

ФГБОУ ВО Уральский государственный горный университет

Инженерно-экономический факультет

Кафедра информатики

**Курсовой проект**

По дисциплине «Программная инженерия»

На тему «Разработка приложения «**Bomberman**»»

Выполнил:

Студент гр. АУБП-22-1 Романов Н.С.

Проверила:

ст. преп. каф. информатики Волкова Е.А.

Екатеринбург, 2024 г.

**СОДЕРЖАНИЕ**

[1. ПОСТАНОВКА ЗАДАЧИ 3](#_Toc167919625)

[1.1 Формулировка задания «Bomberman» 3](#_Toc167919626)

[1.2. Алгоритмическое разрешение задачи 4](#_Toc167919627)

[1.3. Контрольные примеры 6](#_Toc167919628)

[2. РЕШЕНИЕ ЗАДАЧИ 8](#_Toc167919629)

[2.1. Выбор средств реализации 8](#_Toc167919630)

[2.2. Описание основных классов 9](#_Toc167919631)

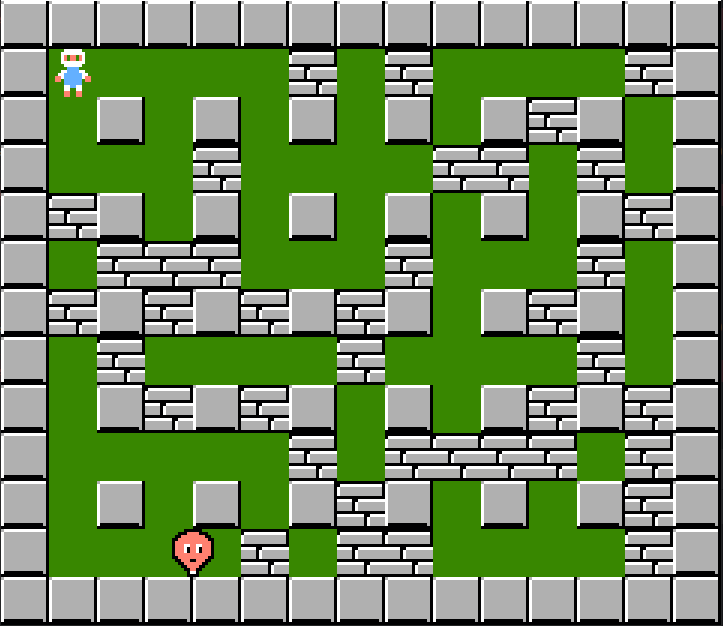
[2.3. Интерфейс приложения 13](#_Toc167919632)

[3. Результаты работы 15](#_Toc167919633)

# **1. ПОСТАНОВКА ЗАДАЧИ**

## **1.1 Формулировка задания «****Bomberman»**

### Реализовать классическую игру «Bomberman». Пример игрового поля:



### Рисунок 1. Пример поля

**Постановка задачи**

Реализовать классическую игру «Bomberman» Задача состоит в разработке приложения на языке Java, которое будет реализовывать основные механики игры "Bomberman": движение персонажа, установка и взрыв бомб, взаимодействие с врагами и препятствиями.

