**1 Анализ задачи**

**1.1 Постановка задачи**

Наименование задачи: “The Binariki”

Цель разработки: повышение уровня знаний учащихся в области арифметико-логических операций через игру и их применения в вычислительной технике через интерактивное обучение.

Назначение: Для студентов, изучающих основы вычислительной техники. Для сотрудников ИТ-отделов, заинтересованных в повышении квалификации. Для образовательных учреждений, включая колледжи и университеты.

Периодичность использования: Регулярное использование в учебных заведениях. Самостоятельное обучение.

Источники и способы получения данных: Учебные материалы по вычислительной технике, конспект по предмету АЛОВТ()

Обзор существующих аналогичных ПП: «The Witness»

**1.2 Функциональные требования**

**Перечень функций:**

**1 для студентов:**

1.1 задания на составление Комбинационных схем и логических функций;

1.2 интерактивные тесты на понимание двоичных систем и логических операций (AND, OR, XOR);

* 1. режим практики с автоматической проверкой.

**2 для преподавателей:**

* 1. инструменты создания новых задач и упражнений;

2.2 мониторинг прогресса учащихся через встроенные отчёты;

Игра-головоломка. Состоит из 10 заданий, при выполнении заданий игрок продвигается на следующий уровень, при неправильном решении загадки поле ввода сбрасывается, а игроку зачисляются штрафные 10 секунд(влияют только на итоговое время)

6 задание

Игроку будет дана 1 часть формулы(какой-нибудь, к примеру (XXX)+(XXX) - 1 скобка - 1часть, 2 часть соответственно) с помощью таблицы истинности он должен будет как по принципу змейки собрать переменные и знаки в нужно порядке

пример

A B F

0 0 0

0 1 0

1 0 0

1 1 1

т.е. игроку надо будет собрать сначала A, потом знак конъюкции(\*), потом B(итоговая формула A\*B)

7 задание

собрав формулу правильно игрок попадет на 2 задание, где ему нужно будет собрать нужные для схемы блоки для 1 части формулы, т.к. в примере 1 задания я написал, что формула A\*B, то ему нужно будет собрать только 1 блок конъюкции, допустим

8 задание

схоже с 1, но вместо таблицы истинности игроку будет дано письменное описание таблицы 2 части формулы по типу:

Имеется 2 сигнала A и B, в результате прохождения этих сигналов 1 приходится на все значения A и B, кроме самого первого(если представить это в виде таблицы истинности, то выйдет

A B F

0 0 0

0 1 1

1 0 1

1 1 1

т.е. A+B)

по этому письменному описанию игрок должен будет как в 1 задании составить формулу

9 задание

так же как во 2 задании игрок должен будет собрать нужные блоки для 2 части формулы

10 задание

собрав 2 части в формулу (ХХХ)+(ХХХ)(т.е. ( (A\*B)+(A+B)) и нужные блоки ему нужно будет соединить эти блоки в правильном порядке, чтобы получить конечный результат, моментами нажимая какую-нибудь клавишу, чтобы раздвоить сигнал(типа некоторые формулы, как допустим A\*A будут требовать 1 сигнал к 1 блоку, который просто раздвоится).  
После прохождения всех заданий игрока перебрасывает в меню. Где отображается время и рекорд прохождения

**1.3 Описание исходной (входной) информации**

Перечень исходной информации:

1. таблицы истинности;
2. базовые учебные материалы по двоичной арифметике и логике;
3. исходные примеры для упражнений.

Формы представления:

1. цифровая таблица (на экране);
2. документ с теоретическим описанием.

Примеры заполнения: Например “Создайте комбинационную схему по заданной логической формуле, учитывая, что все элементы, размещённые игроком, зеркально дублируются."

Пользователи исходной информации:

1. преподаватели;
2. студенты.

**1.4 Описание результатной (выходной) информации**

Перечень результатной информации:

1. результаты тестов и практических заданий;
2. скорость прохождения уровней.

Формы представления:

1. цифровой файл для анализа.

Периодичность и сроки представления:

1. в конце каждой игровой сессии;
2. после прохождения уровня.

Пользователи результатной информации:

1. студенты и преподаватели.

**1.5 Описание используемой условно-постоянной информации**

Перечень:

1. справочники логических операций;
2. таблица формул алгебры-логики;
3. таблица функций алгебры логики;
4. таблица УГО логических элементов.

