|  |  |
| --- | --- |
|  | УТВЕРЖДАЮ |
|  | Руководитель Учебной практики  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Долженкова М. Л.  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г.  М.П. |

ТЕХНИЧЕСКАЯ ДОКУМЕНТАЦИЯ

Игры «Обведи, не отрывая пера»

|  |
| --- |
| СОГЛАСОВАНО |
| Колледж ВятГУ  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г.  М.П. |
|  |
|  |

**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.

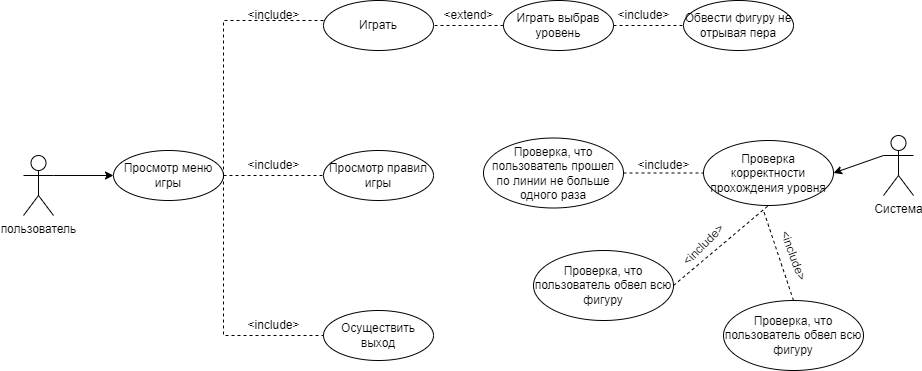


Рисунок 1 – Сценарии взаимодействия с пользователем

1. **Структура программного продукта**

На рисунке 2 представлена диаграмма активности процесса:

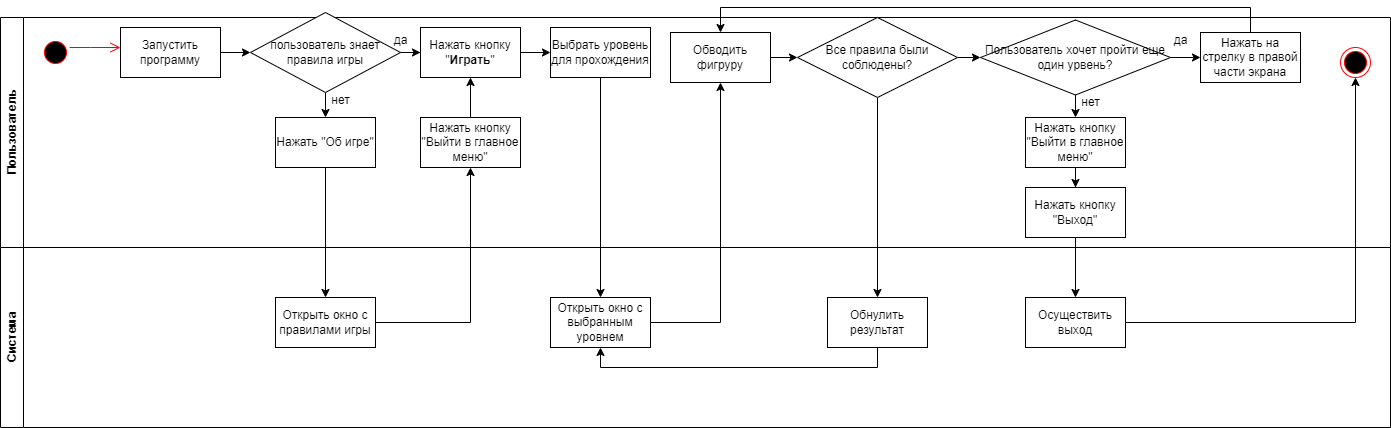


Рисунок 2 – Диаграмма активности процесса

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.

