнены с помощью наставника |
| Определение требований к программному продукту и его функциональных характеристик, поиск и анализ готовых технических решений | V |  |  |
| Разработка технической документации на программный продукта | V |  |  |
| Ревьюирование программного кода. Создание репозитория | V |  |  |
| Разработка эксплуатационной документации |  |  |  |
| Подготовка презентации для представления программного продукта потенциальному потребителю. | V |  |  |
| Подготовка к сдаче и сдача промежуточной аттестации | V |  |  |

Обучающийся ознакомлен с правилами внутреннего трудового распорядка, охраны труда, техники безопасности, противопожарной безопасности, санитарно-эпидемиологическими правилами и гигиеническими нормативами, а также прошел вводный инструктаж и инструктаж на рабочем месте.

Во время прохождения учебной практики обучающимся освоены следующие профессиональные и общие компетенции:

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование компетенции | Показатели оценки | Оценка | |
| Освоена | Не освоена |
| ПК 3.1. Осуществлять ревьюирование программного кода в соответствии с технической документацией. | Способен анализировать программный код с целью выявления некачественных архитектурных решений и критических мест в программе | V |  |
| ПК 3.2. Выполнять процесс измерения характеристик компонент программного продукта для определения соответствия заданным критериям | Способен верифицировать компоненты программного обеспечения в соответствии с заданными критериями | V |  |
| ПК 3.3. Производить исследование созданного программного кода с использованием специализированных программных средств с целью выявления ошибок и отклонения от алгоритма. | Способен готовить тесты для осуществления автоматизированного выявления ошибок в разрабатываемом программном обеспечении | V |  |
| ПК 3.4. Проводить сравнительный анализ программных продуктов и средств разработки, с целью выявления наилучшего решения согласно критериям, определённым техническим заданиям. | Способен подбирать средства разработки ПО наиболее подходящие по критериям определенным в техническом задании. | V |  |
| ПК 3.5. Проводить исследование проектной документации программного модуля. | Способен разрабатывать техническую и эксплуатационную документацию на программное обеспечение | V |  |
| ОК 01. Выбирать способы решения задач профессиональной деятельности, применительно к различным контекстам. | Способен оценивать предметную область и выбирать оптимальные способы решения задач профессиональной деятельности | V |  |
| ОК 02. Осуществлять поиск, анализ и интерпретацию информации, необходимой для выполнения задач профессиональной деятельности. | Способен анализировать специализированную информацию и находить оптимальные пути решения задач профессиональной деятельности | V |  |
| ОК 03. Планировать и реализовывать собственное профессиональное и личностное развитие. | Способен находить пути улучшения имеющихся решений, позволяющих повысить их общий качественный уровень | V |  |
| ОК 04. Работать в коллективе и команде, эффективно взаимодействовать с коллегами, руководством, клиентами. | Способен грамотно формулировать запросы в целях получения разъясняющей информации | V |  |
| ОК 05. Осуществлять устную и письменную коммуникацию на государственном языке с учетом особенностей социального и культурного контекста. | Способен разрабатывать проектную, техническую и пользовательскую документации | V |  |
| ОК 06. Проявлять гражданско-патриотическую позицию, демонстрировать осознанное поведение на основе традиционных общечеловеческих ценностей. | Способен конструктивно обмениваться информацией с коллегами, демонстрировать осознанное поведение в ходе выполнения проектных работ | V |  |
| ОК 07. Содействовать сохранению окружающей среды, ресурсосбережению, эффективно действовать в чрезвычайных ситуациях. | Способен прогнозировать эффективность и ресурсозатратность используемых средств | V |  |
| ОК 08. Использовать средства физической культуры для сохранения и укрепления здоровья в процессе профессиональной деятельности и поддержания необходимого уровня физической подготовленности. | Способен соблюдать требования внутреннего трудового распорядка организации, охраны труда и техники безопасности в целях сохранения собственного здоровья | V |  |
| ОК 09. Использовать информационные технологии в профессиональной деятельности. | Способен применять современные инструменты создания ПО, в том числе для осуществления коллективной работы. | V |  |
| ОК 10. Пользоваться профессиональной документацией на государственном и иностранном языке. | Способен использовать в своей работе специализированную документацию | V |  |
| ОК 11. Планировать предпринимательскую деятельность в профессиональной сфере. | Способен разрабатывать презентации для представления программного продукта потенциальному потребителю. | V |  |

**Краткая характеристика работы обучающегося**

|  |
| --- |
| Программа практики выполнена в полном объеме. Все виды работ выполнялись в срок, |
| без существенных замечаний. |
|  |

|  |  |  |
| --- | --- | --- |
|  |  | Руководитель практики  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Подпись ФИО  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (должность)  Дата «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2023 год |

**СОДЕРЖАНИЕ**

[**ВВЕДЕНИЕ** 4](#_Toc150197160)

[**1. СВЕДЕНИЯ О РАБОТЕ, ВЫПОЛНЕННОЙ В ПЕРИОД ПРОХОЖДЕНИЯ УЧЕБНОЙ ПРАКТИКИ** 5](#_Toc150197161)

[**2. АНАЛИЗ ПОСТАВЛЕННОЙ ЗАДАЧИ** 6](#_Toc150197162)

[**3. РАЗРАБОТКА ТЕХНИЧЕСКОГО ЗАДАНИЯ** 8](#_Toc150197163)

[**3.1 Назначение разработки** 8](#_Toc150197164)

[**3.1.1 Функциональное назначение программы.** 8](#_Toc150197165)

[**3.1.2 Эксплуатационное назначение программы.** 8](#_Toc150197166)

[**3.2 Требования к программе или программному изделию:** 8](#_Toc150197167)

[**3.2.1 Требования к функциональным характеристикам** 8](#_Toc150197168)

[**3.2.2 Возможности пользователя** 8](#_Toc150197169)

[**3.1.3 Требования к надежности** 9](#_Toc150197170)

[**3.1.4 Условия эксплуатации** 9](#_Toc150197171)

[**3.1.5 Требования к составу и параметрам тех**нических средств для корректной работы программы необходимо: 9](#_Toc150197172)

[**3.1.6 Требования к информационной и программной совместимости** 9](#_Toc150197173)

[**3.1.7 Требования к маркировке и упаковке** 9](#_Toc150197174)

[**3.1.8 Требования к транспортированию и хранению** 10](#_Toc150197175)

[**3.1.9 Специальные требования** 10](#_Toc150197176)

[**3.1.10 Требования к программной документации** 10](#_Toc150197177)

[**4. Разработка Технического проекта** 11](#_Toc150197178)

[**4.1 Форма представления входных / выходных данных** 11](#_Toc150197179)

[**4.2 Определение семантики использования языков, библиотек и фреймворков** 11](#_Toc150197180)

[**5. ИСПОЛЬЗУЕМЫЙ ИНСТРУМЕНТАРИЙ И РАБОЧЕЕ ОКРУЖЕНИЕ** 12](#_Toc150197181)

[**ЗАКЛЮЧЕНИЕ** 14](#_Toc150197182)

[**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ** 15](#_Toc150197183)

[**ПРИЛОЖЕНИЕ** 16](#_Toc150197184)

[**АНАЛИТИЧЕСКАЯ ЗАПИСКА** 16](#_Toc150197185)

[**ТЕХНИЧЕСКОЕ ЗАДАНИЕ** 21](#_Toc150197186)

[**ТЕХНИЧЕСКИЙ ПРОЕКТ** 26](#_Toc150197187)

[**РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ** 33](#_Toc150197188)

[**РУКОВОДСТВО ПРОГРАММИСТА** 38](#_Toc150197189)

[**ПРОГРАММА И МЕТОДИКА ИСПЫТАНИЙ** 40](#_Toc150197190)

[**ИСХОДНЫЙ КОД** 51](#_Toc150197191)

# **ВВЕДЕНИЕ**

Учебная практика ПМ.07 проходила на базе Колледжа ФГБОУ ВО «Вятский государственный университет» в период с 15.09.2023 по 20.11.2023.

Цель практики: сформировать у обучающихся навыки разработки программного обеспечения, как законченного продукта с размещением артефактов на онлайн-хостинге.

Задачи практики:

– закрепить полученные в ходе освоения предшествующих дисциплин навыки и умения в области создания программных продуктов;

– закрепить навыки анализа кода с целью выявления неэффективных решений;

– закрепить навыки разработки технической и эксплуатационной документации.

# **1. СВЕДЕНИЯ О РАБОТЕ, ВЫПОЛНЕННОЙ В ПЕРИОД ПРОХОЖДЕНИЯ УЧЕБНОЙ ПРАКТИКИ**

В период 15.09.2022 по 20.11.2022 при прохождении учебной практики ПМ.06 на базе ФГОБУ ВО «Вятский государственный университет» был выполнен следующий перечень работ, представленный в таблице 1.

Таблица 1 – Сведения о работе, выполненной в период практики

| Дата | Краткое содержание выполненных работ |
| --- | --- |
| 15.09.2023 | Ознакомление с правилами внутреннего трудового распорядка организации, требованиями охраны труда и техники безопасности |
| 15.09.2023-29.09.2023 | Подготовка аналитической записки с указанием цели, назначения и функциональных характеристик разрабатываемого программного продукта |
| 29.09.2023-13.10.2023 | Подготовка технического проекта содержащего описание структуры и алгоритмических решений применяемых в программном продукте |
| 13.10.2023-27.10.2023 | Разработка эксплуатационной документации |
| 27.10.20223-10.11.2023 | Подготовка презентации программного продукта и окончательное формирование репозитория. |
| 20.11.2023 | Подготовка к сдаче и сдача промежуточной аттестации |

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата) (подпись)

# **2. АНАЛИЗ ПОСТАВЛЕННОЙ ЗАДАЧИ**

Целевая аудитория: эта игра предназначена для детей 3-12 лет для развития логических способностей. Также этой игрой могут пользоваться как взрослые, так и подростки для того, чтобы проводить свободное время.

Назначение проекта:

Игра предназначена для развития мелкой моторики, координации движений и внимания у игрока. Она помогает развить навыки рисования и улучшить письменные навыки, так как требует точности и контроля движений руки. Также игра способствует развитию логического мышления и пространственного восприятия, так как игрок должен выбирать оптимальный путь для обводки заданной фигуры без отрыва пера от бумаги.

Также программа предназначена для получения и повышения навыков разработки программного обеспечения и технической документации.

Основной функционал:

Цель игры «Обведи, не отрывая пера» заключается в том, что пользователь (игрок) должен обвести сложную замкнутую фигуру, не отрывая пера и не проходить по одним и тем же линиям дважды. Если игрок успешно обвел фигуру, то он может переходить на следующий уровень. Если игрок хочет перейти к следующему уровню, не проходя текущий уровень, то он может это сделать, нажав на стрелку, которая находится справа от игрового поля. Если же игрок не успешно обвел фигуру, то у него будет еще одна попытка, число попыток не ограничено. Игроку дано три уровня сложностей: легкий, средний и сложный. Каждый уровень сложности предполагает пять уровней сложности. Игра заканчивается, когда игрок прошел последний уровень.

Описание взаимодействия с потенциальным пользователем:

Пользователю должны быть доступны следующие возможности:

* если нажать на кнопку играть, пользователь будет проходить все уровни подряд, то есть с самого первого уровня до самого последнего уровня;
* перед началом игры пользователь может прочитать правила, нажав на главном экране на кнопку «об игре»;
* если нажать на кнопку выход, пользователь будет выходить из игры.

# **3. РАЗРАБОТКА ТЕХНИЧЕСКОГО ЗАДАНИЯ**

## **3.1 Назначение разработки**

## **3.1.1 Функциональное назначение программы.**

С помощью этой программы пользователь сможет интересно провести время, также развить логику и повысить свой навык владения компьютерной мышью. Также программа предназначена для получения и повышения навыков разработки программного обеспечения и технической документации.

## **3.1.2 Эксплуатационное назначение программы.**

Программа должна эксплуатироваться как приложение для персонального компьютера.

## **3.2 Требования к программе или программному изделию:**

## **3.2.1 Требования к функциональным характеристикам**

Цель игры «Обведи, не отрывая пера» заключается в том, что пользователь (игрок) должен обвести сложную замкнутую фигуру, не отрывая пера и не проходить по одним и тем же линиям дважды. Если игрок успешно обвел фигуру, то он может переходить на следующий уровень. Если игрок хочет перейти к следующему уровню, не проходя текущий уровень, то он может это сделать, нажав на стрелку, которая находится справа от игрового поля. Если же игрок не успешно обвел фигуру, то у него будет еще одна попытка, число попыток не ограничено. Игра заканчивается, когда игрок прошел последний уровень.

### **3.2.2 Возможности пользователя**

Пользователю должны быть доступны следующие возможности:

* если нажать на кнопку играть, пользователь будет проходить все уровни подряд, то есть с самого первого уровня до самого последнего уровня;
* перед началом игры пользователь может прочитать правила, нажав на главном экране на кнопку «об игре»;
* если нажать на кнопку выход, пользователь будет выходить из игры.

## **3.1.3 Требования к надежности**

Надежное (устойчивое) функционирование программы должно быть обеспечено выполнением Заказчиком совокупности организационно-технических мероприятий, перечень которых приведен ниже:

* организацией бесперебойного питания технических средств;
* использованием лицензионного программного обеспечения;
* регулярным выполнением рекомендаций Министерства труда и социального развития РФ, изложенных в Постановлении от 23 июля 1998 г. Об утверждении межотраслевых типовых норм времени на работы по сервисному обслуживанию ПЭВМ и оргтехники и сопровождению программных средств»;
* регулярным выполнением требований ГОСТ 51188-98. Защита информации. Испытания программных средств на наличие компьютерных вирусов.

## **3.1.4 Условия эксплуатации**

Условия эксплуатации программы должны соответствовать стандартным требованиям, предъявляемым к бытовым помещениям.

## **3.1.5 Требования к составу и параметрам тех**нических средств для корректной работы программы необходимо:

* процессор х86-64 не ниже 800 Мгц;
* ОЗУ не менее 256 Мб;
* VGA-совместимый видеоадаптер;
* не менее 100 Мб на жёстком диске;
* совместимая аудио-карта;
* HID-средства: клавиатура, мышь.

## **3.1.6 Требования к информационной и программной совместимости**

Программа должна быть написана на языке Python.

## **3.1.7 Требования к маркировке и упаковке**

Специальных требований нет.

## **3.1.8 Требования к транспортированию и хранению**

Специальные требования к транспортированию и хранению не предъявляются.

## **3.1.9 Специальные требования**

Простота интерфейса пользователя.

## **3.1.10 Требования к программной документации**

Состав программной документации должен включать:

* техническое задание;
* код программы;
* руководство пользователя.

# **4. Разработка Технического проекта**

## **4.1 Форма представления входных / выходных данных**

Особой формы представления входных данных не предусматривается, для игры предусмотрены экранные формы, которые представлены в техническом задании на разработку игры «Обведи, не отрывая пера».

## **4.2 Определение семантики использования языков, библиотек и фреймворков**

Язык Python - используется для написания основной логики игры, включая алгоритмы проверки существования вершин графа, проверки координат начальной и конечной точки вершин. Python выбран из-за его простоты, гибкости и широкой поддержки различных библиотек.

Использовались библиотеки и фреймворки, такие как: QMainWindow, QVBoxLayout, QWidget, QLabel, QT, buttons\_layout, QPixmap, PyQt5. Они служат для отрисовки окон и кнопок игры через пользовательский интерфейс (UI) - используются для создания графического интерфейса пользователя (GUI) игры. Они предоставляют различные компоненты, такие как окна, кнопки, текстовые поля и т. д., которые могут быть использованы для создания интерактивного пользовательского интерфейса игры. В данном случае, эти библиотеки или фреймворки используются для отображения игровых элементов и обработки пользовательских действий, связанных с ними.

Таким образом, семантика использования языков, библиотек и фреймворков в данном контексте связана с их ролями в разработке игры, включая написание логики игры на языке Python, создание графического интерфейса пользователя с помощью соответствующих библиотек или фреймворков, а также выполнение проверки существования вершины графа с использованием соответствующих алгоритмов или функций.

# **5. ИСПОЛЬЗУЕМЫЙ ИНСТРУМЕНТАРИЙ И РАБОЧЕЕ ОКРУЖЕНИЕ**

В ходе прохождения практики я использовала язык-программирования Python, решал задачи с помощью среды разработки PyCharm, его интерфейс показан на рисунке 1.

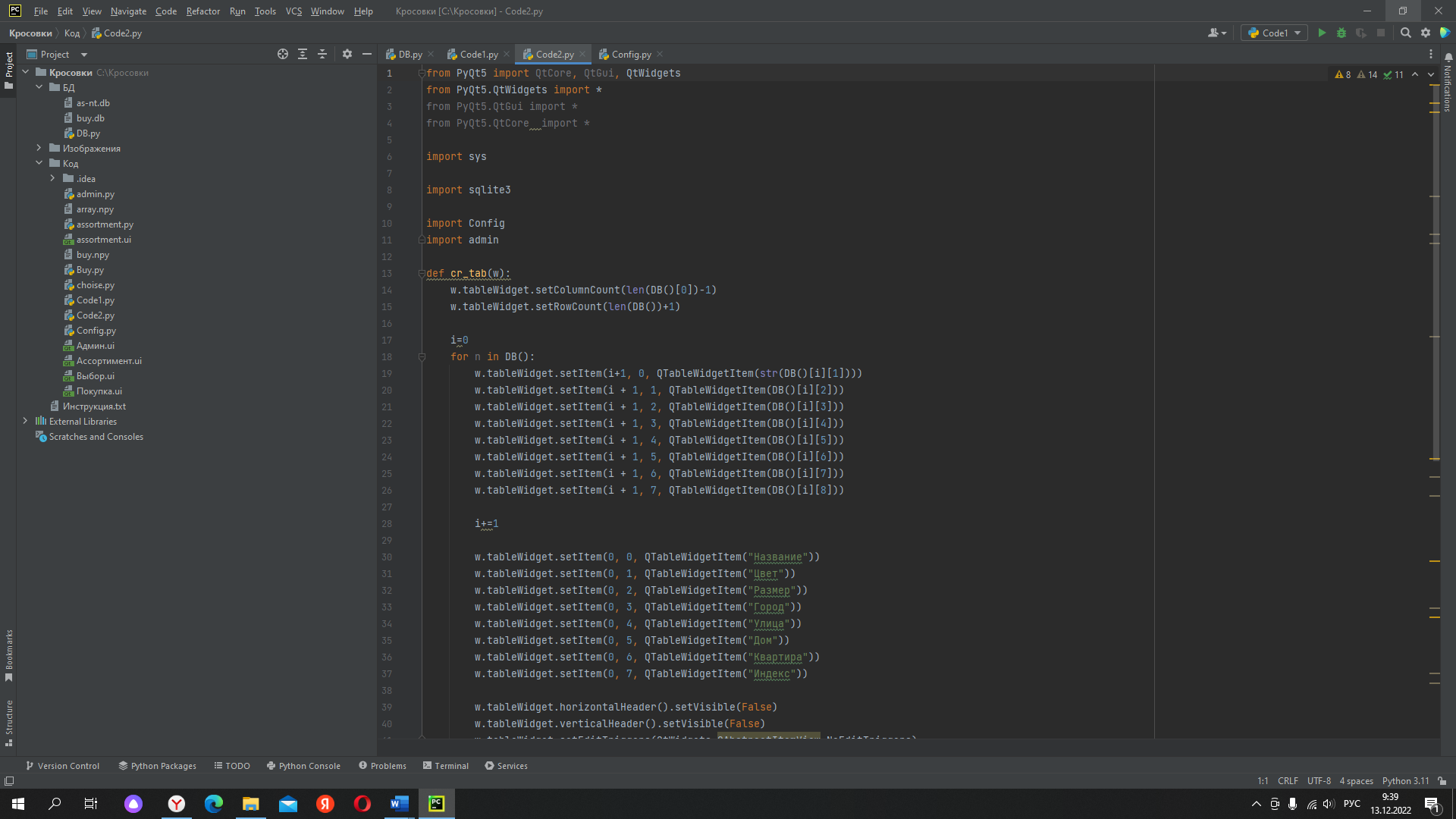


Рисунок 1 – интерфейс среды разработки PyCharm

Синтаксис языка-программирования Python всегда выделял его на фоне других языков программирования. Он не страдает избыточностью, схожесть синтаксиса с обычным английским позволяет понять код даже обычному пользователю, кроме того, программист пишет меньше строк кода, потому что нет необходимости использовать символы: «;», «{», «}». Вложенность обозначается отступами, что повышает читаемость кода и приучает c самого начала его использования правильному оформлению.