### .

## **1.2. Алгоритмическое разрешение задачи**

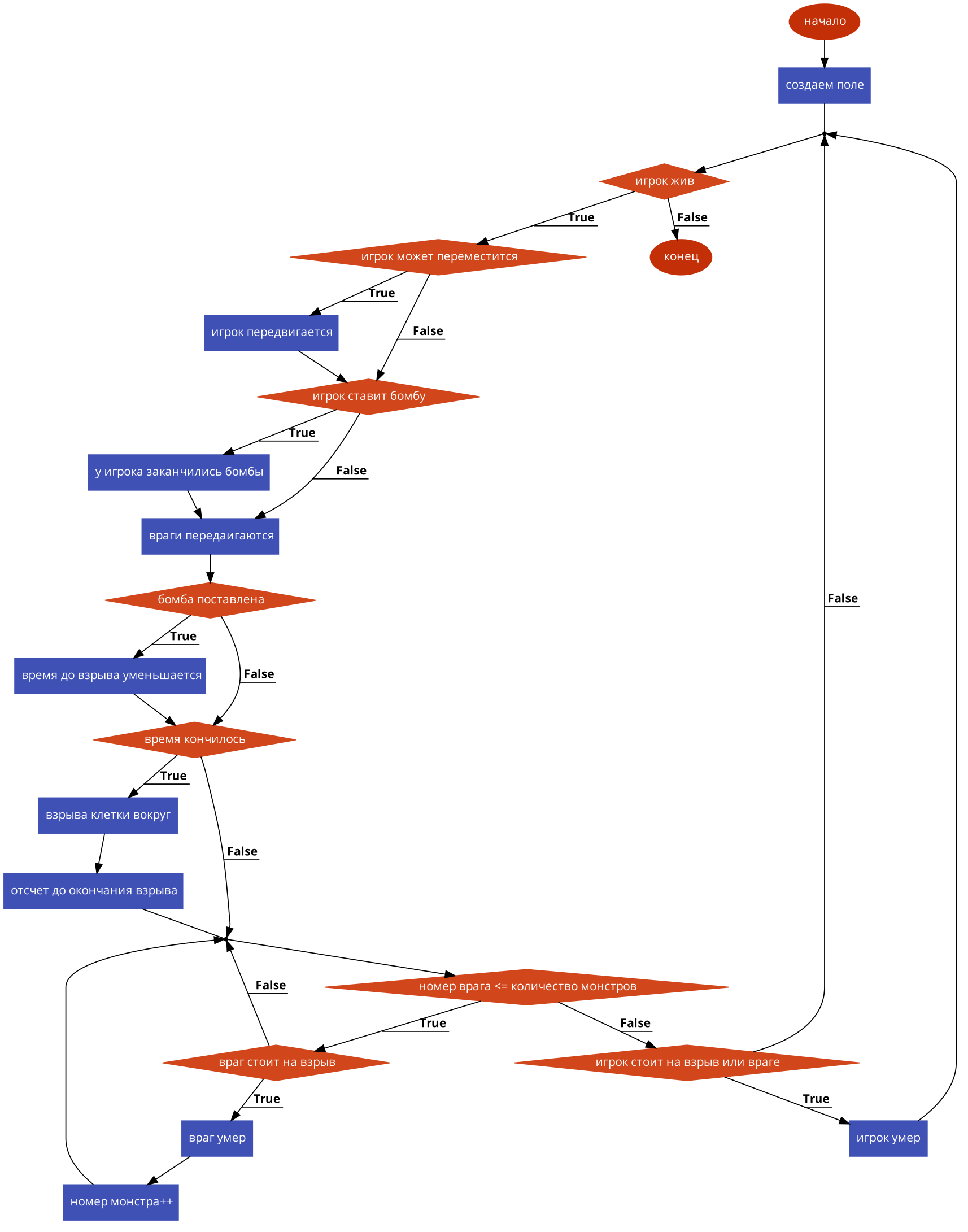
Для реализации игры необходимо разработать алгоритмы, позволяющие:

1.Генерировать игровое поле с препятствиями и врагами.