Форма представления:

1. интерактивное меню внутри игры;
2. справочная информация в виде карточек.

**1.6 Нефункциональные (эксплуатационные требования)**

**Требования к применению:**

* 1. удобный интерфейс с подсказками для начинающих;
  2. интерактивное обучение с примерами и визуализацией.

**Требования к производительности:**

* 1. обработка задач с минимальным временем отклика;
  2. возможность работы на устройствах с низким уровнем производительности.

**Требования к реализации:**

* 1. использование языка программирования C++ с библиотеками для графики;
  2. работа под операционными системами Windows;
  3. десктопное приложения с сохранением данных в облаке.

**Требования к надёжности:**

* 1. cохранение прогресса игроков даже в случае сбоя;
  2. резервное копирование данных.

**Требования к интерфейсу:**

* 1. поддержка взаимодействия с клавиатурой и мышью;
  2. интеграция с внешними учебными платформами.

**2 Проектирование**

**2.1 Выбор стратегии разработки и модели жизненного цикла**

Для разработки приложения «Binariki»» следует выбрать стратегию разработки и модель жизненного цикла. Осуществляем выбор посредством составления таблиц:

Таблица 1 – Выбор модели жизненного цикла на основе характеристик требований

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Критерии категории требований | | Каскадная | | V-образная | | RAD | | Инкрементная | | Быстрого прототипиро вания | | Эволюционная | |
| Являются ли требования к проекту легко определимыми и реализуемыми? | | Да | | Да | | Да | | Нет | | Нет | | Нет | |
| Могут ли требования быть сформулированы в начале ЖЦ? | | Да | | Да | | Да | | Да | | Нет | | Нет | |
| Часто ли будут изменяться  требования на протяжении ЖЦ? | | Нет | | Нет | | Нет | | Нет | | Да | | Да | |
| Нужно ли демонстрировать требования с целью их  определения? | | Нет | | Нет | | Да | | Нет | | Да | | Да | |
| Требуется ли проверка концепции программного средства или системы? | | Нет | | Нет | | Да | | Нет | | Да | | Да | |
| Будут ли требования изменяться или уточняться с ростом сложности системы (программного средства) в ЖЦ? | | Нет | | Нет | | Нет | | Да | | Да | | Дsа | |
| Нужно ли реализовать основные требования на ранних этапах  разработки? | | Нет | | Нет | | Да | | Да | | Да | | Да | |

Вычисления: 3 за каскадную, 3 за V- образную, 4 за RAD, 2 за инкрементную, 4 за быстрого прототипирования и 4 за эволюционную.

Итог: На основе результатов заполнения табл. 3 подходящей является RAD модель и модель быстрого прототипирования и эволюционная модель.

Таблица 2 – Выбор модели жизненного цикла на основе характеристик команды разработчиков

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Критерии категории команды разработчиков  проекта | Каскадная | V-образная | RAD | Инкрементная | Быстрого  прототипирования | Эволюционная |
| Являются ли проблемы предметной области проекта новыми для большинства  разработчиков? | Нет | Нет | Нет | Нет | Да | Да |
| Являются ли инструментальные средства, используемые в проекте, новыми для  большинства разработчиков? | Да | Да | Нет | Нет | Нет | Да |
| Изменяются ли роли участников проекта на протяжении ЖЦ? | Нет | Нет | Нет | Да | Да | Да |
| Является ли структура процесса разработки более значимой для разработчиков, чем гибкость? | Да | Да | Нет | Да | Нет | Нет |
| Важна ли легкость распределения человеческих ресурсов проекта? | Да | Да | Да | Да | Нет | Нет |
| Приемлет ли команда  разработчиков оценки, проверки, стадии разработки? | Да | Да | Нет | Да | Да | Да |

Вычисления: 5 за каскадную, 4 за V-образную, 4 за RAD, 5 за инкрементную, 2 за быстрого прототипирования и 1 за эволюционную.

Итог: На основе результатов заполнения табл. 4 подходящими являются каскадная и инкрементная модели.