Также мне нравится, что у этого языка существует много областей, где могут его применить. Например, как в веб-разработке, в разработке десктопных приложений, мобильной разработке, также его используют для написания видеоигр.

Python – это универсальный, перспективный и относительно несложный в изучении язык. Идеальный выбор для тех, кто хочет попробовать IT-разработку, но еще не знает, какого рода продукты хочет создавать.

Для фиксирования всех артефактов и программных разработок использовался GitHub, все изменения и сохранения проводились в репозитории по ссылке: [https://github.com/polinamanylova/Praktika5/blob/main/код.docx](https://vk.com/away.php?to=https%3A%2F%2Fgithub.com%2Fpolinamanylova%2FPraktika5%2Fblob%2Fmain%2F%EA%EE%E4.docx&cc_key=)

# **ЗАКЛЮЧЕНИЕ**

В результате прохождения учебной практики был проведён анализ поставленных проблем с их дальнейшим решением. Благодаря этому был решён список поставленных задач.

Целью работы было ознакомление с процессом создания технического задания на разработку десктопного приложения, игры «Обведи, не отрывая пера».

В процессе выполнения работы был проведён обзор предметной области, написано техническое задание, технический проект, руководство пользователя и руководство программиста, также описаны решения и концепция проекта, разработаны схемы бизнес-процессов с их описанием, спроектирован прототип пользовательского интерфейса с описанием.

При разработке программного обеспечения были приобретены разнообразные навыки, необходимые для обеспечения качественной работы приложения. Также были получены дополнительный навыки написания технического задания на создание автоматизированного программного обеспечения с учётом особенностей предметной области.

Таким образом, разработанное техническое задание и программное обеспечение может значительно облегчить и ускорить качество работы игрокам, которые будут пользоваться этим приложением.

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. ГОСТ 34.602-89 Техническое задание на создание

автоматизированной системы (пример) – Текст электронный //

Корпоративные хранилища данных. Интеграция систем. Проектная

документация: [сайт] – 2010. – URL: https://www.prj-

exp.ru/patterns/pattern\_tech\_task.php#tech\_task2 (дата обращения: 29.09.2023).

1. Логическая модель ИС – Текст электронный // Студвуд: [сайт] –

2020. – URL: https://studwood.ru/1050371/informatika/logicheskaya\_model (дата

обращения: 18.10.2023).

1. DFD – диаграммы потоков данных – Текст электронный // Инфопедия: [сайт] – 2016. – URL: https://infopedia.su/12x891b.html (дата

обращения: 05.10.2023).

1. Проектирование USE CASE диаграммы – Текст электронный //

Программные решения для бизнеса: [сайт]. – 2020. – URL:

https://nationalteam.worldskills.ru/skills/proektirovanie-use-case-diagrammy-

opredelenie-funktsionalnykh-vozmozhnostey-sistemy/ (дата обращения:

30.10.2023).

1. QT. QT for Python – Текст электронный // QT for Python: [сайт]. – 2016. – URL: https://doc.qt.io/qtforpython-6/ (дата обращения: 18.09.2023).

# **ПРИЛОЖЕНИЕ**

## **АНАЛИТИЧЕСКАЯ ЗАПИСКА**

Целевая аудитория: эта игра предназначена для детей 3-12 лет для развития логических способностей. Также этой игрой могут пользоваться как взрослые, так и подростки для того, чтобы проводить свободное время.

Назначение проекта:

Игра предназначена для развития мелкой моторики, координации движений и внимания у игрока. Она помогает развить навыки рисования и улучшить письменные навыки, так как требует точности и контроля движений руки. Также игра способствует развитию логического мышления и пространственного восприятия, так как игрок должен выбирать оптимальный путь для обводки заданной фигуры без отрыва пера от бумаги.

Также программа предназначена для получения и повышения навыков разработки программного обеспечения и технической документации.

Основной функционал:

Цель игры «Обведи, не отрывая пера» заключается в том, что пользователь (игрок) должен обвести сложную замкнутую фигуру, не отрывая пера и не проходить по одним и тем же линиям дважды. Если игрок успешно обвел фигуру, то он может переходить на следующий уровень. Если игрок хочет перейти к следующему уровню, не проходя текущий уровень, то он может это сделать, нажав на стрелку, которая находится справа от игрового поля. Если же игрок не успешно обвел фигуру, то у него будет еще одна попытка, число попыток не ограничено. Игроку дано три уровня сложностей: легкий, средний и сложный. Каждый уровень сложности предполагает пять уровней сложности.

Игра заканчивается, когда игрок прошел последний уровень.

**Описание взаимодействия с потенциальным пользователем:**

Пользователю должны быть доступны следующие возможности:

* если нажать на кнопку играть, пользователь будет проходить все уровни подряд, то есть с самого первого уровня до самого последнего уровня;
* перед началом игры пользователь может прочитать правила, нажав на главном экране на кнопку «об игре»;
* если нажать на кнопку выход, пользователь будет выходить из игры.

Взаимодействие с потенциальным пользователем представлено на диаграмме Use Case на рисунке 1:

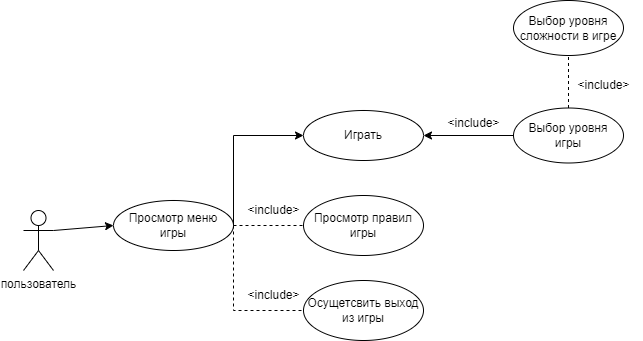


Рисунок 1 – Диаграмма Use-case

Возможные аналоги решения.

Подобных аналогов этой игры очень много, как и онлайн игр на сайтах, так и приложений. Они все универсальные, то есть, в эти игры можно играть, как на компьютерах и планшетах, так и на смартфонах. Самое скачиваемое приложение с аналогом этой игры в Play Маркете это приложение «Draw.ai» и «Рисование для детей. Рисовалка». Они все подобны друг другу, так как одна суть игры. Но по дизайну и подаче каждый разработчик сделал что-то свое.

Игра «Draw.ai»

Игра «Draw.ai» имеет большое количество скачиваний их больше 5 млн. Разработчиком этой игры является Happy coloring. Плюсы этого приложения в том, что оно занимает не очень много места 14 МБ, там не очень сложный интерфейс, я думаю с ним разберется каждый. В игре представлена галерея рисунков, которые можно обвести и их очень много, они разделены по категориям для более удобного поиска тематики картинки. Также там есть возможность начать учится рисовать поэтапно, это отлично подойдет для маленьких детей, которые только учатся рисовать.

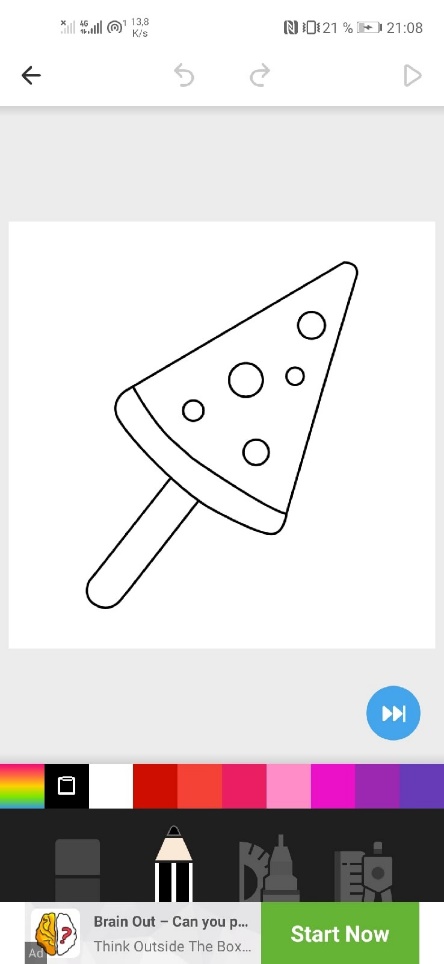


Рис 2. - Уровень из игры «Draw.ai».

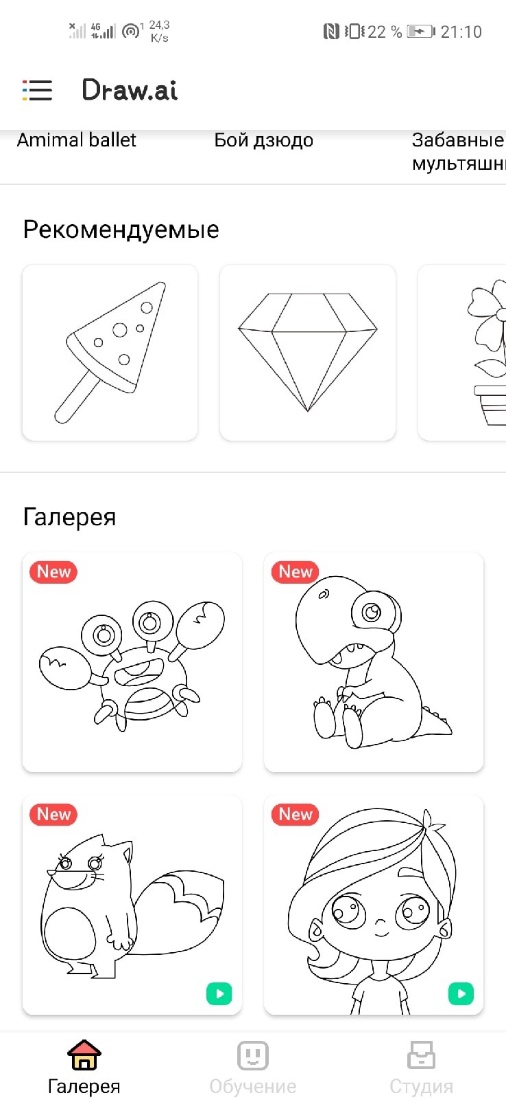


Рис 3. - Меню игры «Draw.ai».

Недостатками приложения является, что для большинства рисунков в игре требуется платная подписка. Также там есть ограничения по совместимости с устройствами. Поиграв в игру, никаких проблем не возникало. Все четко и понятно.

Игра «Рисование для детей. Рисовалка»

Игра «Рисование для детей. Рисовалка» имеет не мало количество скачиваний их более 550 тыс. Его разработчиком является GoKids! Плюсами является то, что приложение может работать без интернета, также занимает не много места на телефоне (31 МБ). Никаких ограничений по версиям устройства нет. Минус этой игры в том, что она только на английском языке.

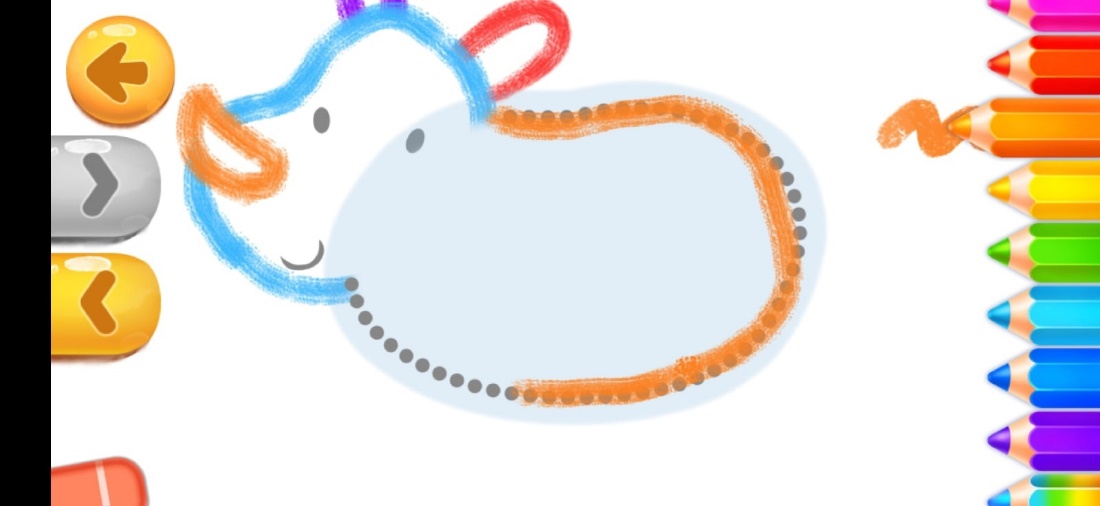


Рис 4. - Уровень из игры «1LINE».



Рис 5. - Меню из игры «1LINE».

Поиграв в эту игру, я могу сказать, что интерфейс в игре понятный, не очень сложный, даже без знания английского языка можно разобраться. Плюсом еще является то, что довольно много уровней и в них интересно будет играть ребенку, так как там после каждой нарисованной линии происходит анимация и издаются звуки, еще при выборе цвета пера (карандаша), озвучивается какой это цвет и тем самым ребёнку интересно играть и он может познать для себя что-то новое.

## **ТЕХНИЧЕСКОЕ ЗАДАНИЕ**

**1 Введение**

**1.1 Наименование программы**

Наименование программы - «Обведи, не отрывая пера».

**1.2 Краткая характеристика области применения программы**

Игра «Обведи, не отрывая пера» может быть использована в развлекательных целях, также в целях развития логики и базовыми навыками владения компьютерной мышью.

**2 Основания для разработки**

Основанием для разработки является учебный план специальности 09.02.07 «Информационные системы и программирование».

Сроки проведения работы:

начало разработки – 01.09.2023;

окончание разработки – 03.11.2023;

Заказчик: коллектив преподавателей Колледжа ВятГУ.

Исполнитель: студент колледжа ВятГУ группы ИСПк-403-52-00 Манылова. П. М.

**3 Назначение разработки**

**3.1 Функциональное назначение программы.**

С помощью этой программы пользователь сможет интересно провести время, также развить логику и повысить свой навык владения компьютерной мышью.

Также программа предназначена для получения и повышения навыков разработки программного обеспечения и технической документации.

**3.2 Эксплуатационное назначение программы.**

Программа должна эксплуатироваться как приложение для персонального компьютера.

**4 Требования к программе или программному изделию**

**4.1 Требования к функциональным характеристикам**

**4.1.1 Правила игры**

Цель игры «Обведи, не отрывая пера» заключается в том, что пользователь (игрок) должен обвести сложную замкнутую фигуру, не отрывая пера и не проходить по одним и тем же линиям дважды. Если игрок успешно обвел фигуру, то он может переходить на следующий уровень. Если игрок хочет перейти к следующему уровню, не проходя текущий уровень, то он может это сделать, нажав на стрелку, которая находится справа от игрового поля. Если же игрок не успешно обвел фигуру, то у него будет еще одна попытка, число попыток не ограничено. Игра заканчивается, когда игрок прошел последний уровень.

**4.1.2 Описание игрового поля**

На игровом поле находится обозначение на каком уровне сейчас находится игрок, само поле с фигурой и точками для обведения, также есть кнопка перехода на следующий уровень и две кнопки для того, чтобы вернуться назад к списку уровней и выход в главное меню. Примерный вид игрового поля показан на рис. 1.