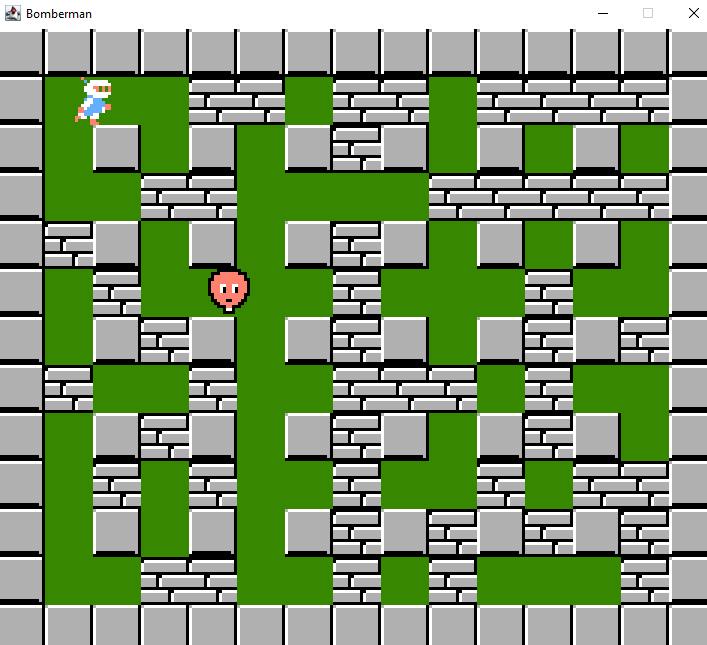
2.Обрабатывать пользовательский ввод для управления персонажем.

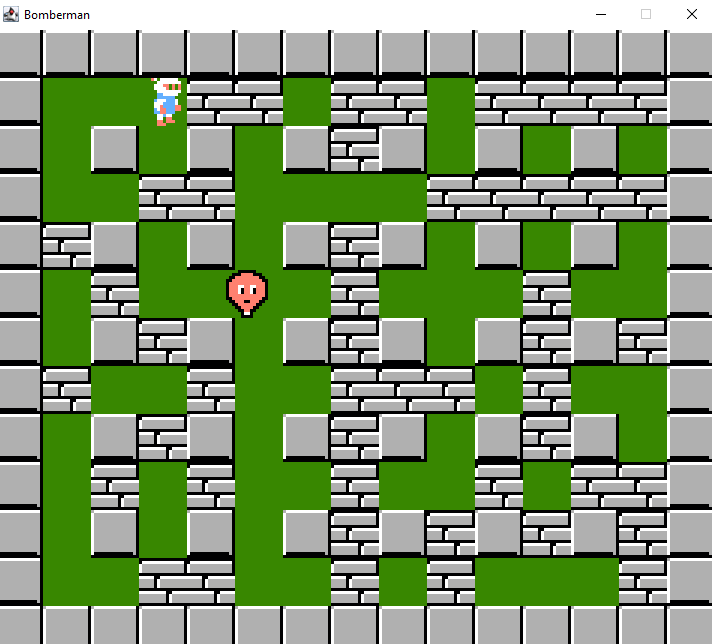
4.Устанавливать и взрывать бомбы.