Таблица 3 **–** Выбор модели жизненного цикла на основе характеристик коллектива пользователей

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Критерии категории коллектива пользователей | Каскадная | V- образная | RAD | Инкрементная | Быстрого  прототипирова ния | Эволюционная |
| Будет ли присутствие пользователей ограничено в ЖЦ разработки? | Да | Да | Нет | Да | Нет | Да |

Продолжение таблицы 3

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Будут ли пользователи оценивать  текущее состояние программного продукта (системы) в процессе  разработки? | Нет | Нет | Нет | Да | Да | Да |
| Будут ли пользователи вовлечены во все фазы ЖЦ разработки? | Нет | Нет | Да | Нет | Да | Нет |
| Будет ли заказчик отслеживать ход выполнения проекта? | Нет | Нет | Нет | Нет | Да | Да |

Вычисления: 1 за каскадную, 1 за V-образную, 3 за RAD, 0 за инкрементную, 3 за быстрого прототипирования и 1 за эволюционную.

Итог: На основе результатов заполнения табл. 5 подходящей является модель быстрого проектирования и RAD модель.

Таблица 4 – Выбор модели жизненного цикла на основе характеристик типа проектов и рисков

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Критерии категории типов проекта и рисков | Каскадная | V- образная | RAD |  | Быстрого прототипиро вания | Эволюционная |
| Разрабатывается ли в проекте  продукт нового для организации направления? | Нет | Нет | Нет | Да | Да | Да |
| Будет ли проект являться расширением существующей системы? | Да | Да | Да | Да | Нет | Нет |
| Будет ли проект крупно- или среднемасштабным? | Нет | Нет | Нет | Да | Да | Да |
| Ожидается ли длительная эксплуатация продукта? | Да | Да | Нет | Да | Нет | Да |
| Необходим ли высокий уровень надежности продукта проекта? | Нет | Да | Нет | Да | Нет | Да |
| Предполагается ли эволюция  продукта проекта в течение ЖЦ? | Нет | Нет | Нет | Да | Да | Да |
| Велика ли вероятность изменения системы (продукта) на этапе  сопровождения? | Нет | Нет | Нет | Да | Да | Да |
| Является ли график сжатым? | Нет | Нет | Да | Да | Да | Да |
| Предполагается ли повторное использование компонентов? | Нет | Нет | Да | Да | Да | Да |
| Являются ли достаточными  ресурсы (время, деньги,  инструменты, персонал)? | Нет | Нет | Нет | Нет | Да | Да |

Вычисления: 4 за каскадную, 3 за V-образную, 3 за RAD, 5 за инкрементную, 7 за быстрого прототипирования и 7 за эволюционную.

Итог: На основе результатов заполнения табл. 6 подходящей является быстрого прототипирования и эволюционная модели.

Общий итог: в итоге заполнения табл. 3 – 6 наиболее подходящей является эволюционная модель

* 1. **Диаграмма вариантов использования**

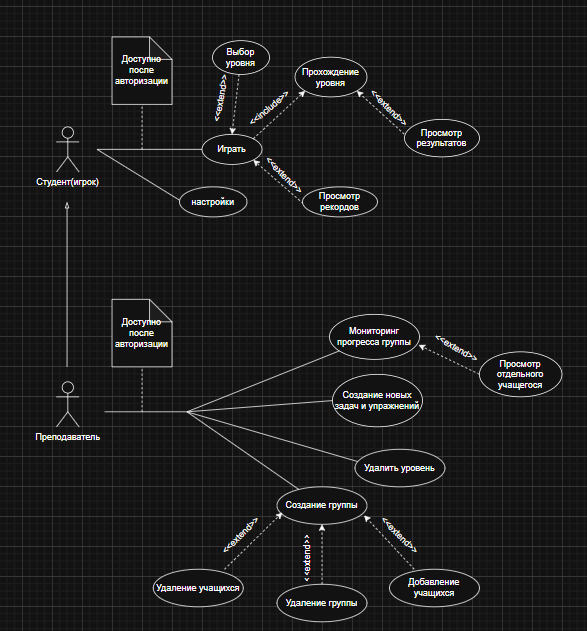
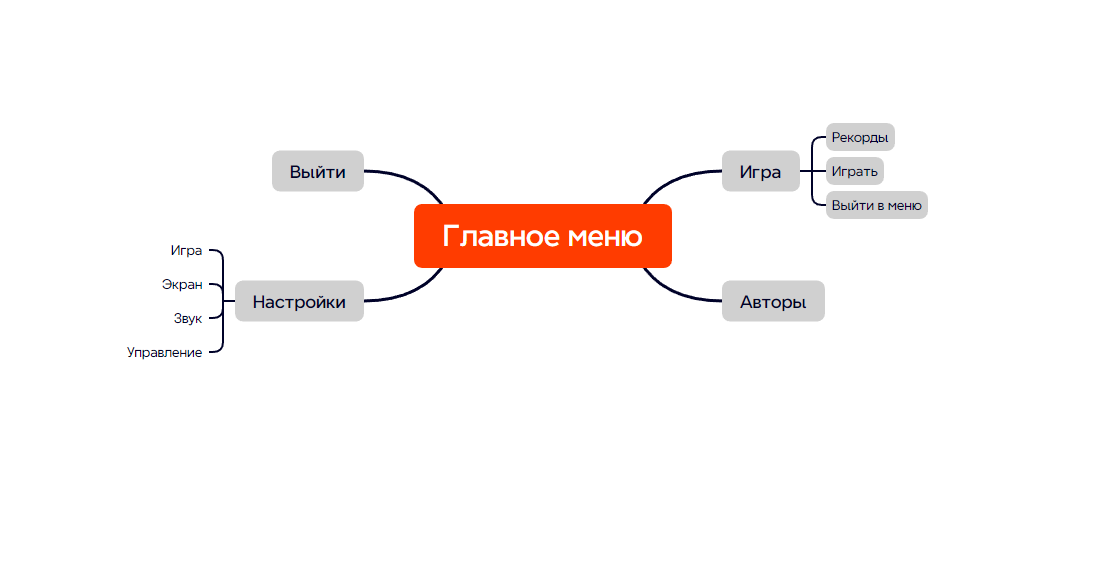