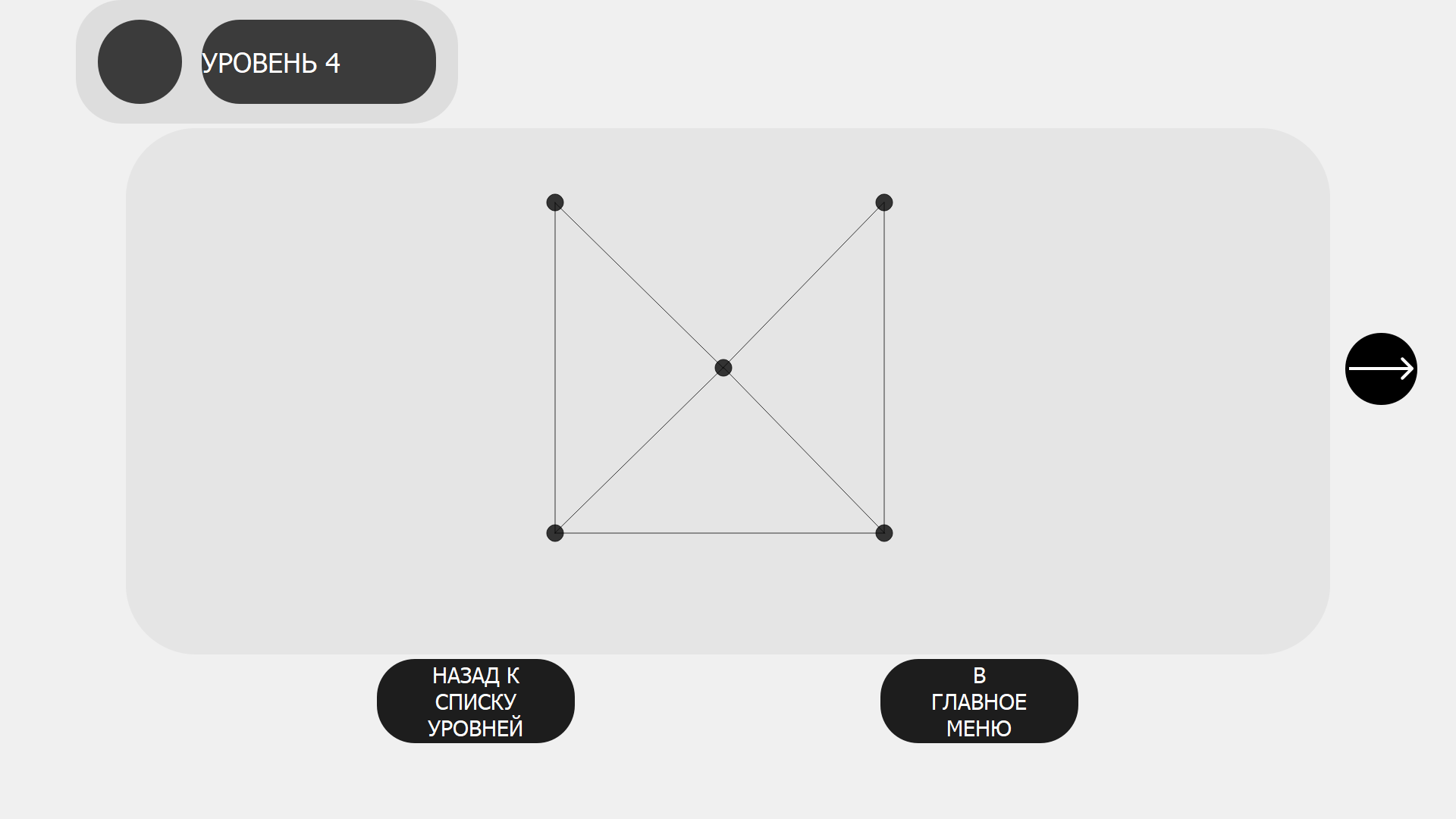


Рис 1 – Примерный вид игрового поля

В игре будет один уровень сложности в котором будет содержаться 5 уровней. На поле главного меню (рис.2) будут находиться кнопки начала игры, кнопка с краткими правилами и кнопка выхода из игры. Также здесь будет находиться логотип с названием игры.



Рис 2 – Примерный вид главного меню

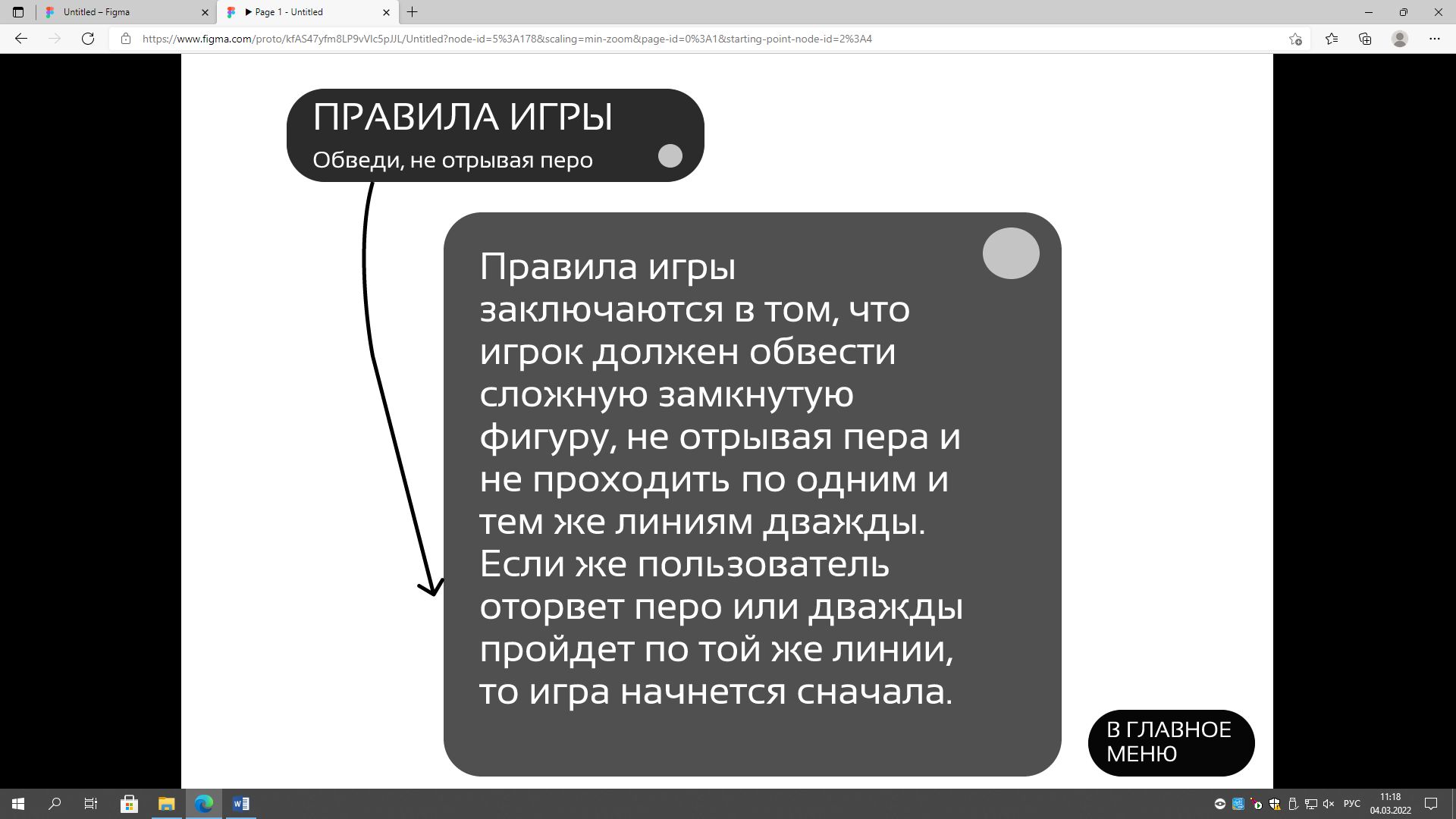


Рис 3 – Правила игры

Также еще на рис.3 прочитав правила, есть кнопка, которая позволит пользователю вернуться в главное меню для того, чтобы начать игру.

Примерный вид окна выбора уровня показан на рис.4

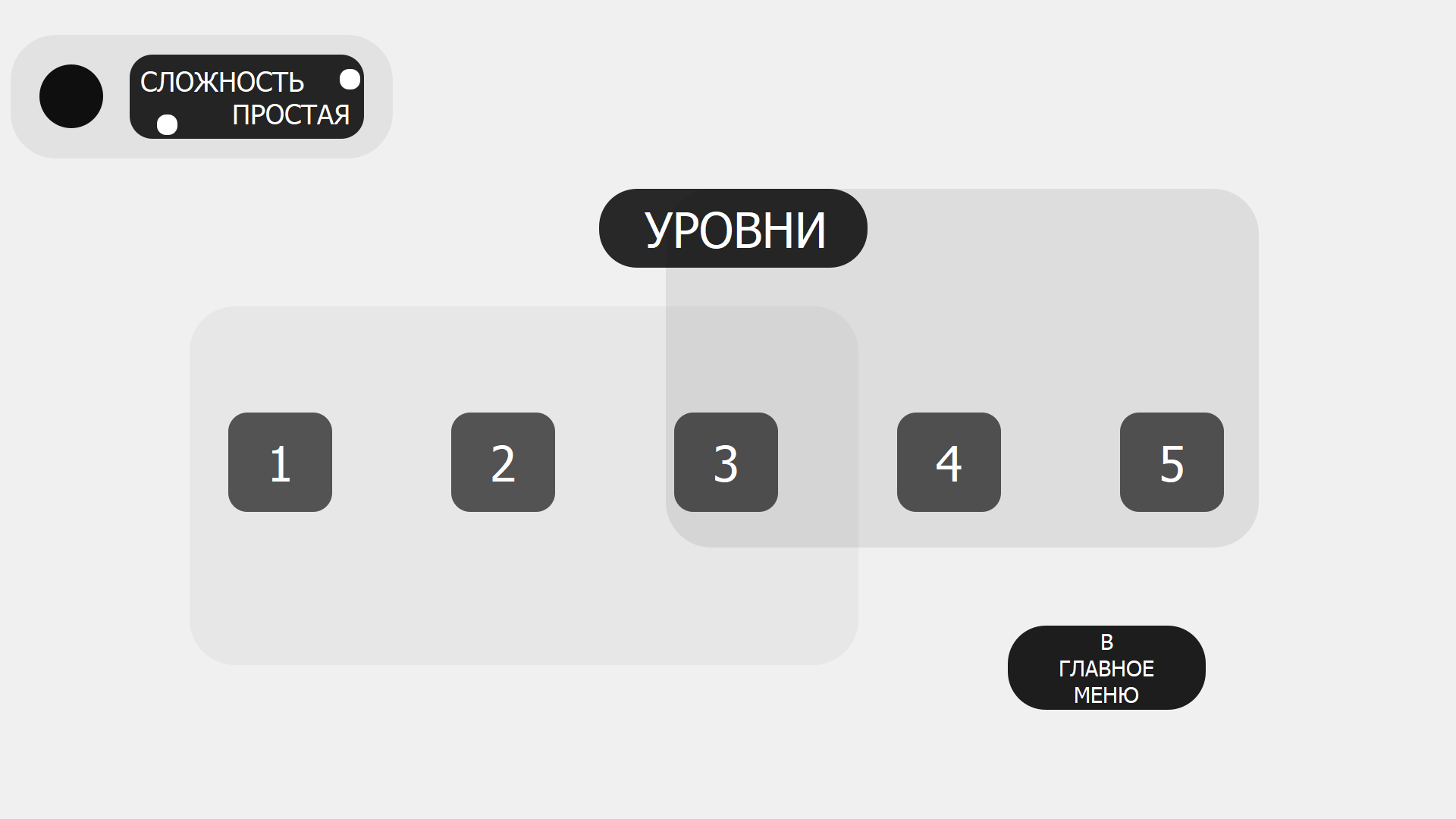


Рис 4 – Уровни игры

В окне выбора уровня игрок может перейти назад в главное меня для того, чтобы выйти из игры. Еще он может выбрать уровень, который он хочет пройти, нажав на нужную кнопку.

**4.1.3 Возможности пользователя**

Пользователю должны быть доступны следующие возможности:

* если нажать на кнопку играть, пользователь будет проходить все уровни подряд, то есть с самого первого уровня до самого последнего уровня;
* перед началом игры пользователь может прочитать правила, нажав на главном экране на кнопку «об игре»;
* если нажать на кнопку выход, пользователь будет выходить из игры.

**4.2 Требования к надежности**

Надежное (устойчивое) функционирование программы должно быть обеспечено выполнением Заказчиком совокупности организационно-технических мероприятий, перечень которых приведен ниже:

* организацией бесперебойного питания технических средств;
* использованием лицензионного программного обеспечения;
* регулярным выполнением рекомендаций Министерства труда и социального развития РФ, изложенных в Постановлении от 23 июля 1998 г. Об утверждении межотраслевых типовых норм времени на работы по сервисному обслуживанию ПЭВМ и оргтехники и сопровождению программных средств»;
* регулярным выполнением требований ГОСТ 51188-98. Защита информации. Испытания программных средств на наличие компьютерных вирусов.

**4.3 Условия эксплуатации**

Условия эксплуатации программы должны соответствовать стандартным требованиям, предъявляемым к бытовым помещениям.

**4.4 Требования к составу и параметрам технических средств**

Для корректной работы программы необходимо:

* процессор х86-64 не ниже 800 Мгц;
* ОЗУ не менее 256 Мб;
* VGA-совместимый видеоадаптер;
* не менее 100 Мб на жёстком диске;
* совместимая аудио-карта;
* HID-средства: клавиатура, мышь.

**4.5 Требования к информационной и программной совместимости**

Программа должна быть написана на языке Python.

**4.6 Требования к маркировке и упаковке**

Специальных требований нет.

**4.7 Требования к транспортированию и хранению**

Специальные требования к транспортированию и хранению не предъявляются.

**4.8 Специальные требования**

Простота интерфейса пользователя.

**5 Требования к программной документации**

Состав программной документации должен включать:

* техническое задание;
* код программы;
* руководство пользователя.

## **ТЕХНИЧЕСКИЙ ПРОЕКТ**

**Структура входных / выходных данных**

Входными данными для игры являются:

1. Запрос на загрузку конструирование и вызов экземпляра базового виджета;
2. Запрос на подключение qt-библиотеки, где при смене значений которых в qt подаётся сигнал и создаются несколько таких переменных;
3. Запрос на подключение к папе UI, которая служит для отрисовки графических элементов в игре;
4. Различные запросы из файлов скриптов (.py).

Выходными данными являются окно программы, его наполнение, а также все варианты взаимодействия пользователя с элементами программы.

1. **Форма представления входных / выходных данных**

Особой формы представления входных данных не предусматривается, для игры предусмотрены экранные формы, которые представлены в техническом задании на разработку игры «Обведи, не отрывая пера».

1. **Определение семантики использования языков, библиотек и фреймворков**

Язык Python - используется для написания основной логики игры, включая алгоритмы проверки существования вершин графа, проверки координат начальной и конечной точки вершин. Python выбран из-за его простоты, гибкости и широкой поддержки различных библиотек.

Использовались библиотеки и фреймворки, такие как: QMainWindow, QVBoxLayout, QWidget, QLabel, QT, buttons\_layout, QPixmap, PyQt5. Они служат для отрисовки окон и кнопок игры через пользовательский интерфейс (UI) - используются для создания графического интерфейса пользователя (GUI) игры. Они предоставляют различные компоненты, такие как окна, кнопки, текстовые поля и т. д., которые могут быть использованы для создания интерактивного пользовательского интерфейса игры. В данном случае, эти библиотеки или фреймворки используются для отображения игровых элементов и обработки пользовательских действий, связанных с ними.

Таким образом, семантика использования языков, библиотек и фреймворков в данном контексте связана с их ролями в разработке игры, включая написание логики игры на языке Python, создание графического интерфейса пользователя с помощью соответствующих библиотек или фреймворков, а также выполнение проверки существования вершины графа с использованием соответствующих алгоритмов или функций.

1. **Взаимодействие с пользователем**

Сценарии взаимодействия пользователя с игрой представлены на диаграмме Use Case на рисунке 1.

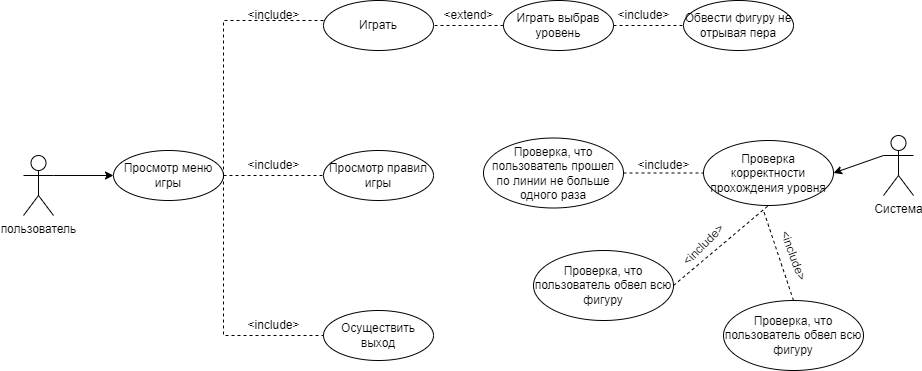


Рисунок 1 – Сценарии взаимодействия с пользователем

1. **Структура программного продукта**

На рисунке 2 представлена диаграмма активности процесса:

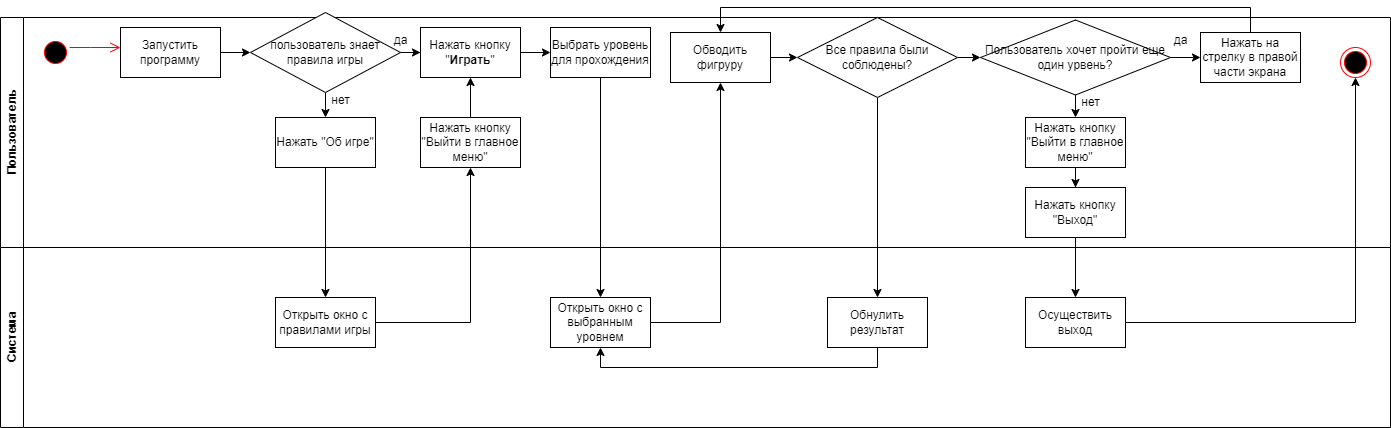


Рисунок 2 – Диаграмма активности процесса

Диаграмма активности используется для моделирования последовательности действий или процессов в системе. Она помогает визуализировать и понять, как различные активности взаимодействуют друг с другом в рамках процесса.