5. Регистрация урона для врагов и игрока

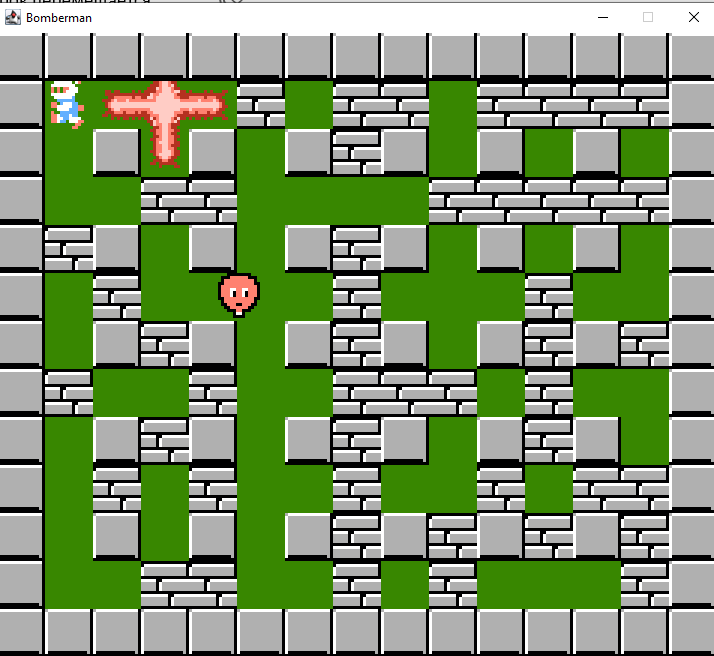
Основной алгоритм игры

## **1.3. Контрольные примеры**



Игрок перемещается  
  


Игрок не может переместится



Игрок поставил бомбу и она взорвала разрушаемые блоки

### 

### Игрок попал на взрыв и умер

# **2. РЕШЕНИЕ ЗАДАЧИ**

## **Выбор средств реализации**

Язык программирования и платформа

Для реализации игры выбран язык программирования Java. Основные причины этого выбора:

Широкие возможности для создания графических интерфейсов: Java предоставляет обширную библиотеку Swing для создания графических элементов, что упрощает разработку визуальных компонентов игры.

Платформонезависимость: Java-приложения могут быть запущены на различных операционных системах без изменений в коде, что делает игру доступной для широкой аудитории.

Широкая поддержка и сообщество: Java имеет большое сообщество разработчиков и множество библиотек, что облегчает решение различных задач и получение поддержки.

Графическая библиотека

Для реализации графической части игры используется библиотека Swing

В качестве IDE было выбрано IntelliJ IDEA. IntelliJ IDEA является одной из самых популярных и мощных интегрированных сред разработки (IDE) для языка Java. Выбор IntelliJ IDEA для разработки игры на Java обоснован следующими преимуществами:

Интеллектуальная поддержка кода: IntelliJ IDEA предоставляет умные подсказки, автозаполнение и рефакторинг кода, что значительно ускоряет разработку.

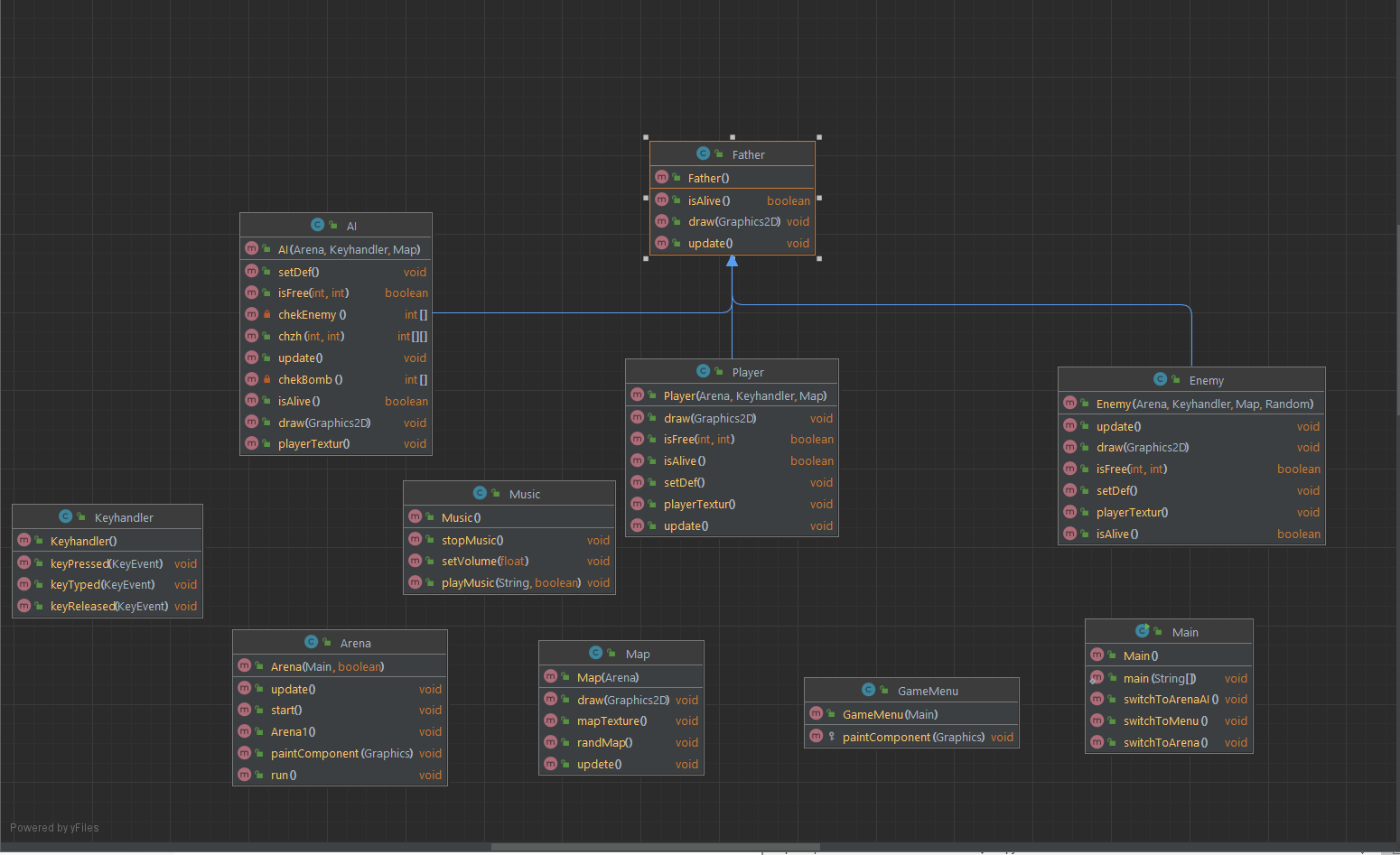
Мощный отладчик: IntelliJ IDEA предоставляет мощные инструменты для отладки, позволяющие устанавливать точки останова, пошагово выполнять код и анализировать состояние программы.