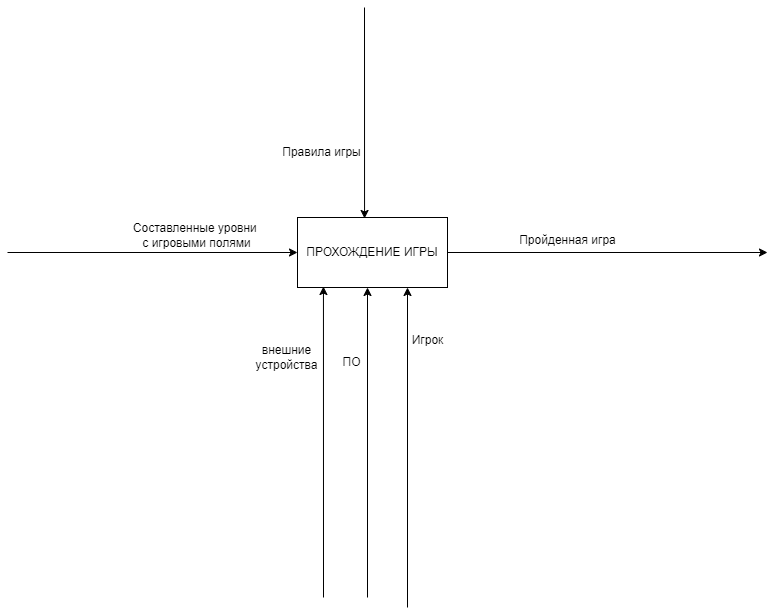


Рисунок 3 – Диаграмма IDEF0

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.

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.

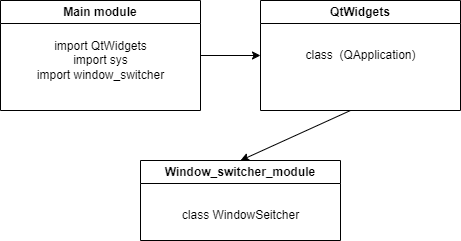


Рисунок 4 – Диаграмма классов

Из предоставленной схемы видно, что главный модуль импортирует модули QtWidgets, sys и window\_switcher. 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.