Рисунок 1 – Диаграмма вариантов использования

* 1. **Проектирования системы главного меню**

Рисунок 2 – Система главного меню

* 1. **Моделирование бизнес-процессов**

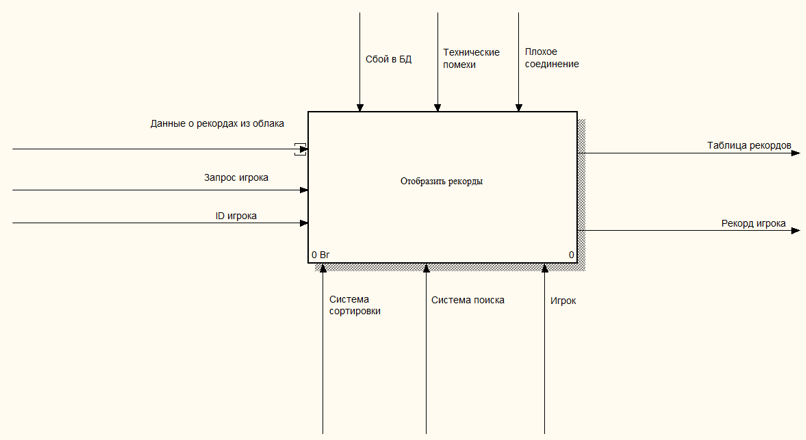
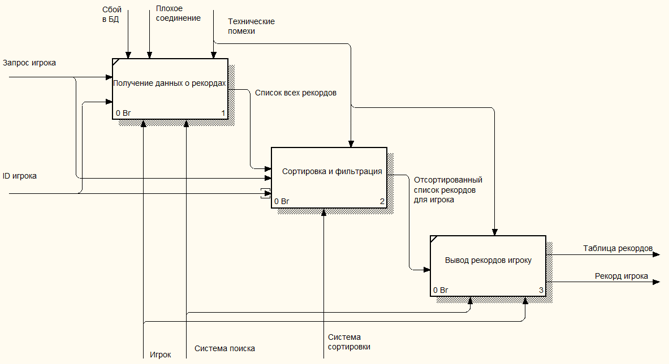
****