Поддержка графических библиотек: Легкость работы с Java Swing и другими графическими библиотеками.

Загрузка и управление ресурсами: Упрощает работу с изображениями, звуковыми файлами и другими ресурсами, необходимыми для разработки игры.

## Описание основных классов

### При решении задачи было разработано 10 классов показанные ниже (рисунок 5).



### Рисунок 5.

**Класс Main создает и управляет главным окном игры "Bomberman" с использованием Java Swing. Он инициализирует окно (JFrame), устанавливает его параметры (заголовок, невозможность изменения размеров, поведение при закрытии), и добавляет начальное игровое меню (GameMenu). Класс предоставляет методы для переключения между игровым меню и игровой ареной (Arena), с возможностью выбора режима игры с искусственным интеллектом. При переключении, он удаляет текущий компонент из окна и добавляет новый, обновляя интерфейс. Также в методе main запускается музыкальное сопровождение игры и создается экземпляр класса Main в потоке графического интерфейса пользователя.**

**Класс Arena представляет собой игровую арену для игры "Bomberman", реализованную с использованием Java Swing. Он управляет отображением и логикой игры, включая инициализацию карты, игрока (в зависимости от режима - с ИИ или без), и врагов. Класс реализует игровой цикл через метод run, обновляющий состояние игры и перерисовывающий экран с определенной частотой. В зависимости от состояния игрока и врагов, класс обрабатывает проигрыш или переход на следующий уровень. Он также обрабатывает ввод с клавиатуры и воспроизводит соответствующую музыку при завершении уровня или проигрыше.**

**Класс GameMenu создает главное меню для игры "Bomberman" с использованием Java Swing. Он загружает фоновое изображение для меню и размещает три кнопки: "Start Game" для начала обычной игры, "Bot play" для начала игры с ИИ, и "Exit" для выхода из приложения. Кнопки размещены вертикально и центрированы на панели. Каждая кнопка имеет обработчик событий, который переключает игру на соответствующий режим или завершает программу. При перерисовке панели метод paintComponent отображает фоновое изображение, адаптированное к размеру панели.**