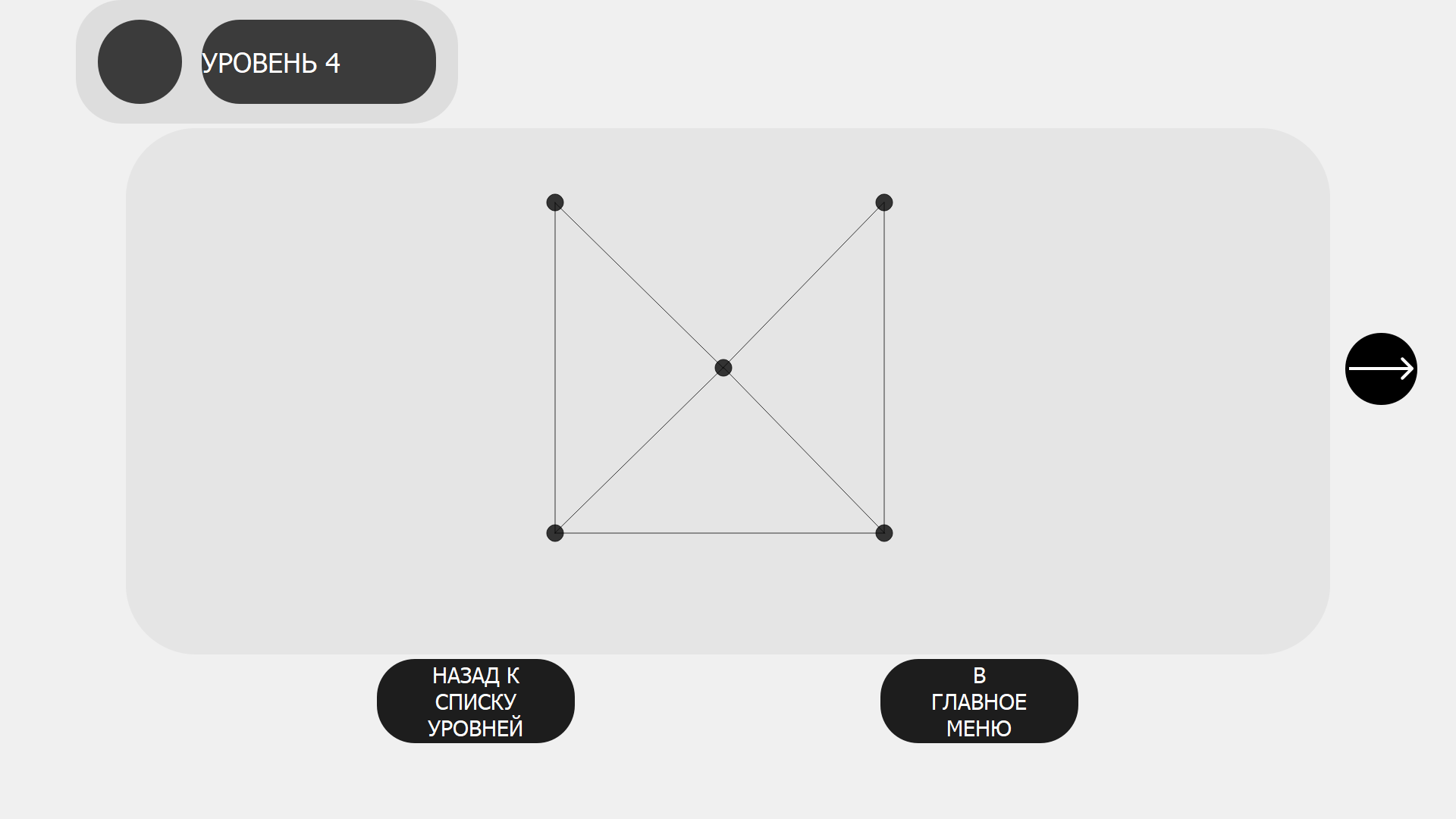


Рисунок 5 – Примерный вид игрового поля

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

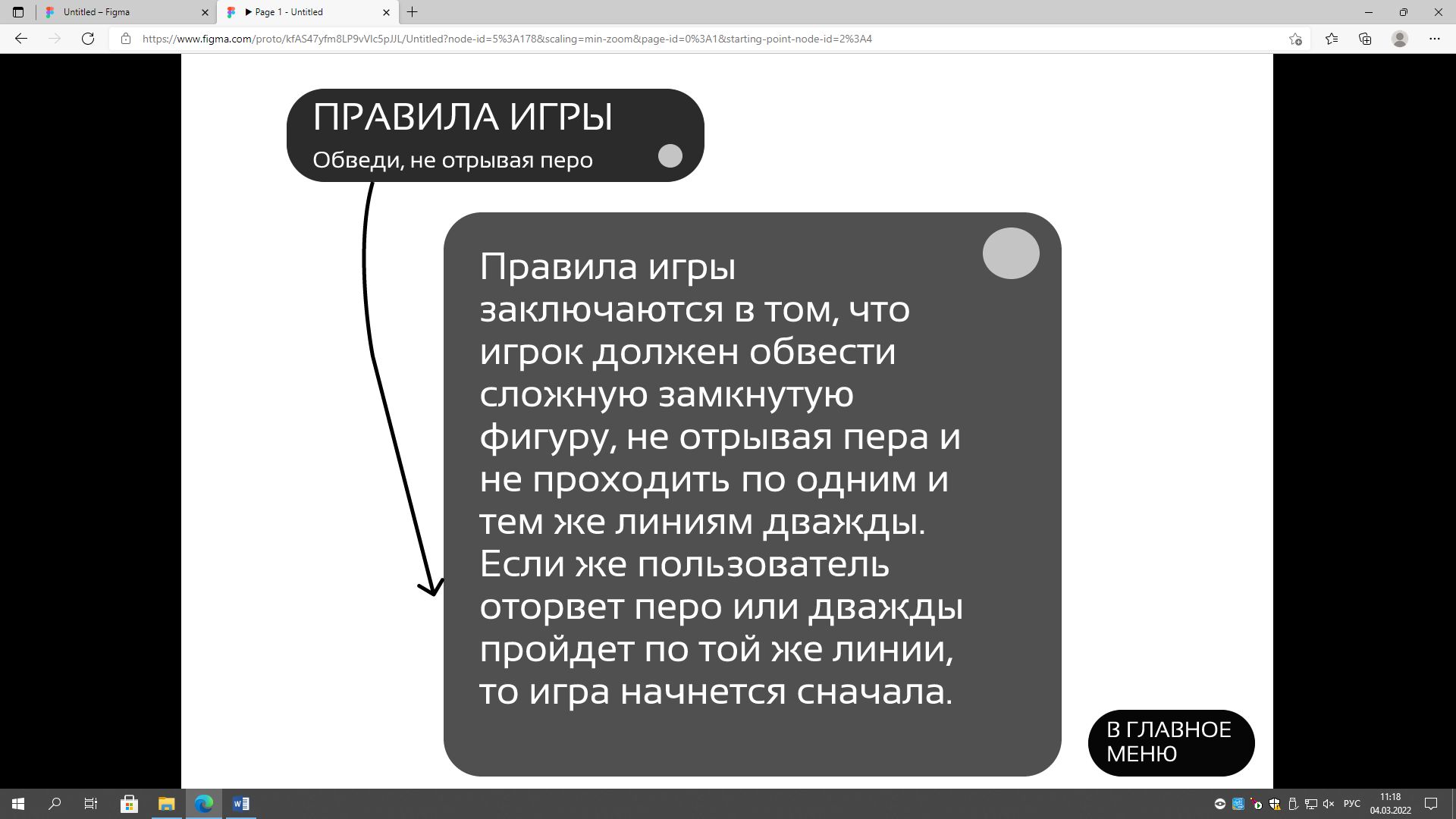