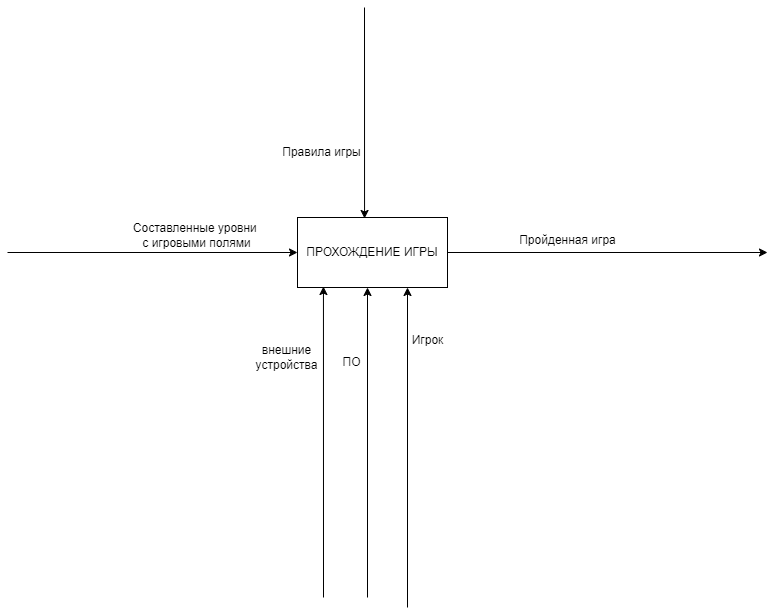


Рисунок 3 – Диаграмма IDEF0

IDEF0 диаграмма на рисунке 3 позволяет идентифицировать основные функции и их взаимосвязи в рамках процесса, что помогает выявить узкие места, оптимизировать процессы и улучшить эффективность работы.

По диаграмме можно определить необходимые функции и их взаимосвязи, что поможет разработчику создать систему, соответствующую требованиям и целям.

Внешней входящей информацией являются:

* Составленные уровни с игровыми полями. Они включают в себя различные уровни или этапы в компьютерных играх, где игроку предстоит пройти определенные задания или испытания на специально созданных игровых полях. Эти уровни могут быть разнообразными по своему дизайну и сложности. Игровые поля могут включать в себя ловушки, головоломки и другие элементы, которые игрок должен преодолеть, чтобы завершить уровень в игре. Создание уровней с игровыми полями требует тщательного планирования и балансировки сложности, чтобы обеспечить интересный и захватывающий геймплей для игрока.

Управляющим воздействием будут являться:

* Правила игры.

Механизмом управления будут являться:

* Внешние устройства;
* Программное обеспечение;
* Игрок.

Внешней исходящей информацией будет являться:

* Пройденная игра.

Программа игры «Обведи, не отрывая пера» состоит из трех модулей.  
Разрабатываемый модуль представляет из себя пользовательский интерфейс. основная его задача – визуализация данных, следовательно, здесь необходимо описать форматы взаимодействия разрабатываемого модуля с другими компонентами. Архитектура программного продукта определяет организацию компонентов и связей между ними, чтобы обеспечить эффективную разработку, поддержку и масштабируемость приложения. В данном случае, предложенная файловая структура может служить основой для простой архитектуры модульного приложения. Общее представление архитектуры модулей представлено на рисунке 4.

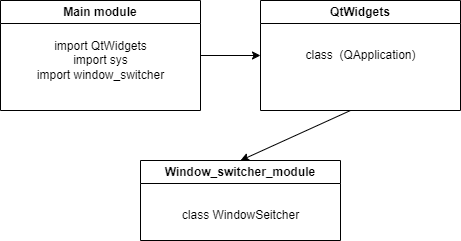


Рисунок 4 – Диаграмма классов

Из предоставленной схемы видно, что представлении архитектуры есть главный модуль (Main module), который импортирует модули QtWidgets, sys и window\_switcher. QtWidgets импортирует набор виджетов и классов для создания графического пользовательского интерфейса в приложениях, использующих фреймворк Qt. Это включает в себя кнопки, поля ввода, метки, таблицы, списки и другие элементы интерфейса, которые могут быть использованы для создания интерактивных приложений. Модуль sys импортирует системные параметры и функции, связанные с интерпретатором Python. Например, с помощью модуля sys можно получить информацию о версии Python, пути поиска модулей, аргументы командной строки и другие системные параметры. Модуль window\_switcher импортирует функции и классы, связанные с переключением окон или управлением окнами в операционной системе Windows. Например, он может содержать функции для переключения между открытыми окнами, изменения размеров окон или другие операции, связанные с управлением окнами на рабочем столе. Модуль QtWidgets содержит класс QApplication, который используется в главном модуле для создания приложения. Модуль window\_switcher содержит класс WindowSwitcher, который также используется в главном модуле для создания объекта окна.

* 1. **Описание взаимодействия с main module**

Файл main\_module содержит главный модуль приложения. В нем происходит импорт нескольких модулей, включая QtWidgets, sys, window\_switcher, main\_window и settings\_window.

* 1. **Описание взаимодействия с QtWidgets**

Модуль QtWidgets содержит класс QApplication, который используется в главном модуле для создания приложения. Этот класс предоставляет основные функции и методы для создания и управления графическим интерфейсом.

* 1. **Описание взаимодействия с модулем window\_switcher**

Модуль window\_switcher содержит класс WindowSwitcher, который также используется в главном модуле для создания объекта окна. Этот класс отвечает за переключение между различными окнами приложения.

* 1. **Файловая структура разрабатываемого модуля**

- main\_module.py (главный модуль приложения)

- window\_switcher.py (модуль для переключения между окнами)

- main\_window.py (модуль для основного окна приложения)

- settings\_window.py (модуль для окна настроек)

В данной структуре каждый модуль находится в отдельном файле, что облегчает организацию и поддержку кода. Каждый модуль содержит свои классы и функции, необходимые для работы соответствующего окна или функционала приложения.  
Такая структура позволяет разделить функциональность приложения на логические блоки, что упрощает разработку, тестирование и поддержку кода. Каждый модуль может быть разработан и изменен независимо от других модулей, что обеспечивает гибкость и масштабируемость приложения.

1. **Прототипы экранных форм**

В соответствие с техническим заданием, разработаны следующие прототипы экранных форм (представлены на рисунках с 5 по 8)

На игровом поле находится обозначение на каком уровне сейчас находится игрок, само поле с фигурой и точками для обведения, также есть кнопка перехода на следующий уровень и две кнопки для того, чтобы вернуться назад к списку уровней и выход в главное меню. Примерный вид игрового поля показан на рис. 5.

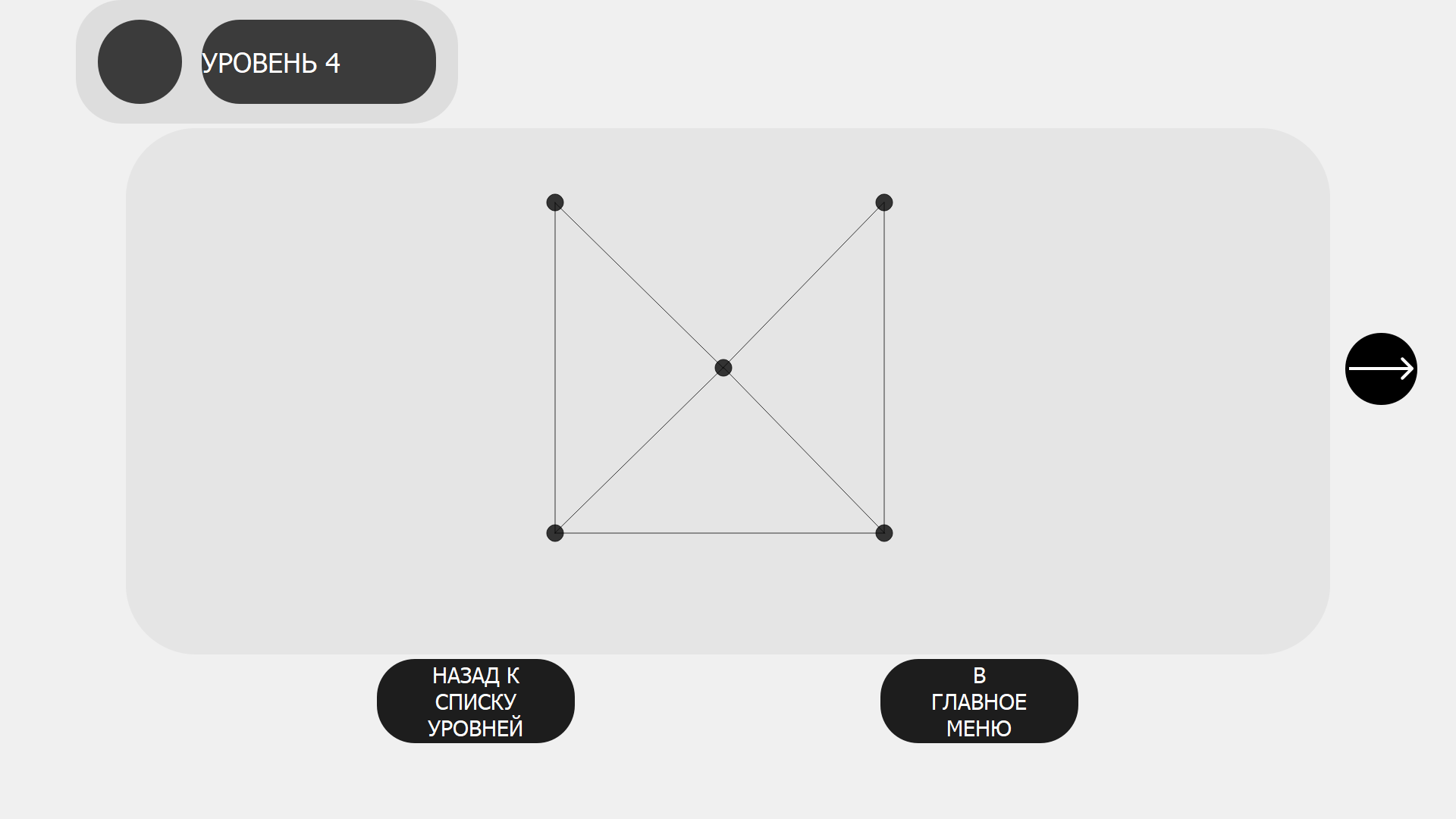


Рисунок 5 – Примерный вид игрового поля

В игре будет один уровень сложности, в котором будет содержаться 5 уровней. На поле главного меню (рис.6) будут находиться кнопки начала игры, кнопка с краткими правилами и кнопка выхода из игры. Также здесь будет находиться логотип с названием игры.



Рисунок 6 – Примерный вид главного меню

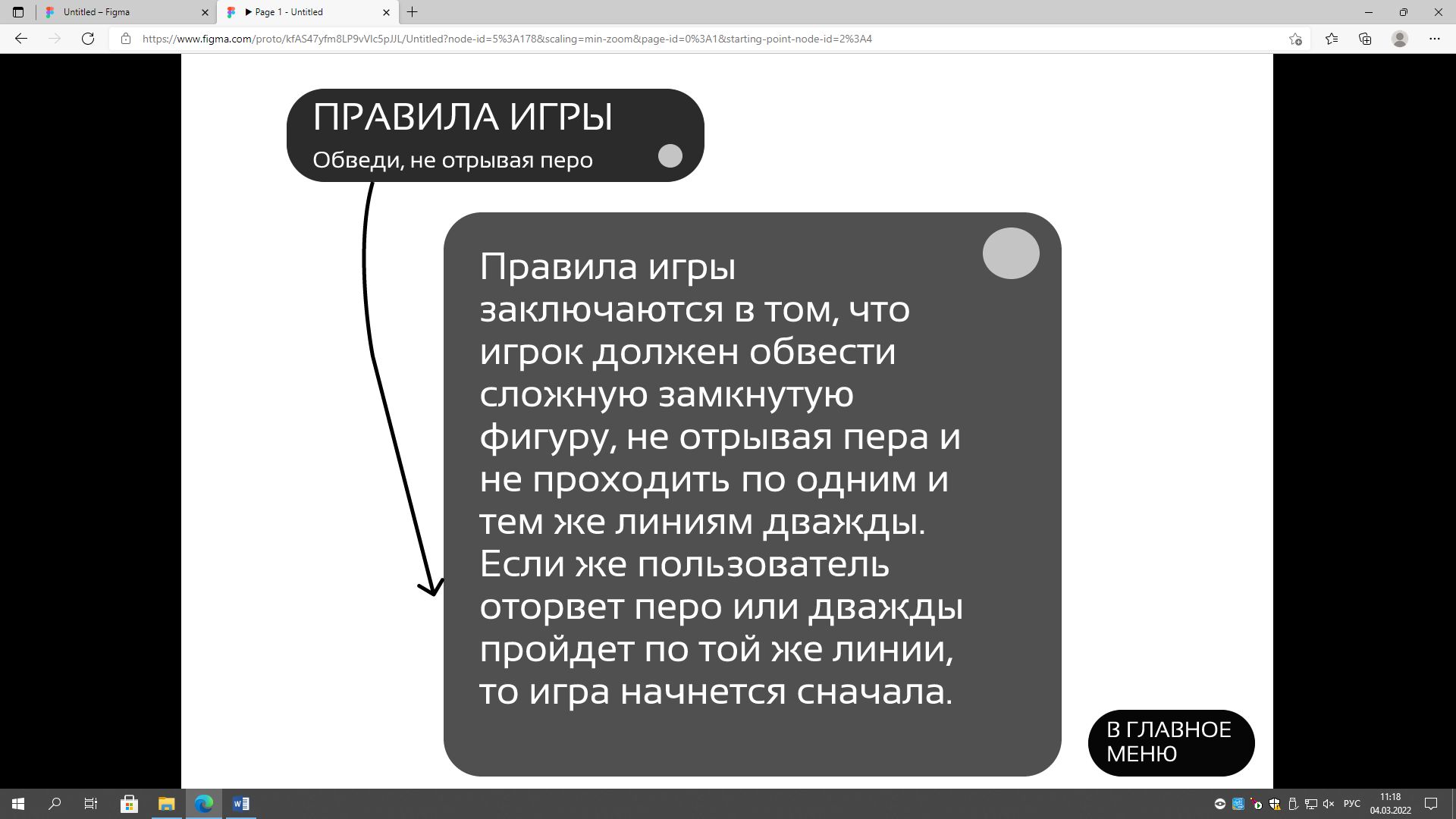


Рисунок 7 – Правила игры

Также еще на рис.7 прочитав правила, есть кнопка, которая позволит пользователю вернуться в главное меню для того, чтобы начать игру.

Примерный вид окна выбора уровня показан на рис.8

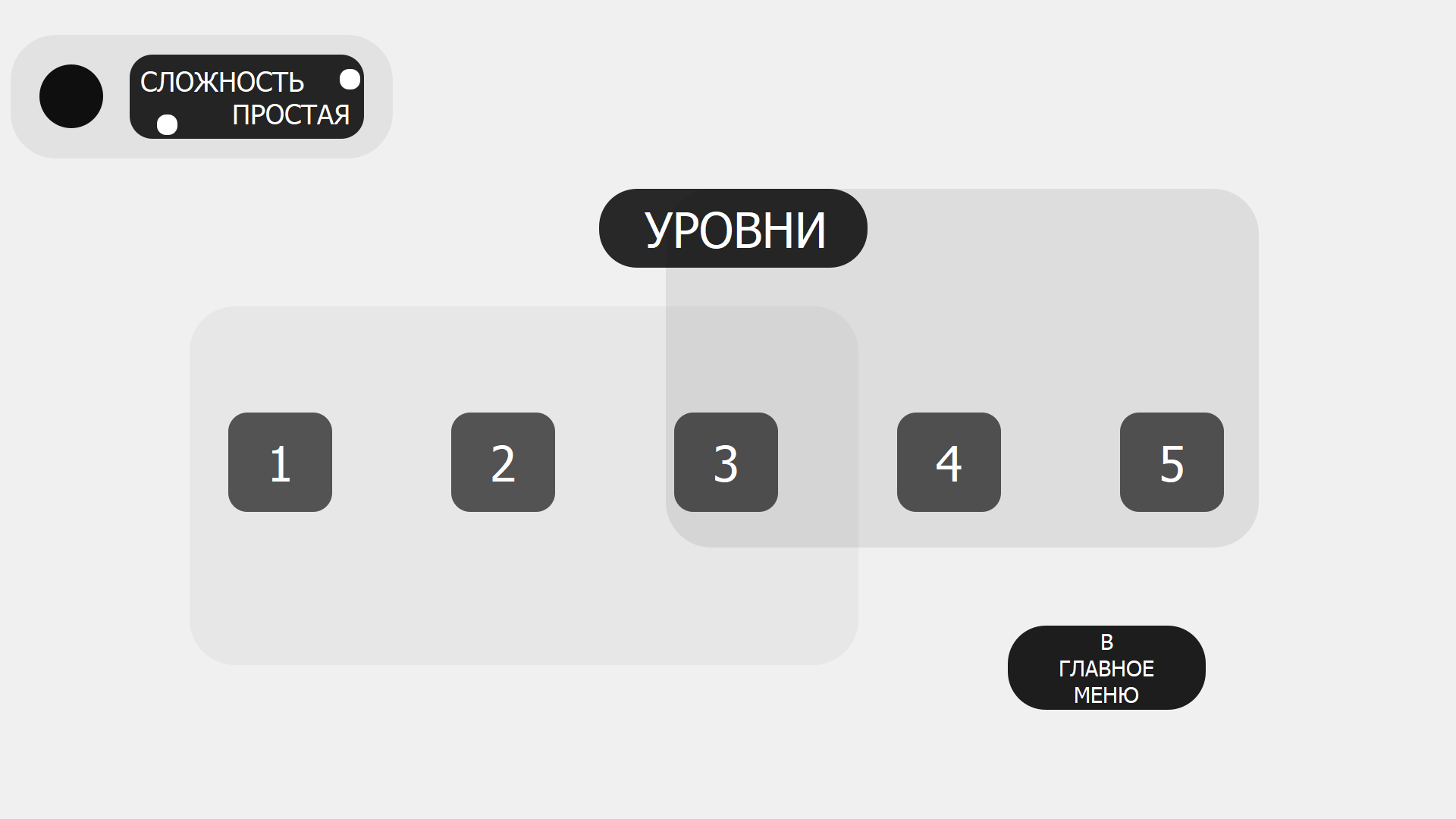


Рисунок 8 – Уровни игры

В окне выбора уровня игрок может перейти назад в главное меня для того, чтобы выйти из игры. Еще он может выбрать уровень, который он хочет пройти, нажав на нужную кнопку.