Рисунок 3 – Блок А0



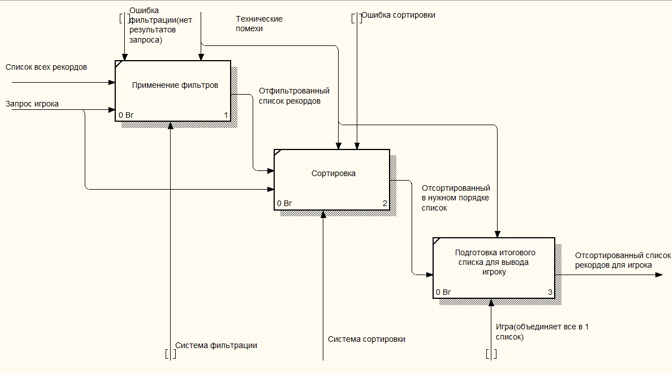
Рисунок 4 – Блоки А1, А2, А3

Рисунок 5 – Блоки А2.1, А2.2, А2.3

* 1. **Моделирование данных**

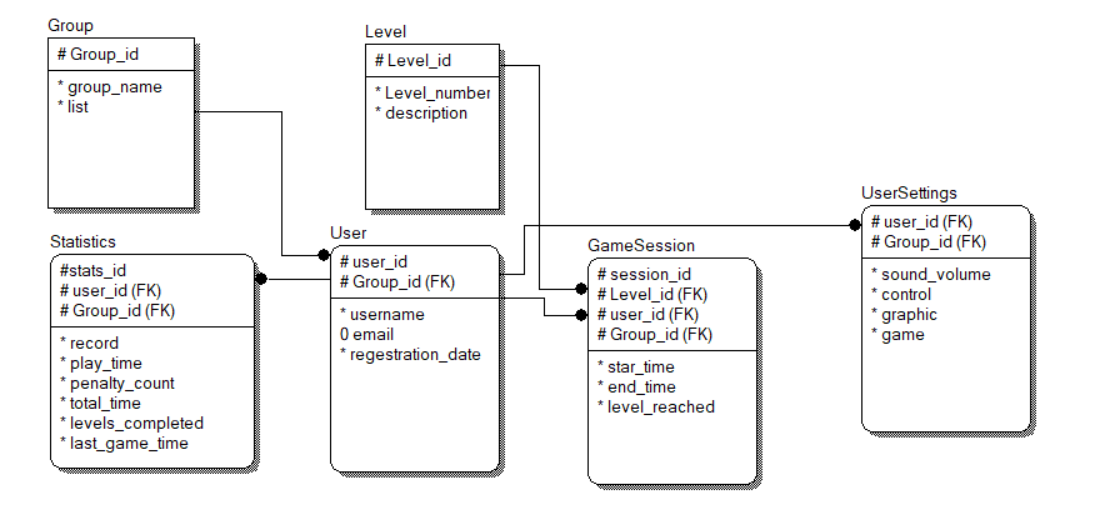
****

Рисунок 6 – Модель данных

**2.6 Диаграмма последовательности**

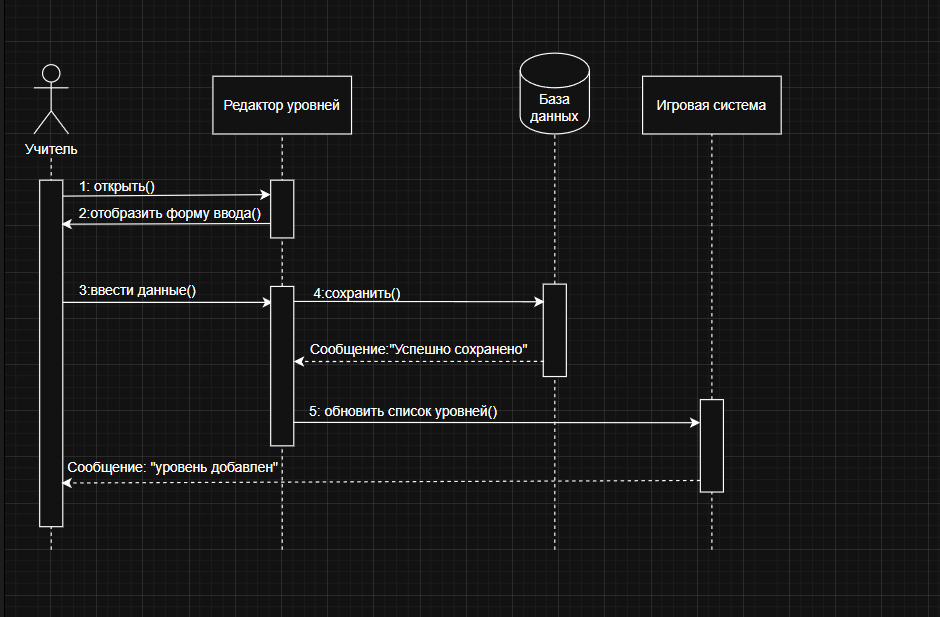