Рисунок 7 – Правила игры

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

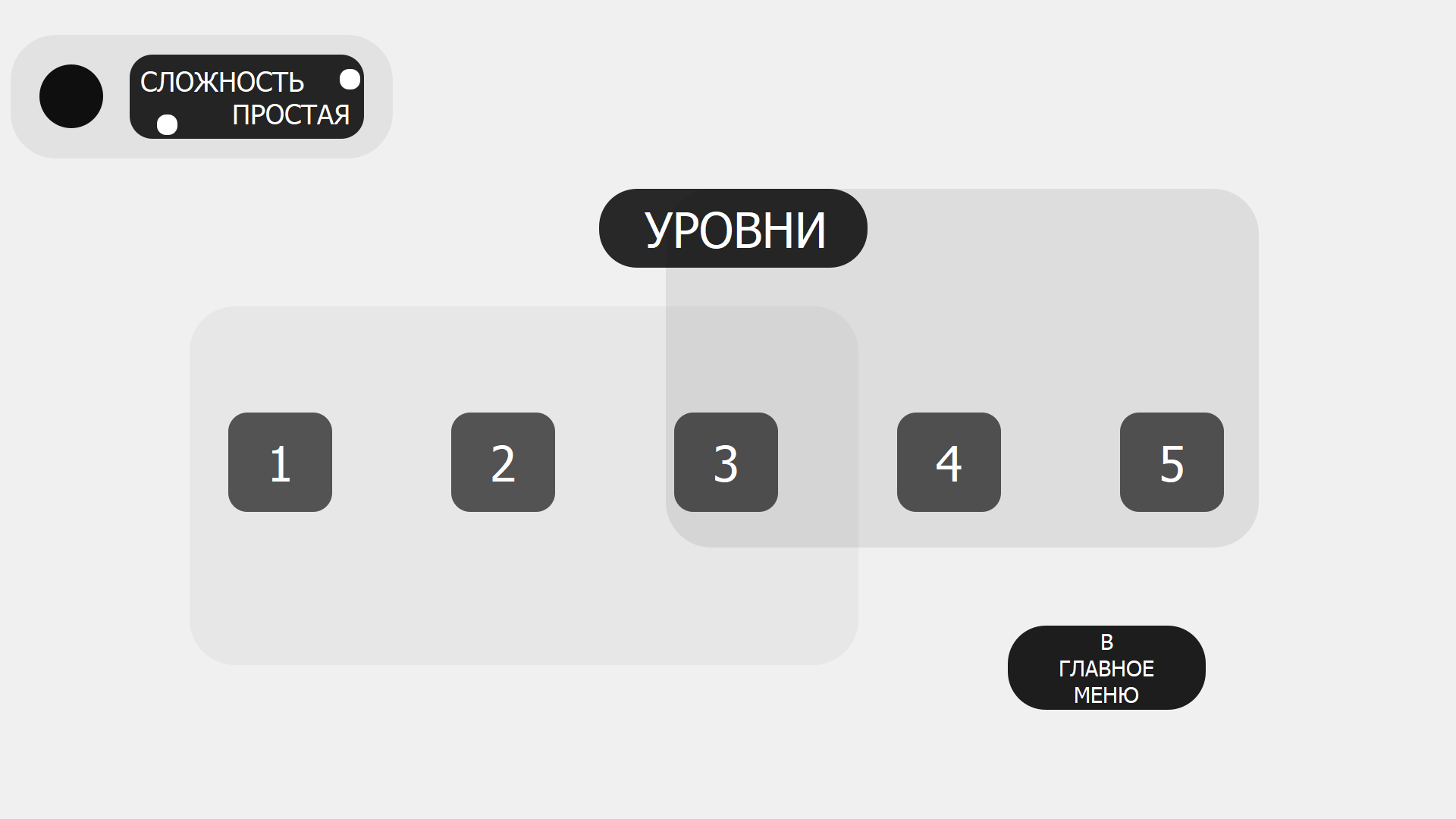


Рисунок 8 – Уровни игры

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.