1. **Конфигурация технических средств**

Требованиями к конфигурации технических средств является ПК любого форм-фактора, соответствующий следующим минимальными требованиям:

* клавиатура;
* устройство воспроизведения звука;
* компьютерная мышь или тачпад;
* монитор с разрешением не менее 1280 на 1080 пикселей;
* место на накопителе в размере 200 Мб;
* процессор двухъядерный с тактовой частотой не менее 2,4 ГГц;
* ОЗУ объемом 4 Гб;
* видеоадаптер DirectX 11 или более поздняя версия.

## **РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ**

**1 Введение**

Настоящий документ является Руководством Пользователя (Далее РП) для игры «Обведи, не отрывая пера».

**1.1 Область применения**

Игра может быть использована любым пользователем младшего школьного и дошкольного возраста и старше с целью отдыха и развлечения.

**1.2 Краткое описание возможностей**

Данное приложение позволяет играть в игру «Обведи, не отрывая пера», проходя заранее подготовленные уровни.

**1.3 Уровень подготовки пользователя**

Все пользователи приложения должны иметь базовые навыки работы на персональном компьютере с графическим пользовательским интерфейсом системы (мышь).

**1.4 Перечень эксплуатационной документации, с которыми необходимо ознакомится пользователю**

Руководство пользователя настоящей игры «Обведи, не отрывая пера».

**2 Назначение и условия применения**

**2.1 Назначение**

Игра «Обведи, не отрывая пера» предназначена для развития логических навыков.

**2.2 Условие применения**

Работа пользователей игры «Обведи, не отрывая пера» возможна при наличие следующих технических средств ЭВМ или ПК:

* Операционная система Microsoft Windows 10/11;
* Оперативная память объемом, не менее 2 Гб;
* Процессор с тактовой частотой от 1.65 ГГц с двумя или более вычислительными ядрами;
* Монитор с минимальным разрешением 1280х720 точек;
* Видеокарта, компьютерная мышь, клавиатура;
* Свободное место на накопителе 125 МБ.

**3 Подготовка к работе**

**3.1 Состав и содержание дистрибутивного носителя данных**

Рабочие места пользователей игры «Обведи, не отрывая пера» не требуют разворачивания приложения из дистрибутива.

**3.2 Порядок загрузки данных и программ**

Для обеспечения правильной работы игры «Обведи, не отрывая пера» необходимо настроить свое рабочее место в соответствии с требованиями к программному обеспечению.

**3.3 Порядок проверки работоспособности**

Проверка работоспособности осуществляется путём запуска соответствующего исполняемого файла и использования представленных возможностей программы.

**4 Описание операций**

**4.1 Запуск приложения**

Для запуска игры «Обведи, не отрывая пера» необходимо войти в папку Pero\_non\_compilable и запустить файл main.py, после чего откроется окно с приложением (См. рисунок 4.1).

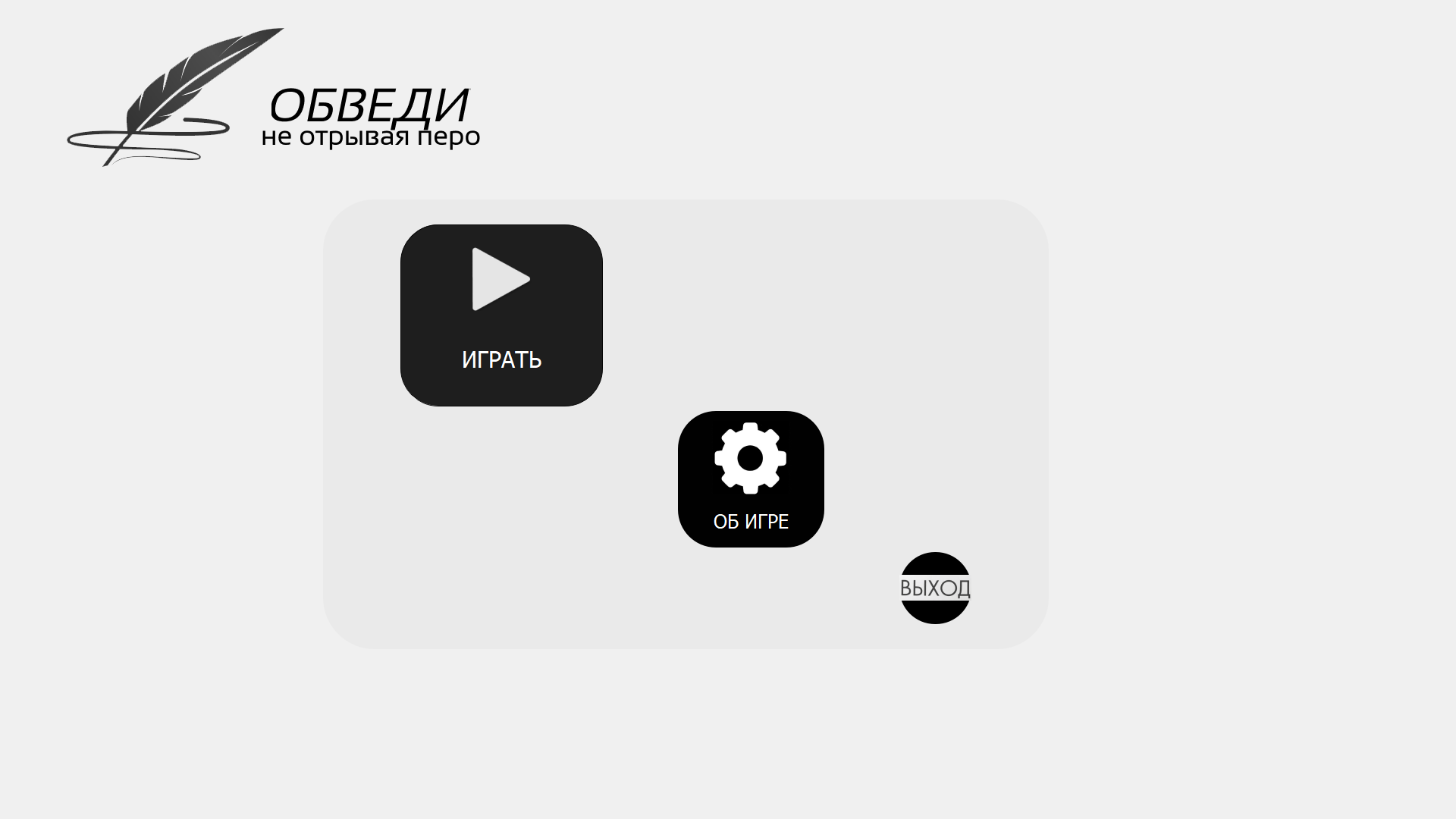


Рисунок 4.1 – Окно при открытии приложения

**4.2 Вывод справки с правилами игры**

Для вывода правил игры «Обведи, не отрывая пера» необходимо нажать на кнопку «Об игре» в главном меню.

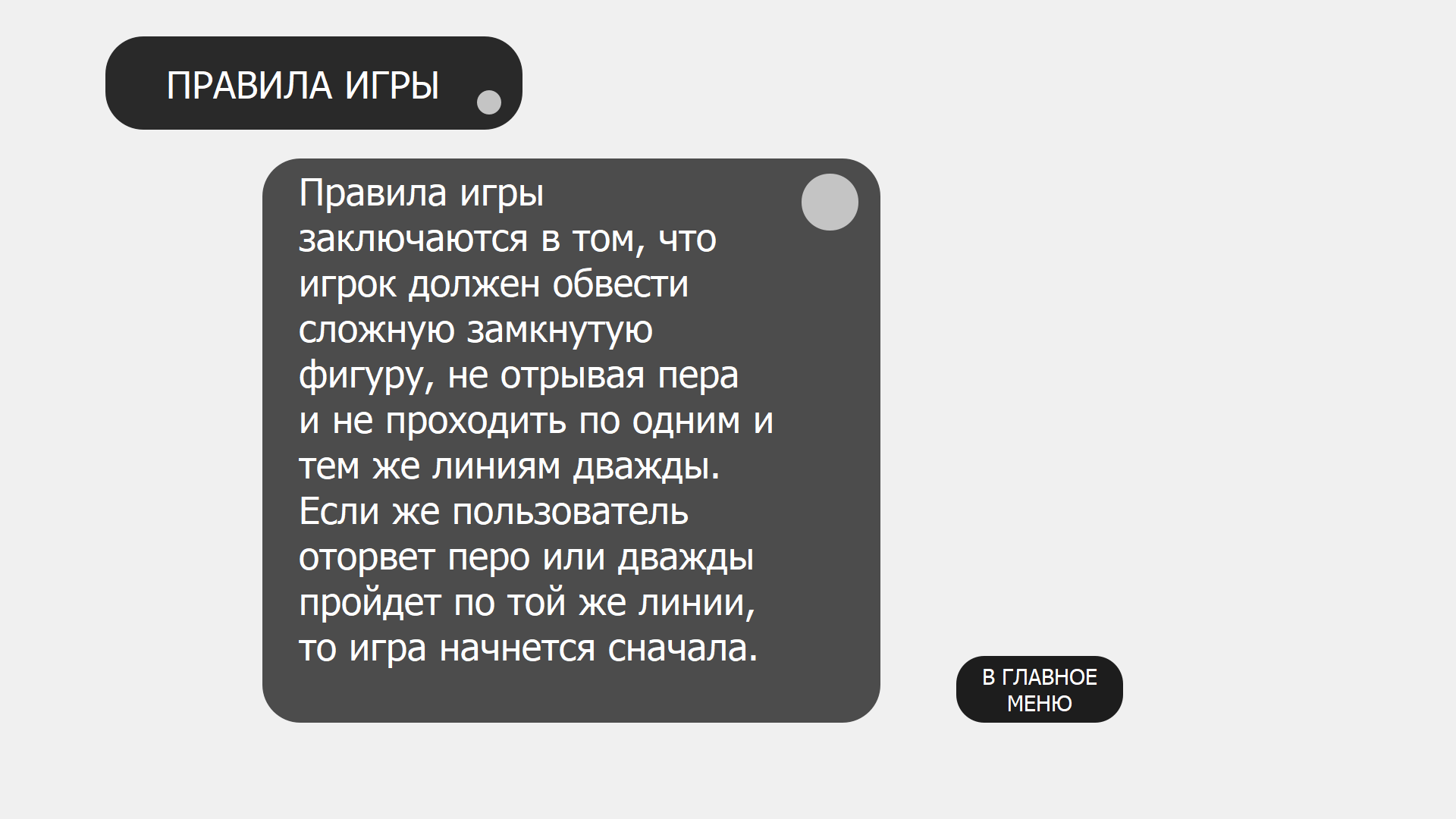


Рисунок 4.2 – Вывод окна с правилами игры на экран

**4.3 Переход к игровому процессу**

Переход на игровое поле может осуществляться таким способом:

* В главном меню при нажатии на кнопку «Играть» происходит открытие экранной формы с выбором уровней (См. рисунок 4.4.1), после выбора, одного из которых происходит переход на игровое поле (См. рисунок 4.4.3);



Рисунок 4.4.1 – Экранная форма с выбором уровней

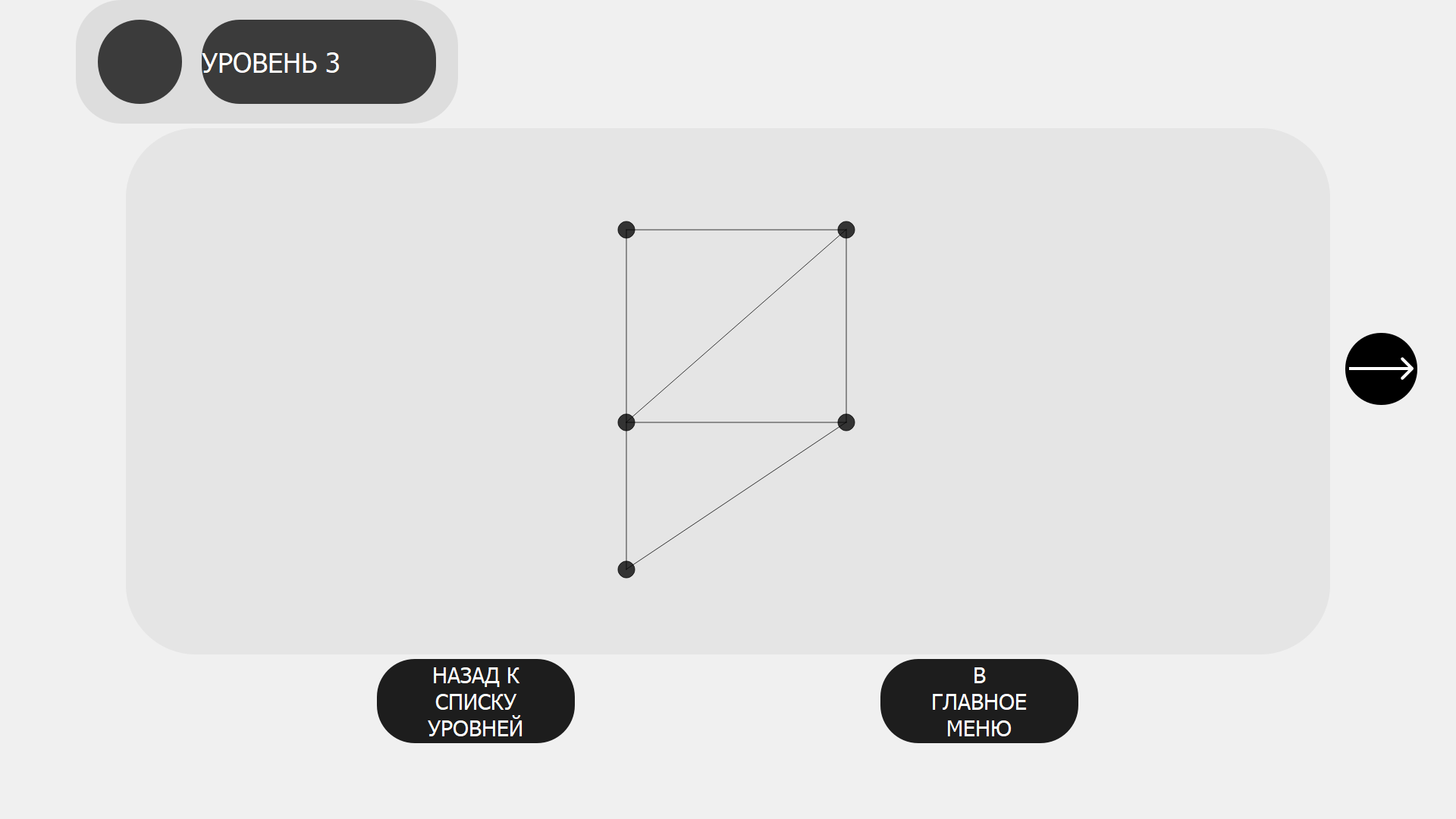


Рисунок 4.4.3 – Игровое поле

**4.4 Правила игры**

Правила игры «Обведи, не отрывая перо» заключается в том, что пользователь (игрок) должен обвести сложную замкнутую фигуру, не отрывая пера и не проходить по одним и тем же линиям дважды. Если же пользователь оторвет перо или дважды пройдет по той же линии, то игра начнется сначала.

**4.5 Выход в главное меню**

Выход в главное меню с игрового поля возможен в двух случаях:

* При нажатии на кнопку «Главное меню» на игровом поле или «Назад к списку уровней» (См. рисунок 4.6.1);

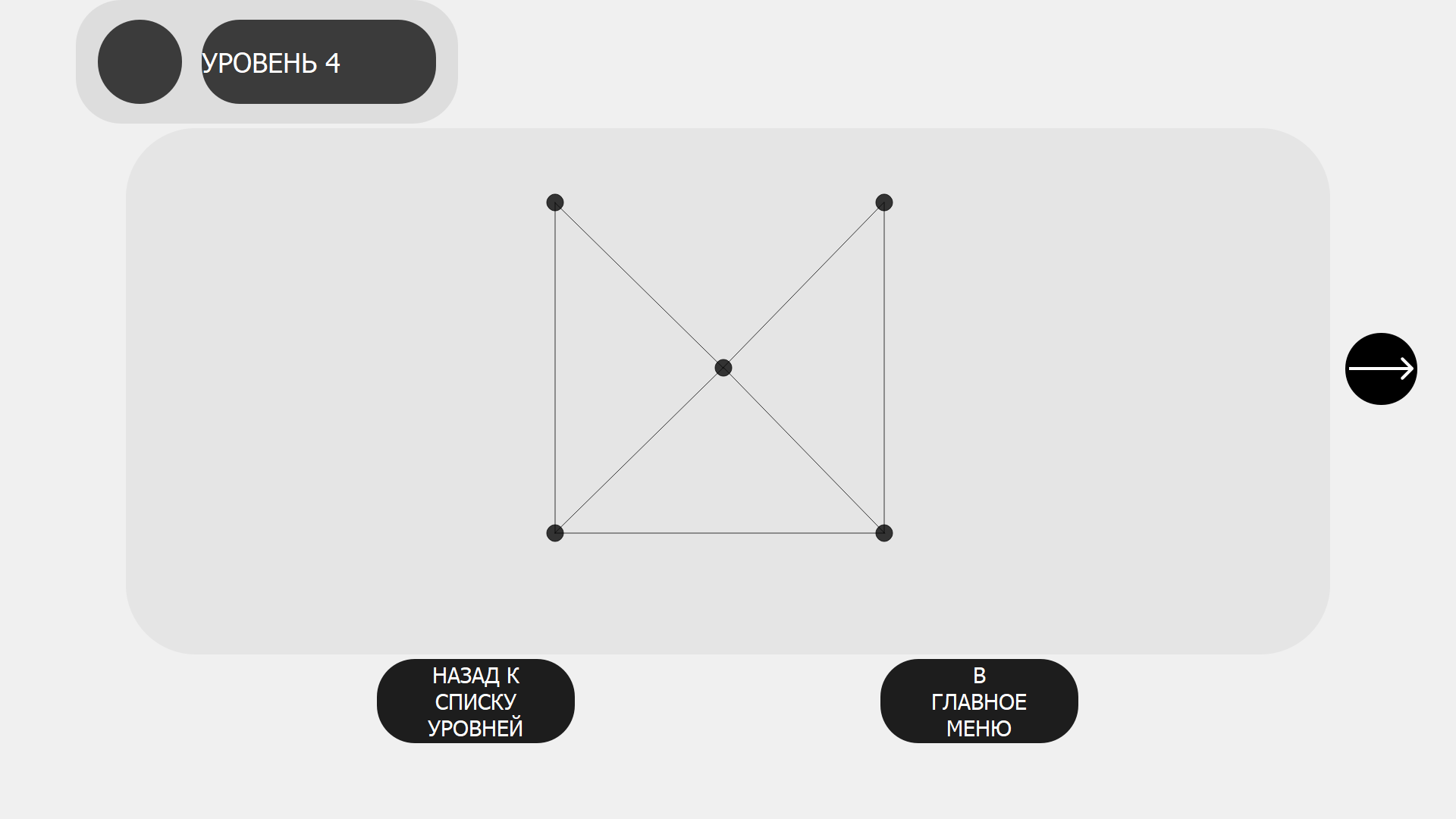


Рисунок 4.6.1 – кнопка «Главное меню»