Рисунок 7 – Диаграмма последовательности

**2.7 Диаграмма деятельности**

Диаграмма деятельности отражена в приложении А

**2.8 Диаграмма классов**

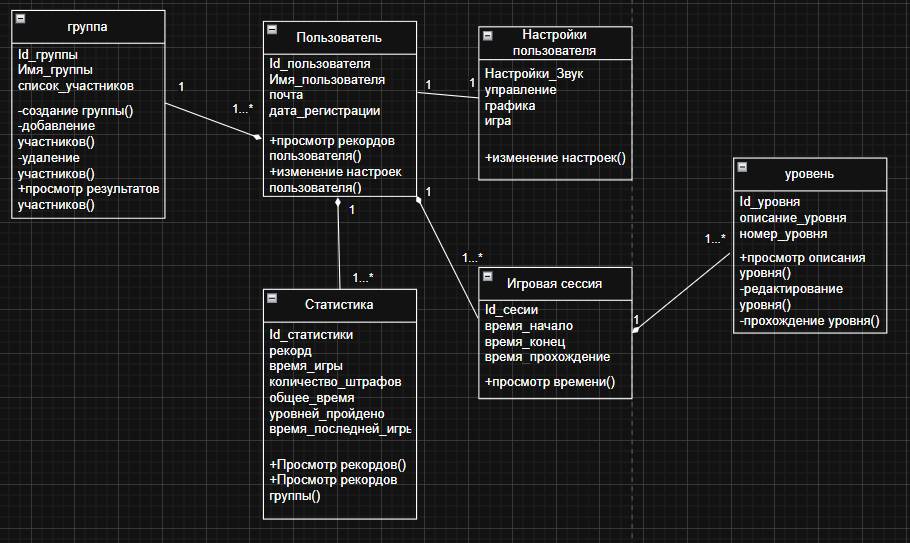
****

Рисунок 8 – Диаграмма классов

**2.9 Диаграмма объектов**

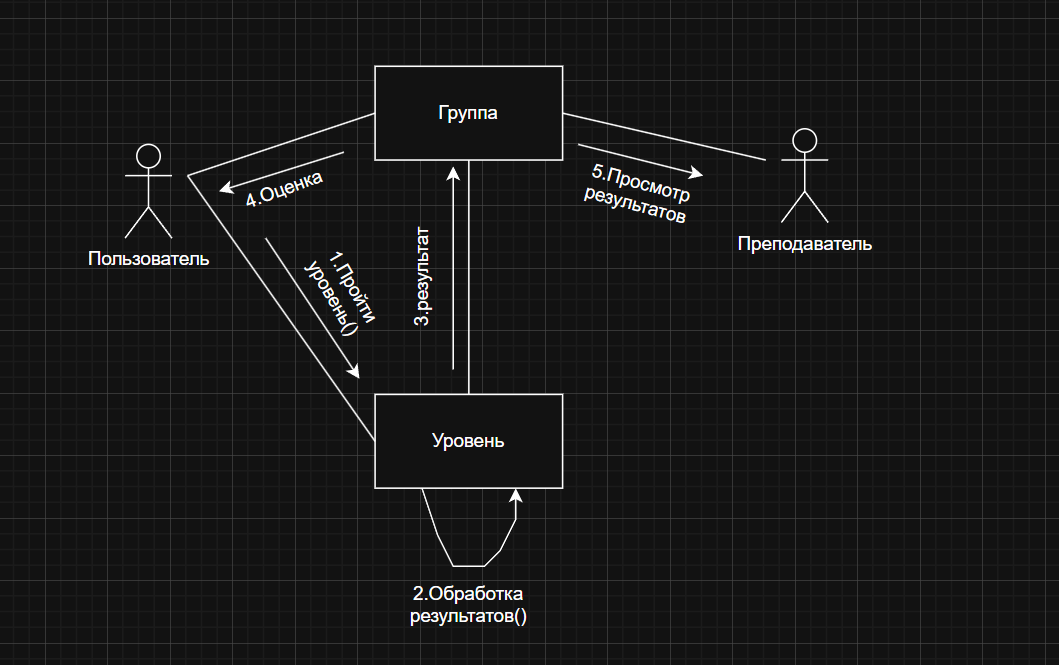


Рисунок 9 – Диаграмма объектов