* 1. **Выход из игры**

Для выхода из игры пользователю необходимо воспользоваться одним из следующих способов:

* Нажать на кнопку закрытия программы «Выход» в нижнем верхнем углу главного меню (См. рисунок 4.7.1);

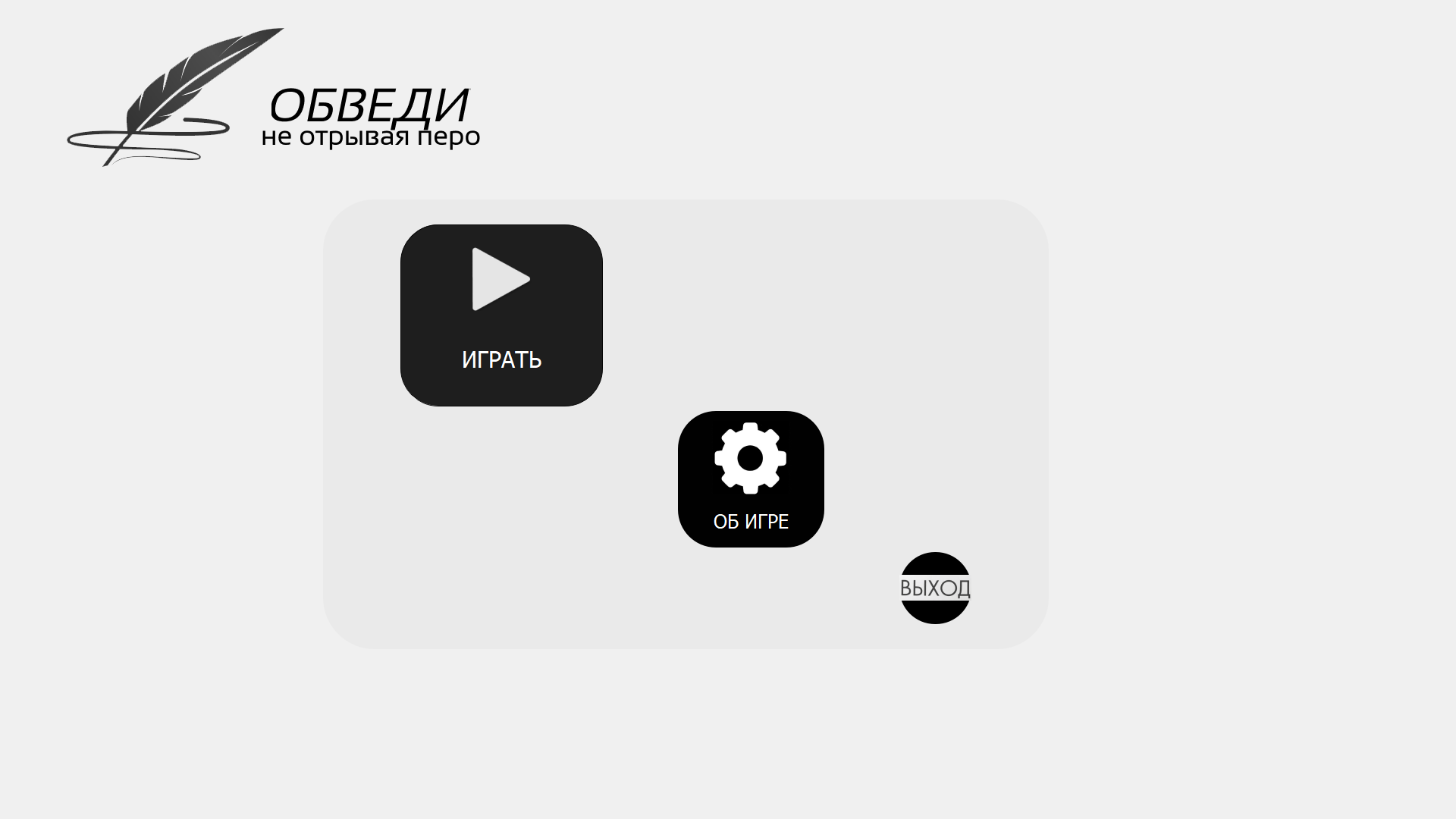


Рисунок 4.7.1 – Кнопка выхода из игры

**5 Аварийные ситуации**

В случае возникновения аварийных ситуаций необходимо перезапустить программу.

**6 Рекомендации к освоению**

Для корректной работы с приложением необходимо:

* Ознакомиться с данным руководством пользователя;
* Иметь базовые умения работы с персональным компьютером.



## **РУКОВОДСТВО ПРОГРАММИСТА**

**1 Инсталляция программы**

**1.1 Пользовательский вариант**

* + 1. Скачать файлы для работоспособности программы, которые находится по ссылке <https://github.com/polinamanylova/Praktika5.git>.
    2. Для установки необходимо запустить файл main.
    3. Проверить работоспособность программы.

**1.2 Вариант поддержания и модернизации**

1. Убедитесь, что на вашем компьютере установлен Python, с библиотекой PyQt5.
2. Склонируйте репозиторий программы с GitHub с помощью команды git clone https://github.com/Deserag/Telegram-bot\_for\_analogov
3. Перейдите в папку вашего проекта с помощью команды cd repository.
4. Установите зависимости проекта, выполнив команду yarn install.
5. Запустите программу для проверки успешности установки командой yarn start.
6. Запустите программу и приступите к прохождению уровней.

**2 Проверка работоспособности игры**

Для проверки работоспособности требуется запустить программу с помощью команд (“start”).

**3 Поддержка программы**

Регулярно проверяйте обновления библиотеки PyQt5, QtWidgets, sys.

Отслеживайте сообщения об ошибках, поступающие от пользователей, и предоставляйте им поддержку, анализируя и исправляя проблемы.

Ведите документацию по программе, включая инструкции по устранению проблем и ответы на часто задаваемые вопросы и храните ее в репозитории на GitHub.

Мониторьте производительность программы и проводите оптимизацию кода, включая Python.

Регулярно резервируйте данные программы, чтобы предотвратить потерю информации в случае сбоев или сбоев системы, используя соответствующие инструменты и методы, и храните их в безопасном месте, отдельно от репозитория.

**4 Модернизация**

Анализируйте обратную связь пользователей и их потребности, чтобы определить возможности для улучшения программы.

Создайте новую ветку в репозитории для разработки новых функций.

Разрабатывайте и реализуйте новые функции или улучшения, основываясь на исследованиях и требованиях пользователей, используя Python для разработки логики программы.

Тестируйте новые функции перед их выпуском, чтобы убедиться в их правильной работе и отсутствии ошибок, используя соответствующие инструменты для тестирования Python кода.

Обновляйте документацию программы, чтобы отразить новые функции и изменения, и убедитесь, что она соответствует актуальной версии программы, и отправьте изменения в репозиторий.

Проводите регулярное обслуживание программы, включая оптимизацию кода, в том числе Python и PyQt5, и обновление зависимостей.

## **ПРОГРАММА И МЕТОДИКА ИСПЫТАНИЙ**

**1 Аннотация**

Настоящая Программа и методика испытаний системы (ЛИ «Обведи, не отрывая пера») предназначена для проверки выполнения заданных функций Системы, определения и проверки соответствия требованиям ТЗ количественных и качественных характеристик Системы, выявления и устранения недостатков в действиях Системы и в разработанной документации на этапе проведения приёмочных испытаний.

Программа и методика испытаний разработана в соответствии с требованиями ГОСТ 34.603–92, ГОСТ 19.301–79.

1. **Объект испытаний**

**2.1 Наименование системы**

Система, предназначенная для организации образовательного и игрового процесса и развития логики в виде игры «Обведи, не отрывая пера».

**2.2 Область применения системы**

Система является программным продуктом для обучения детей работе с компьютером или для использования в развлекательных целях.

**2.3 Условное обозначение системы**

Условное обозначение Системы – «Обведи, не отрывая пера».

1. **Цель испытаний**

Целью проводимых по настоящей программе и методике испытаний «Обведи, не отрывая пера» является определение функциональной работоспособности системы на этапе проведения испытаний, состав и качество сопроводительной документации.

Программа испытаний должна удостоверить работоспособность «Обведи, не отрывая пера» в соответствии с функциональным предназначением.

1. **Общие положения**

**4.1 Перечень руководящих документов, на основании которых проводятся испытания**

Приёмочные испытания «Обведи, не отрывая пера» проводятся на основании следующих документов:

* Утверждённое Техническое задание на разработку «Обведи, не отрывая пера»;
* Настоящая Программа и методика приёмочных испытаний.

**4.2 Место и продолжительность испытаний**

Место проведения испытаний – ПЭВМ, соответствующая минимальным системным требованиям

Продолжительность испытаний устанавливается Приказом Заказчика о составе приёмочной комиссии и проведении приёмочных испытаний.

**4.3 Организации, участвующие в испытаниях**

В приёмочных испытаниях участвуют следующий перечень лиц:

* ВятГУ(колледж) Самоделкин П. А. (Заказчик);
* ВятГУ(колледж) Ржаникова Е. Д. (Заказчик);
* ВятГУ(колледж) Кошкин О. В. (Заказчик)
* Манылова П. М. (Исполнитель)

**4.4 Перечень предъявляемых на испытания документов**

Для проведения испытаний Исполнителем предъявляются следующие документы:

* Методика испытаний «Обведи, не отрывая пера»;
* Техническое задание на создание «Обведи, не отрывая пера».

1. **Объём испытаний**

**5.1 Перечень этапов испытаний и проверок**

Приемочные испытания включают проверку:

* состава и качества сопроводительной документации;
* полноты и качества реализации функций, требований, указанных в ТЗ;
* взаимодействия программы с рекомендуемыми операционными системами;
* взаимодействие программы с архитектурой ПЭВМ, на которой происходит тестирование.

1. **Методика проведения испытаний**

|  |  |  |  |
| --- | --- | --- | --- |
| №  п. п. | Наименование проверки | Выполняемые действия | Ожидаемый результат |
| Сопроводительная документация | | | |
| 1 | Проверка состава и качества сопроводительной документации | Исполнитель предоставляет комиссии:  1) техническое задание на разработку;  2) настоящую программу и методику испытаний;  3) руководство пользователя на разработанное ПО | Документация состоит из Технического задания, Программы и методики испытаний. Техническое задание должно соответствовать ГОСТ 34.602–2020, ГОСТ 19.201–78. |
| Проверка требований | | | |
| 2 | Запуск программы | 1) Нажатие левой кнопкой мыши для открытия файла с расширением exe. | 1) На экране появляется программное окно, где происходит отрисовка главного меню. Главное меню состоит из набора кнопок: «Играть», «Об игре», «Выход». |
| 3 | Нажатие кнопки «Об игре» | 1) Нажатие ЛКМ на кнопку «Об игре»  2) Нажатие ЛКМ на кнопку для выхода «В главное меню». | 1) Происходит открытие правил игры. В главном меню появляется текст с правилами игры, кнопки «Играть» и «Выход», становятся не доступны к использованию.  2) В главном меню исчезает текст с правилами игры, кнопки «Играть» и «Выход» становятся доступны к использованию. |
| 4 | Переход в «Меню выбора уровня» | 1) Нажатие ЛКМ на кнопку «Играть». | 1) Переход к «Меню выбора уровня»  Меню выбора уровня состоит из набора кнопок: «1», «2», «3», «4», «5», «В главное меню». |
| 5 | Переход на игровое поле | 1) Нажатие ЛКМ кнопки «1»/«2»/«3»/«4»/«5». | 1) Переход к игровому полю. Игровое поле состоит из набора игровых элементов и кнопок: точки в количестве от 5 до 10 штук, которые соединены линиями, кнопки «Назад к списку уровней», «В главное меню». |
| 6 | Осуществление правильного хода | 1) Соединены точки, не отпуская ЛКМ и не проведено дважды по одной и той де линии. | 1) Происходит изменение цвета линии там, где провел игрок. |
| 7 | Осуществление неправильного хода | 1) ЛКМ была отпущена  2) Дважды проведено по одной и той же линии | 1) Все проведенные линии сбрасываются и их цвет меняется на исходный  2) Все проведенные линии сбрасываются и их цвет меняется на исходный |
| 8 | Прохождение уровня | 1) Многократное повторение пункта 6 | 1) Взаимодействие с игровыми элементами на игровом поле становиться невозможным. |
| 9 | Использование элементов ввода, которые не осуществляют ввод данных в программу | 1) Нажатие ПКМ в пределах программного окна.  2) Нажатие на любую кнопку клавиатуры. | 1) Программа не реагирует на действие.  2) Программа не реагирует на действие. |
| 10 | Взаимодействие с объектами не имеющими кликабельных поверхностей | 1) Нажатие ЛКМ в пределах окна, но за пределами кликабельных поверхностей (кнопки, игровые элементы). | 2) Программа не реагирует на действие. |
| 11 | Перезапуск уровня | 1) Нажатие ЛКМ на кнопку «Назад к списку уровней» и снова выбрать ЛКМ тот же уровень. | 1) Выбранный ранее уровень принимает исходное состояние. |
| 12 | Осуществление перехода назад в «Назад к списку уровней» и «Главное меню» | 1) Нажатие ЛКМ на кнопку «Назад к списку уровней».  2) Нажатие ЛКМ на кнопку «Главное меню». | 1) Произошел переход в «Меню выбора уровня», появились его экранные формы.  2) Произошел переход в «Главное меню», появились его экранные формы. |
| 13 | Выход из программы | Нажатие ЛКМ на кнопку «Выход» | Программа завершает работу, программное окно закрывается. |

1. **Требования по испытаниям программных средств**

Испытания программных средств ЛИ «Обведи, не отрывая пера» проводятся в процессе функционального тестирования Системы (п. 4.1).

Других требований по испытаниям программных средств ЛИ «Обведи, не отрывая пера» не предъявляется.

1. **Перечень работ, проводимых после завершения испытаний**

По результатам испытаний делается заключение о соответствии ЛИ «Обведи, не отрывая пера» требованиям ТЗ. При этом производится (при необходимости) доработка программных средств и документации.

1. **Условия и порядок проведения испытаний**

Испытания ЛИ «Обведи, не отрывая пера» должны проводиться на целевом оборудовании Заказчика. Оборудование должно быть предоставлено в той конфигурации, которая запланирована для начального развёртывания системы, и указана в Техническом задании.

Во время испытаний проводится полное функциональное тестирование, согласно требованиям, указанным в Техническом задании.

1. **Материально-техническое обеспечение испытаний**

Приёмочные испытания проводятся на программно-аппаратном комплексе Заказчика в следующей минимальной конфигурации:

Рабочее место:

* ПК в составе АРМ пользователя, соответствующий минимальным системным требованиям, указанным в ТЗ;
* Рекомендуемая операционная система MS Windows 10;
* Интерпретатор Python 3.10 64-bit

1. **Метрологическое обеспечение испытаний**

Программа испытаний предполагает использование «Диспетчера задач» для оценки потребления программой ресурсов ПК (недопустимы критические значения во время работы программы).

1. **Отчётность**

Результаты испытаний ЛИ «Обведи, не отрывая пера», предусмотренные настоящей программой, фиксируются в протоколах, содержащих следующие разделы:

* Назначение испытаний и номер раздела требований ТЗ на ЛИ «Обведи, не отрывая пера», по которому проводят испытание;
* Состав технических и программных средств, используемых при испытаниях;
* Указание методик, в соответствии с которыми проводились испытания, обработка и оценка результатов;
* Условия проведения испытаний и характеристики исходных данных;
* Средства хранения и условия доступа к тестирующей программе;
* Обобщённые результаты испытаний;
* Выводы о результатах испытаний и соответствии созданной Системы определённому разделу требований ТЗ на ЛИ «Обведи, не отрывая пера».

В протоколах могут быть занесены замечания персонала по удобству эксплуатации Системы.

Этап проведения предварительных испытаний завершается оформлением «Акта предварительных и приемочных испытаний ЛИ «Обведи, не отрывая пера».

**ПРОТОКОЛ**

**Предварительных и приемочных испытаний ЛИ «Обведи, не отрывая пера»**

В соответствии с требованиями индивидуального задания были проведены испытания ЛИ «Обведи, не отрывая пера» в соответствии с утвержденной «Программой и методикой испытаний».

Общие сведения об испытаниях приведены в таблице 1.

Результаты испытаний приведены в таблице 2.

Таблица 1 - Общие сведения

|  |  |  |
| --- | --- | --- |
| Испытываемый образец: | Специализированная информационная системы «Информационный портал малого предпринимательства» | |
| Дата проведения испытаний: | «30» Октября 2023 г. | |
| Место проведения испытаний | Площадка заказчика(ВятГУ (колледж), аудитория 5-203) | |
| Испытания проводили: | Фамилия, И.О. | Должность |
| От Исполнителя | Манылова П. М. | Студент ИСПк-403-52-00 |
| От Заказчика | Долженкова М. Л. | Преподаватель дисциплины «УП. 08 УЧЕБНАЯ ПРАКТИКА» |

Таблица 2 - Результаты испытаний

| № | Шаг испытаний (проверок) | № пункта  Методики | Отметка о прохождении (да/нет) | Примечания |
| --- | --- | --- | --- | --- |
| 1 | Проверка состава и качества сопроводительной документации | 1 | да | — |
| 2 | Запуск программы | 2 | да | — |
| 3 | Нажатие кнопки «Открытие/закрытие правил игры» | 3 | да | — |
| 4 | Переход в «Меню выбора уровня» | 4 | да | — |
| 5 | Переход на игровое поле | 5 | да | Выбрали 4 уровень |
| 6 | Осуществление правильного хода | 6 | да | Выбрали фигуру из 3 уровня |
| 7 | Осуществление неправильного хода | 7 | да | Выбрали фигуру из 1 уровня |
| 8 | Прохождение уровня | 8 | да | — |
| 9 | Использование элементов ввода, которые не осуществляют ввод данных в программу | 9 | да | — |
| 10 | Взаимодействие с объектами не имеющими кликабельных поверхностей | 10 | да | — |
| 11 | Перезапуск уровня | 11 | да | — |
| 12 | Осуществление перехода назад в «Меню выбора уровня» и «Главное меню» | 12 | да | — |
| 13 | Переход в «Меню выбора уровня» | 13 | да | — |
| 14 | Переход на игровое поле | 14 | да | Выбрали 2 уровень |
| 15 | Прохождение уровня | 15 | да | — |
| 16 | Выход из программы | 16 | да | — |
| 17 | Тестирование приложения на Windows 10 | 17 | да | Программа запустилась и работает без ошибок |
| 18 | Оценка потребления ресурсов ПЭВМ во время работы программы | 18 | да | Программа не использовала ресурсы ПЭВМ свыше чем на 50% |

| № | Шаг испытаний (проверок) | № пункта  Методики | Отметка о прохождении (да/нет) | Примечания |
| --- | --- | --- | --- | --- |
| 1 | Проверка состава и качества сопроводительной документации | 1 |  |  |
| 2 | Запуск программы | 2 |  |  |
| 3 | Нажатие кнопки «Открытие/закрытие правил игры» | 3 |  |  |
| 4 | Переход в «Меню выбора уровня» | 4 |  |  |
| 5 | Переход на игровое поле | 5 |  |  |
| 6 | Осуществление правильного хода | 6 |  |  |
| 7 | Осуществление неправильного хода | 7 |  |  |
| 8 | Прохождение уровня | 8 |  |  |
| 9 | Использование элементов ввода, которые не осуществляют ввод данных в программу | 9 |  |  |
| 10 | Взаимодействие с объектами не имеющими кликабельных поверхностей | 10 |  |  |
| 11 | Перезапуск уровня | 11 |  |  |
| 12 | Осуществление перехода назад в «Меню выбора уровня» и «Главное меню» | 12 |  |  |
| 13 | Переход в «Меню выбора уровня» | 13 |  |  |
| 14 | Переход на игровое поле | 14 |  |  |
| 15 | Прохождение уровня | 15 |  |  |
| 16 | Выход из программы | 16 |  |  |
| 17 | Тестирование приложения на Windows 10/11 | 17 |  |  |
| 18 | Оценка потребления ресурсов ПЭВМ во время работы программы | 18 |  |  |

Таблица 3 - Результаты испытаний

## **ИСХОДНЫЙ КОД**

**Main.py**  
if \_\_name\_\_ == '\_\_main\_\_':  
 from PyQt5 import QtWidgets  
 import sys  
 app = QtWidgets.QApplication(sys.argv) # Конструтирование QApp  
 from UI.window\_switcher import WindowSwitcher  
 window\_switcher = WindowSwitcher()  
 window\_switcher.showFullScreen() # Создание и показ основного окна на полный экран  
 sys.exit(app.exec\_()) # Запуск QApp

**Global.py**

from PyQt5.QtWidgets import QWidget  
from PyQt5.QtCore import pyqtSignal  
  
# Класс переменной с сигналом. Базируется на QWidget, поскольку по-другому QT сигналы работают через раз.  
# Нужен [сигнал] для плавных двусторонних бесшовных переходов между экранными формами  
class VariableWithSignal(QWidget):  
 valueChanged = pyqtSignal(object)  
 def \_\_init\_\_(self):  
 super(VariableWithSignal, self).\_\_init\_\_()  
 self.text = ''  
  
 def setText(self, str):  
 self.text = str  
 # Объявление своего метода "геттера" свойства text  
 @property  
 def text(self):  
 return self.\_text  
 # Объявление своего метода "сеттера" свойства text (нужно для отслеживания и активации сигнала)  
 @text.setter  
 def text(self, value):  
 self.\_text = value  
 self.valueChanged.emit(value)  
  
  
DIFFICULTY = VariableWithSignal()  
DIFFICULTY.text = 'EASY'  
CURRENT\_LEVEL\_NUMBER = VariableWithSignal()  
CURRENT\_LEVEL\_NUMBER.text = 1  
CURRENT\_WINDOW = VariableWithSignal()  
CURRENT\_WINDOW.text = 'MainMenu'  
# Гениальный костыль для интерфейса во многом. Словарь подошел как нельзя кстати  
difficulty\_translator = {'EASY': 'ПРОСТАЯ', 'MEDIUM': 'СРЕДНЯЯ', 'HARD': 'ВЫСОКАЯ'}

**Level\_sample.py**

# Шаблон уровня  
class Level:  
 # Объявление некоторых переменных, отвечающих за прохождение и наполнение каждого из урвоней  
 def \_\_init\_\_(self, vertices\_positions, edges\_list):  
 self.vertices\_positions = vertices\_positions  
 self.edges\_list = edges\_list  
 self.edges\_positions\_list = calculate\_edges\_positions\_list(self.vertices\_positions, self.edges\_list)  
 self.bypassed\_edges = []  
 self.current\_painted\_vertex = -1  
 self.previous\_painted\_vertex = -1  
  
 # Проверка того, что заданное ребро имеется в списке ребер  
 def edge\_exists(self, edge):  
 return edge in self.edges\_list  
  
 # Проверка существования вершины графа (разница между заданной координатой и пройденной пользователем может быть  
 # чуть больше нуля, поскольку вершины графа изображаются кругом с некоторым радиусом (a/2)  
 # В случае нахождения такой вершины возвращает ее номер в списке вершин, иначе -1  
 def vertex\_exists(self, vertex\_x, vertex\_y):  
 for i in range(len(self.vertices\_positions)):  
 vertex = self.vertices\_positions[i]  
 if abs(vertex[0] - vertex\_x) <= 15 and abs(vertex[1] - vertex\_y) <= 15:  
 return i  
 return -1  
  
# эта функция пишет координаты  
# (координаты начальной и конечной точки)  
def calculate\_edges\_positions\_list(vertices\_positions, edges\_list):  
 edges\_positions\_list = []  
 for edge in edges\_list:  
 edges\_positions\_list.append(vertices\_positions[edge[0]] + vertices\_positions[edge[1]])  
 return edges\_positions\_list

**Levels.py**

from core.level\_sample import Level as Level  
  
# Описание всех уроовней в игре  
# Level ([(vertex\_x, vertex\_y)], [(vertex\_number, vertex\_number)])  
EASY\_1 = Level([(411, 162), (411, 454), (565, 454), (565, 250), (699, 294), (699, 103), (565, 162)],  
 [[0, 1], [1, 2], [2, 3], [3, 4], [4, 5], [5, 6], [0, 6], [3, 6]])  
EASY\_2 = Level([(287, 234), (542, 67), (795, 234), (721, 475), (369, 475)], [[0, 3], [0, 2], [1, 3], [1, 4], [2, 4]])  
EASY\_3 = Level([(426, 78), (426, 332), (426, 526), (716, 78), (716, 332)], [[0, 1], [0, 3], [1, 2], [1, 3], [1, 4],  
 [2, 4], [3, 4]])  
EASY\_4 = Level([(332, 42), (332, 478), (766, 42), (766, 478), (554, 260)], [[0, 1], [0, 4], [1, 3], [1, 4], [2, 3],  
 [2, 4], [3, 4]])  
EASY\_5 = Level([(391, 155), (542, 155), (693, 155), (468, 53), (618, 53),  
 (391, 395), (542, 395), (693, 395), (468, 478), (618, 478)  
 ], [[0, 1], [0, 3], [1, 2], [1, 3], [1, 4], [2, 4],  
 [0, 5], [2, 7],  
 [5, 8], [6, 8], [6, 9], [7, 9]])

**back\_menu.py**

class BackMenu(QGridLayout):  
 # full отвечает за наличие/отсутствие кнопки возвращения к списку уровней  
 def \_\_init\_\_(self, full: bool = False):  
 super(BackMenu, self).\_\_init\_\_()  
 # Объявление кнопки  
 self.back\_to\_main\_menu\_button = QToolButton()  
 self.back\_to\_main\_menu\_button.setFixedSize(261, 111)  
 self.back\_to\_main\_menu\_button.setText('В\nГЛАВНОЕ\nМЕНЮ')  
 # Установка стиля кнопки (css)  
 self.back\_to\_main\_menu\_button.setStyleSheet('''background: rgba(0, 0, 0, 0.88);  
 border-radius: 50px;  
 font-size: 29px;  
 color: #FFFFFF;''')  
 # Установка функции-ответчика на нажатие на кнопку  
 self.back\_to\_main\_menu\_button.clicked.connect(lambda: return\_to('MainMenu'))  
 self.back\_to\_level\_list = QToolButton()  
 self.back\_to\_level\_list.setFixedSize(261, 111)  
 self.back\_to\_level\_list.setText('НАЗАД К\nСПИСКУ\nУРОВНЕЙ')  
 self.back\_to\_level\_list.clicked.connect(lambda: return\_to('LevelSwitcher'))  
 self.back\_to\_level\_list.setStyleSheet('''background: rgba(0, 0, 0, 0.88);  
 border-radius: 50px;  
 font-size: 29px;  
 color: #FFFFFF;''')  
 # Добавление виджетов в список виджетов. Последние две цифры означают координатное положение в сетке  
 # (строка, столбец)  
 if full:  
 self.addWidget(self.back\_to\_level\_list, 0, 2)  
 self.addWidget(self.back\_to\_main\_menu\_button, 0, 3)  
 else:  
 self.addWidget(self.back\_to\_main\_menu\_button, 0, 3)  
  
# Простая функция перехода к другой экранной форме (указывается в аргументе, позже анализируется в WindowSwitcher)  
# Как было описано ранее, аргумент этой функции передается через qtSignal в WindowSwitcher  
def return\_to(widget):  
 core.GLOBAL.CURRENT\_WINDOW.text = widget

**Ellipse.py**

from PyQt5.QtWidgets import QLabel  
  
# В игре много эллипсов. Решил запилить класс, отвечающий за создание такой красоты  
# Название аргументов буквально отражает их суть  
class Ellipse(QLabel):  
 def \_\_init\_\_(self, color, radius, border\_radius=50):  
 super(Ellipse, self).\_\_init\_\_()  
 self.setStyleSheet(f'''background: rgba{color};  
 border-radius: {border\_radius}%;   
 ''')  
 self.color = color  
 self.radius = radius  
 self.setFixedSize(radius, radius)  
  
 def correctStyleSheet(self, newStyleSheet):  
 self.setStyleSheet(newStyleSheet)

**Level\_switcer.py**

# Создание класса формы с кнопками уровней  
class LevelSwitcher(QMainWindow):  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
  
 self.central\_widget = QWidget()  
 self.setCentralWidget(self.central\_widget)  
  
 # Объявление главного шаблона формы и добавление в него индикатора сложности (добавляется извне)  
 self.main\_layout = QVBoxLayout()  
 self.difficulty\_indicator = difficulty\_indicator.DifficultyIndicator()  
 self.difficulty\_indicator.main\_widget.setParent(self.central\_widget)  
 self.difficulty\_indicator.main\_widget.move(14, 46)  
  
 # Создание и добавление таблички УРОВНИ  
 self.level\_label = QLabel(self.central\_widget)  
 self.level\_label.move(890 - self.level\_label.width(), 249)  
 self.level\_label.setText(' УРОВНИ ')  
 self.level\_label.setFixedSize(354, 104)  
 self.level\_label.setStyleSheet('''background: rgba(3, 3, 3, 0.85);  
 border-radius: 50px;  
 font-weight: 400;  
 color:white;  
 font-size: 64px;''')  
  
 # Создание формы с кнопками  
 self.buttons\_layout = QHBoxLayout()  
 self.buttons\_layout.setContentsMargins(150, 0, 150, 0)  
 # Последовательная генерация кнопок  
 for i in range(1, 6):  
 self.level\_button = QToolButton()  
 self.level\_button.setText(str(i))  
 self.level\_button.setFixedSize(137, 131)  
 # Добавление только что созданной кнопке функции по нажатию. (sender запоминает и выдает указатель на конкретный объект)  
 self.level\_button.clicked.connect(lambda: setAndStartLevel(self.sender().text()))  
 self.level\_button.setStyleSheet('''  
 background: rgba(6, 6, 6, 0.66);  
 border-radius: 25px;  
 color: white;  
 font-size: 64px;  
 ''')  
 self.buttons\_layout.addWidget(self.level\_button)  
  
 # Добавление меню возвращения  
 self.back\_menu = BackMenu()  
 self.back\_menu.setContentsMargins(1000, 0, 0, 0)  
 self.main\_layout.setContentsMargins(0, 400, 0, 0)  
 self.main\_layout.addLayout(self.buttons\_layout)  
 self.main\_layout.addLayout(self.back\_menu)  
  
 self.central\_widget.setLayout(self.main\_layout)  
  
  
def setAndStartLevel(level\_number):  
 core.GLOBAL.CURRENT\_LEVEL\_NUMBER.text = level\_number  
 core.GLOBAL.CURRENT\_WINDOW.text = 'LevelWindow'

**Difficulty\_indicator.py**

from PyQt5.QtWidgets import QLabel, QWidget, QHBoxLayout  
from UI.common.ellipse import Ellipse  
import core.GLOBAL  
  
# Класс индикатора сложности. Встраивается в LevelSwitcher  
class DifficultyIndicator:  
 def \_\_init\_\_(self):  
 # Объявление главного (центрального виджета)  
 self.main\_widget = QWidget()  
 self.main\_widget.setFixedSize(504, 163)  
 self.main\_widget.setStyleSheet('background: rgba(1, 1, 1, 0.06);\  
 border-radius: 60px;')  
 # Объявление элементов индикатора  
 self.elements\_layout = QHBoxLayout()  
 self.ellipse = QLabel()  
 self.ellipse.setFixedSize(84, 84)  
 self.ellipse.setStyleSheet('background: rgba(1, 1, 1, 0.94);\  
 border-radius: 42px;')  
 self.rectangle\_with\_difficulty = QLabel()  
 self.rectangle\_with\_difficulty\  
 .setText(f'СЛОЖНОСТЬ\n {core.GLOBAL.difficulty\_translator[core.GLOBAL.DIFFICULTY.text]}')  
 core.GLOBAL.DIFFICULTY.valueChanged.connect(lambda: self.rectangle\_with\_difficulty.setText(  
 f'СЛОЖНОСТЬ\n {core.GLOBAL.difficulty\_translator[core.GLOBAL.DIFFICULTY.text]}'))  
 self.rectangle\_with\_difficulty.setStyleSheet("background: rgba(0, 0, 0, 0.84);\  
 font-size:36px;\  
 color:white;\  
 padding: 5px;\  
 border-radius: 30px;")  
  
 self.rectangle\_with\_difficulty.setFixedSize(309, 111)  
 self.elements\_layout.addWidget(self.ellipse)  
 self.elements\_layout.addWidget(self.rectangle\_with\_difficulty)  
 self.main\_widget.setLayout(self.elements\_layout)  
 # Создание некоторых беллых эллипсов возле текста индикатора  
 self.white\_ellipse\_1 = Ellipse((255, 255, 255, 1), 27)  
 self.white\_ellipse\_1.correctStyleSheet(f'''background: rgba{self.white\_ellipse\_1.color};  
 border-radius: 12px; ''')  
 self.white\_ellipse\_1.setParent(self.main\_widget)  
 self.white\_ellipse\_2 = Ellipse((255, 255, 255, 1), 27)  
 self.white\_ellipse\_2.correctStyleSheet(f'''background: rgba{self.white\_ellipse\_1.color};  
 border-radius: 12px; ''')  
 self.white\_ellipse\_2.setParent(self.main\_widget)  
 self.white\_ellipse\_1.move(193, 105)  
 self.white\_ellipse\_2.move(434, 45)

**Level\_window.py**

from PyQt5.QtWidgets import QMainWindow, QVBoxLayout, QWidget, QToolButton  
from PyQt5.QtGui import QPainter, QColor, QPen, QIcon  
from PyQt5.QtCore import Qt, QSize  
import core.GLOBAL  
from UI.LevelWindow.elements.level\_painter import LevelPainter  
from UI.LevelWindow.elements.level\_indicator import LevelIndicator  
from UI.common.back\_menu import BackMenu  
  
# Класс Окна с уровнем  
class LevelWindow(QMainWindow):  
 def \_\_init\_\_(self):  
 super(LevelWindow, self).\_\_init\_\_()  
 self.central\_widget = QWidget()  
 self.setCentralWidget(self.central\_widget)  
 self.mouse\_pos\_start = 0  
 self.mouse\_pos\_end\_list = []  
 self.clicked = False # Некоторые переменные для выполнения игровых действий  
 self.level\_over = False  
  
 self.central\_vertical\_layout = QVBoxLayout(self.central\_widget)  
 self.central\_vertical\_layout.setContentsMargins(100, 0, 100, 100)  
 self.level\_indicator = LevelIndicator()  
  
 self.back\_menu = BackMenu(True)  
  
 # Генерация содержимого окна  
 self.level\_content = LevelPainter(core.GLOBAL.CURRENT\_LEVEL\_NUMBER.text)  
 core.GLOBAL.CURRENT\_LEVEL\_NUMBER.valueChanged.connect(self.retake\_level)  
 self.central\_vertical\_layout.addWidget(self.level\_indicator.main\_widget)  
 self.central\_vertical\_layout.addWidget(self.level\_content)  
 self.central\_vertical\_layout.addLayout(self.back\_menu)  
  
 # Создание кнопки перехода на следующий уровень  
 self.next\_button = QToolButton(self)  
 self.next\_button.setIcon(QIcon('UI/MainMenu/elements/arrow.png'))  
 self.next\_button.setIconSize(QSize(95, 95))  
 self.next\_button.setFixedSize(95, 95)  
 self.next\_button.setStyleSheet('''color:white;  
 border-radius:47%;  
 background:#000000;  
 font-size:96px;''')  
 self.next\_button.clicked.connect(self.play\_next\_level)  
 self.next\_button.move(1774, 439)  
  
 # Функция перерисовки и смены уровня + генерация содержимого окна  
 def retake\_level(self):  
 self.central\_vertical\_layout.removeWidget(self.level\_indicator.main\_widget)  
 self.central\_vertical\_layout.removeWidget(self.level\_content)  
 self.central\_vertical\_layout.removeItem(self.back\_menu)  
 self.level\_indicator = LevelIndicator()  
 self.level\_content = LevelPainter(core.GLOBAL.CURRENT\_LEVEL\_NUMBER.text)  
 self.central\_vertical\_layout.addWidget(self.level\_indicator.main\_widget)  
 self.central\_vertical\_layout.addWidget(self.level\_content)  
 self.central\_vertical\_layout.addLayout(self.back\_menu)  
 self.restart\_level()  
 self.update()  
 def play\_next\_level(self):  
 core.GLOBAL.CURRENT\_LEVEL\_NUMBER.text = \  
 int(core.GLOBAL.CURRENT\_LEVEL\_NUMBER.text) + 1 if int(core.GLOBAL.CURRENT\_LEVEL\_NUMBER.text) + 1 < 6 else 1  
 self.retake\_level()  
 self.update()  
  
 # Отслеживание того что мышь зажата  
 def mousePressEvent(self, event):  
 super(LevelWindow, self).mousePressEvent(event)  
 if event.button() == Qt.MouseButton.LeftButton and not self.level\_over:  
 self.clicked = True  
 self.mouse\_pos\_start = event.pos()  
 self.mouse\_pos\_end\_list.append(event.pos())  
 self.update()  
  
 # Отслеживание того что ЛКМ отпущена  
 def mouseReleaseEvent(self, event):  
 if event.button() == Qt.MouseButton.LeftButton:  
 self.clicked = False  
 if not self.level\_over: self.restart\_level()  
 self.update()  
 super(LevelWindow, self).mouseReleaseEvent(event)  
  
 # Отслеживание того что мышь передвигается  
 def mouseMoveEvent(self, event):  
 super(LevelWindow, self).mouseMoveEvent(event)  
 if self.clicked:  
 self.mouse\_pos\_end\_list.append(event.pos())  
 self.do\_core\_operations()  
 if not self.level\_over:  
 self.update()  
 else:  
 self.update()  
 self.mouse\_pos\_end\_list.clear()  
  
 # Функция отрисовки перемещения мыши  
 def paintEvent(self, paint\_event):  
 super(LevelWindow, self).paintEvent(paint\_event)  
 if self.clicked:  
 content\_painter = QPainter(self.level\_content)  
 # Antialiasing on  
 content\_painter.setRenderHint(QPainter.RenderHint.Antialiasing)  
 color = QColor(255, 0, 0)  
 pen = QPen(Qt.PenStyle.SolidLine)  
 pen.setColor(color)  
 content\_painter.setPen(pen)  
 if not self.level\_over:  
 content\_painter.drawLine(self.mouse\_pos\_start, self.mouse\_pos\_end\_list[0])  
 for i in range(1, len(self.mouse\_pos\_end\_list) - 1):  
 content\_painter.drawLine(self.mouse\_pos\_end\_list[i], self.mouse\_pos\_end\_list[i + 1])  
 content\_painter.end()  
  
 # Функция нахождения пройденных вершин и ребер графа + передача оных в Level и LevelPainter  
 def do\_core\_operations(self):  
 # Try to find vertex in level  
 level = self.level\_content.level  
 index = self.level\_content.level.vertex\_exists(  
 self.mouse\_pos\_end\_list[len(self.mouse\_pos\_end\_list) - 1].x() - 300 - 66 \* 3 + 100,  
 self.mouse\_pos\_end\_list[len(self.mouse\_pos\_end\_list) - 1].y() - 216)  
 if index != -1:  
 level.previous\_painted\_vertex = level.current\_painted\_vertex  
 if index != level.current\_painted\_vertex:  
 level.current\_painted\_vertex = index  
 new\_edge = sorted((level.previous\_painted\_vertex, level.current\_painted\_vertex))  
 if level.edge\_exists(new\_edge):  
 new\_edge = self.level\_content.level.vertices\_positions[new\_edge[0]] \  
 + self.level\_content.level.vertices\_positions[new\_edge[1]]  
 if not (new\_edge in level.bypassed\_edges):  
 level.bypassed\_edges.append(new\_edge)  
 else:  
 self.restart\_level()  
  
 # Условие окончания уровня  
 if len(level.bypassed\_edges) == len(level.edges\_list):  
 self.level\_over = True  
  
 # Функция обнуления уровня  
 def restart\_level(self):  
 self.level\_over = False  
 self.clicked = False  
 level = self.level\_content.level  
 self.mouse\_pos\_start = 0  
 self.mouse\_pos\_end\_list.clear()  
 level.previous\_painted\_vertex = -1  
 level.current\_painted\_vertex = -1  
 level.bypassed\_edges.clear()

**Level\_indicator.py**

from PyQt5.QtWidgets import QLabel, QWidget, QHBoxLayout  
import core.GLOBAL  
  
# Индикатор уровня. Описание и применение аналогично индикатору сложности. За исключением лишь того, что номер уровня  
# берется из ядра  
class LevelIndicator:  
 def \_\_init\_\_(self):  
 self.main\_widget = QWidget()  
 self.main\_widget.setFixedSize(504, 163)  
 self.main\_widget.setStyleSheet('background: rgba(6, 6, 6, 0.08);\  
 border-radius: 60px;')  
 self.elements\_layout = QHBoxLayout()  
 self.ellipse = QLabel()  
 self.ellipse.setFixedSize(111, 111)  
 self.ellipse.setStyleSheet('background:rgba(6, 6, 6, 0.76);\  
 border-radius: 55%;')  
 self.rectangle\_with\_level\_number = QLabel()  
 self.rectangle\_with\_level\_number.setText(f'УРОВЕНЬ {core.GLOBAL.CURRENT\_LEVEL\_NUMBER.text}')  
 self.rectangle\_with\_level\_number.setStyleSheet("background: rgba(6, 6, 6, 0.76);\  
 font-size:36px;\  
 color:white;\  
 border-radius: 50px;")  
  
 self.rectangle\_with\_level\_number.setFixedSize(309, 111)  
 self.elements\_layout.addWidget(self.ellipse)  
 self.elements\_layout.addWidget(self.rectangle\_with\_level\_number)  
 self.main\_widget.setLayout(self.elements\_layout)

**Level\_painter.py**

from PyQt5.QtWidgets import QMainWindow, QLabel  
from PyQt5.QtGui import QPainter, QColor, QPen  
from PyQt5.QtCore import QPoint, Qt  
  
from core.levels import \*  
import core.GLOBAL  
  
# Класс отрсиовщика уровня. Используется для первоначальной (фоновой) отросвки графа-уровня  
class LevelPainter(QMainWindow):  
 def \_\_init\_\_(self, number):  
 super().\_\_init\_\_()  
 self.setContentsMargins(66, 0, 66, 0)  
 self.level\_number = number  
 self.dot\_radius = 11 # радиус отрисовываемых точек  
 self.line\_width = 1 # ширина длинны  
 self.horizontal\_margin = 300  
 self.vertical\_margin = 56  
 self.level = None # Placeholder  
 self.level\_content\_holder = QLabel()  
 self.level\_content\_holder.setStyleSheet('background: rgba(0, 0, 0, 0.05);border-radius: 92px;\  
 OneMorePoint{background:rgba(6, 6, 6, 0.76);\  
 border-radius: 55%};')  
 self.setCentralWidget(self.level\_content\_holder)  
 self.get\_and\_set\_level()  
  
 # Вызов операции получения уровня  
 def get\_and\_set\_level(self):  
 self.level = self.get\_level\_content()  
  
 # Запуск отрисовки фона (графа). Выполняет каждый раз при вызове self.update()  
 def paintEvent(self, event):  
 content\_painter = QPainter(self) # Создание экземпляра "отрисовщика"  
 # Antialiasing on  
 content\_painter.setRenderHint(QPainter.RenderHint.Antialiasing)  
 # Установка цвета для отрисовки линий и вершин  
 color = QColor(3, 3, 3)  
 color.setAlpha(200)  
 pen = QPen(Qt.PenStyle.SolidLine)  
 pen.setColor(color)  
 content\_painter.setPen(pen)  
 content\_painter.setBrush(color)  
 vertices\_list = self.level.vertices\_positions  
 edges\_list = self.level.edges\_positions\_list  
 bypassed\_edges\_list = self.level.bypassed\_edges  
 for vertex in vertices\_list:  
 content\_painter.drawEllipse(QPoint(vertex[0] + self.horizontal\_margin, vertex[1] + self.vertical\_margin),  
 self.dot\_radius, self.dot\_radius)  
 pen.setWidth(self.line\_width)  
 content\_painter.setPen(pen)  
 for edge in edges\_list:  
 content\_painter.drawLine(edge[0] + self.horizontal\_margin, edge[1] + self.vertical\_margin, edge[2]  
 + self.horizontal\_margin, edge[3] + self.vertical\_margin)  
 # Установка красного цвета для отрисовки пройденных ребер графа  
 color.setRgb(255, 0, 0)  
 pen.setColor(color)  
 content\_painter.setPen(pen)  
 for edge in bypassed\_edges\_list:  
 content\_painter.drawLine(edge[0] + self.horizontal\_margin, edge[1] + self.vertical\_margin, edge[2]  
 + self.horizontal\_margin, edge[3] + self.vertical\_margin)  
  
 content\_painter.end() # Обязательное заверещение цикла отрисовки  
  
 def get\_level\_content(self):  
 needed\_level\_name = f'{core.GLOBAL.DIFFICULTY.text}\_{self.level\_number}'  
 match needed\_level\_name:  
 case 'EASY\_1':  
 needed\_level = EASY\_1  
 case 'EASY\_2':  
 needed\_level = EASY\_2  
 case 'EASY\_3':  
 needed\_level = EASY\_3  
 case 'EASY\_4':  
 needed\_level = EASY\_4  
 case 'EASY\_5':  
 needed\_level = EASY\_5  
 return needed\_level

**Buttons\_layout.py**

from PyQt5.QtWidgets import QGridLayout, QToolButton, QVBoxLayout  
from PyQt5.QtGui import QIcon  
from PyQt5.QtCore import QSize, Qt  
import core.GLOBAL  
from sys import exit  
  
# Файл с содержанием кнопок главного меню. Название отражают содержание  
  
main\_layout = QGridLayout()  
play\_button = QToolButton()  
play\_button.setFixedSize(267, 240)  
play\_button.setStyleSheet('''  
 background: rgba(3, 3, 3, 0.89);  
 border: 1px solid #060606;  
 color:white;  
 font-size: 30px;  
 border-radius: 50px;''')  
play\_button.setIcon(QIcon('UI/MainMenu/elements/play\_button.png'))  
play\_button.setText('ИГРАТЬ')  
play\_button.setToolButtonStyle(Qt.ToolButtonStyle.ToolButtonTextUnderIcon)  
play\_button.setIconSize(QSize(111, 111))  
play\_button.clicked.connect(lambda : send\_change\_signal("LevelSwitcher"))  
  
about\_game\_button = QToolButton()  
about\_game\_button.clicked.connect(lambda : send\_change\_signal('RulesWindow'))  
about\_game\_button.setFixedSize(193, 180)  
about\_game\_button.setText('ОБ ИГРЕ')  
about\_game\_button.setToolButtonStyle(Qt.ToolButtonStyle.ToolButtonTextUnderIcon)  
about\_game\_button.setIcon(QIcon('UI/MainMenu/elements/settings\_button.png'))  
about\_game\_button.setIconSize(QSize(129, 112)) #размер шестеренки  
about\_game\_button.setStyleSheet('''  
 background: #010101;  
 font-size:25px;  
 color: white;  
 border-radius: 50px;  
 ''') #значения текста об игре  
  
next\_button = QToolButton()  
next\_button.clicked.connect(exit)  
next\_button.setIcon(QIcon('UI/MainMenu/elements/exit.png'))  
next\_button.setIconSize(QSize(95, 95))  
next\_button.setFixedSize(95, 95)  
next\_button.setStyleSheet('''color:white;  
 border-radius:47%;  
 background:#000000;  
 font-size:96px;''')  
main\_layout.addWidget(play\_button, 1, 1)  
main\_layout.addWidget(about\_game\_button, 2, 2)  
main\_layout.addWidget(next\_button, 3, 3)  
  
def send\_change\_signal(widget):  
 core.GLOBAL.CURRENT\_WINDOW.setText(widget)

**Main\_menu.py**

from PyQt5.QtWidgets import QMainWindow, QVBoxLayout, QWidget, QLabel  
from PyQt5.QtCore import Qt  
from UI.MainMenu.elements import buttons\_layout  
from PyQt5.QtGui import QPixmap  
  
# Класс главного меню  
class MainMenu:  
 def \_\_init\_\_(self):  
 self.main\_window = QMainWindow()  
 self.main\_window.setWindowTitle('Обведи не отрывая перо')  
 self.central\_widget = QWidget()  
  
 # Объявление подвиджета для правильной отрисовки кнопок над фоном  
 self.sub\_widget = QWidget(parent=self.central\_widget)  
 self.sub\_layout = QVBoxLayout()  
 self.main\_layout = QVBoxLayout()  
  
 self.sub\_layout.addLayout(self.main\_layout)  
 self.feather\_logo = QLabel(parent=self.central\_widget)  
 self.app\_name = QLabel(parent=self.central\_widget)  
 self.main\_layout.addLayout(buttons\_layout.main\_layout)  
  
 self.sub\_widget.setLayout(self.sub\_layout)  
 self.sub\_widget.setFixedSize(957, 593)  
 self.sub\_widget.setStyleSheet('''  
 background: rgba(46, 38, 38, 0.03);  
 border-radius: 68px;  
 ''')  
  
 self.sub\_widget.move(426, 263)  
 self.construct\_logo\_and\_app\_name()  
 self.main\_window.setCentralWidget(self.central\_widget)  
  
 # Создание лого в главном меню  
 def construct\_logo\_and\_app\_name(self):  
 self.feather\_logo.move(26, 31) #расположение пера  
 self.feather\_logo.setPixmap(QPixmap('UI/MainMenu/elements/feather\_logo\_t.png'))  
 self.app\_name.setFixedSize(345, 158) #расположение текста  
 self.app\_name.setPixmap(QPixmap('UI/MainMenu/elements/app\_namee.png'))  
 self.app\_name.setAlignment(Qt.AlignmentFlag.AlignCenter)  
 self.app\_name.move(317, 78)

**Rules\_window.py**

from PyQt5.QtWidgets import QWidget, QLabel, QToolButton, QMainWindow  
from PyQt5.QtGui import QPixmap  
import core.GLOBAL  
from UI.common.ellipse import Ellipse  
  
# Класс окна с правилами  
class RulesWindow(QMainWindow):  
 def \_\_init\_\_(self):  
 super(RulesWindow, self).\_\_init\_\_()  
 self.central\_widget = QWidget()  
 self.sub\_widget = QWidget()  
  
 # Создание таблички ПРАВИЛА ИГРЫ  
 self.rules\_label = QLabel(parent=self.central\_widget)  
 self.rules\_label.setFixedSize(550, 123)  
 self.rules\_label.setText(' ПРАВИЛА ИГРЫ')  
 self.rules\_label.setStyleSheet('''background: rgba(0, 0, 0, 0.83);  
 color:white;  
 font-size: 50px;  
 border-radius: 50px;''')  
 self.rules\_label.move(139, 48)  
  
 # Создание таблички с правилами непосредственно  
 self.rules\_itself = QLabel(parent=self.central\_widget)  
 self.rules\_itself.setText(  
 ''' Правила игры\n заключаются в том, что\n игрок должен обвести\n сложную замкнутую\n фигуру, не отрывая пера\n и не проходить по одним и\n тем же линиям дважды.\n Если же пользователь\n оторвет перо или дважды\n пройдет по той же линии,\n то игра начнется сначала.\n ''')  
 self.rules\_itself.setStyleSheet('''  
 background: rgba(1, 1, 1, 0.69);  
 color:white;  
 font-weight: 400;  
 font-size: 50px;  
 border-radius: 50px;''')  
 self.rules\_itself.setFixedSize(815, 744)  
 self.rules\_itself.move(346, 209)  
  
 # Объявление кнопки возвращения в главное меню  
 self.back\_to\_main\_menu\_button = QToolButton(parent=self.central\_widget)  
 self.back\_to\_main\_menu\_button.setFixedSize(220, 88)  
 self.back\_to\_main\_menu\_button.setText('В ГЛАВНОЕ\nМЕНЮ')  
 self.back\_to\_main\_menu\_button.setStyleSheet('''background: rgba(0, 0, 0, 0.88);  
 border-radius: 37px;  
 font-size: 29px;  
 color: #FFFFFF;''')  
 self.back\_to\_main\_menu\_button.clicked.connect(lambda: return\_to('MainMenu'))  
 self.back\_to\_main\_menu\_button.move(self.rules\_itself.x() + self.rules\_itself.width() + 100, 865)  
 self.setCentralWidget(self.central\_widget)  
  
 # Создание эллипсов  
 self.white\_ellipse\_1 = Ellipse((196, 196, 196, 1), 32)  
 self.white\_ellipse\_1.correctStyleSheet(f'''background: rgba{self.white\_ellipse\_1.color};  
 border-radius: 16px; ''')  
 self.white\_ellipse\_1.setParent(self.central\_widget)  
 self.white\_ellipse\_2 = Ellipse((196, 196, 196, 1), 75)  
 self.white\_ellipse\_2.correctStyleSheet(f'''background: rgba{self.white\_ellipse\_1.color};  
 border-radius: 37px; ''')  
 self.white\_ellipse\_2.setParent(self.central\_widget)  
 self.white\_ellipse\_1.move(629, 119)  
 self.white\_ellipse\_2.move(1057, 229)  
  
  
def return\_to(widget):  
 core.GLOBAL.CURRENT\_WINDOW.text = widget

**Window\_switcher.py**

from PyQt5.QtWidgets import QStackedWidget  
import core.GLOBAL  
from UI.MainMenu.main\_menu import MainMenu  
from UI.LevelWindow.level\_window import LevelWindow  
from UI.LevelSwitcher.level\_switcher import LevelSwitcher  
from UI.RulesWindow.rules\_window import RulesWindow  
  
# Главный класс Ui, отвечающий за плавные переходы между экранными формами  
class WindowSwitcher(QStackedWidget):  
 # Инициализация основных экранных форм  
 def \_\_init\_\_(self):  
 super(WindowSwitcher, self).\_\_init\_\_()  
 self.main\_menu = MainMenu().main\_window  
 self.level\_switcher = LevelSwitcher()  
 self.level\_window = None  
 self.rules\_window = RulesWindow()  
  
 self.addWidget(self.main\_menu)  
 self.addWidget(self.level\_switcher)  
 self.addWidget(self.rules\_window)  
 core.GLOBAL.CURRENT\_WINDOW.valueChanged.connect(self.switch\_window\_to)  
  
 # Функция смены текущего виджета  
 def switch\_window\_to(self, widget):  
 if self.level\_window is not None: self.removeWidget(self.level\_window) # Проверка на наличие окна уровня (требуется для корректной перезаписи)  
 if widget == 'MainMenu':  
 self.setCurrentIndex(0)  
 elif widget == 'LevelSwitcher':  
 self.setCurrentIndex(1)  
 elif widget == 'RulesWindow':  
 self.setCurrentIndex(2)  
 elif widget == 'LevelWindow':  
 self.level\_window = LevelWindow()  
 self.addWidget(self.level\_window)  
 self.setCurrentIndex(3)  
 self.update()