**Класс Map отвечает за управление и отрисовку карты в игре "Bomberman". Он инициализирует и случайным образом генерирует карту с различными объектами (стенами, блоками, бомбами), загружает текстуры и анимации для различных элементов карты (блоки, взрывы и бомбы), а также обновляет состояние карты по мере игры. Метод draw отвечает за отрисовку карты и всех её элементов, включая анимацию взрывов. Когда бомба взрывается, она уничтожает соседние блоки и запускает соответствующую анимацию и звуковые эффекты. Метод randMap создает случайную конфигурацию блоков на карте, а метод mapTexture загружает текстуры из файла. update метод на данный момент не реализован.**

**Класс Father определяет общие свойства и методы для всех персонажей. У него есть координаты (x, y), скорость (speed), и флаг жизни (hp). Метод isAlive() возвращает true, если персонаж жив, и false в противном случае.**

**Класс Player наследует от Father и представляет игрового персонажа, управляемого игроком. Он содержит специфические для игрока свойства, такие как клавиатурный обработчик (WASD) и карту (map). Метод update() обновляет состояние игрока в соответствии с пользовательским вводом и текущим состоянием карты. Метод draw() отвечает за отображение изображения игрока на экране. Метод isFree() проверяет, можно ли переместить игрока в заданные координаты, учитывая текущую карту. Метод playerTexture() загружает текстуры игрока из файла.**

**Класс Enemy представляет вражеский персонаж в игре "Bomberman". Он также наследует общие свойства и методы от класса Father, что позволяет ему использовать методы проверки жизни и свойства, такие как координаты и скорость.**

**Конструктор Enemy принимает ссылки на объекты Arena, Keyhandler и Map, а также объект Random, который используется для генерации случайных чисел. Враги движутся случайным образом, меняя направление движения в зависимости от своей текущей позиции и обстановки на карте.**

**Метод isFree() проверяет, является ли следующая позиция для врага свободной от препятствий. Метод setDef() устанавливает начальные координаты и скорость для врага.**

**Метод update() обновляет состояние врага, перемещая его в соответствии с текущим направлением движения. Если враг находится на клетке с бомбой (3), то он стоит на месте, чтобы избежать взрыва. Если враг находится на клетке с огнем (4), то его жизненные показатели (hp) устанавливаются в false, и он "убивается".**

**Метод draw() отвечает за отображение врага на экране, используя текстуры изображения в зависимости от текущего направления движения.**

**Класс AI представляет искусственный интеллект (ИИ) для управления персонажем в игре "Bomberman". Он также наследует общие свойства и методы от класса Father.**

**Конструктор AI принимает ссылки на объекты Arena, Keyhandler и Map. Он также инициализирует начальные координаты и скорость для персонажа ИИ.**

**Методы chekEnemy() и chekBomb() ищут врагов и бомбы на карте, чтобы принять решение о действиях ИИ.**

**Метод chzh(int yyy,int xxx) проверяет доступность позиций для перемещения персонажа ИИ.**

**Метод update() обновляет состояние персонажа ИИ. Он перемещает его в направлении врага, если тот находится в пределах видимости. В противном случае персонаж ИИ перемещается случайным образом.**

**Метод draw() отвечает за отображение персонажа ИИ на экране, используя текстуры изображения в зависимости от его текущего направления движения.**

**Метод playerTextur() загружает текстуры для персонажа ИИ из файла изображения.**

**Класс AI представляет искусственный интеллект (ИИ) для управления персонажем в игре "Bomberman". Он также наследует общие свойства и методы от класса Father.**

**Конструктор AI принимает ссылки на объекты Arena, Keyhandler и Map. Он также инициализирует начальные координаты и скорость для персонажа ИИ.**

**Методы chekEnemy() и chekBomb() ищут врагов и бомбы на карте, чтобы принять решение о действиях ИИ.**

**Метод chzh(int yyy,int xxx) проверяет доступность позиций для перемещения персонажа ИИ.**

**Метод update() обновляет состояние персонажа ИИ. Он перемещает его в направлении врага и ставит бомбы. В противном случае персонаж ИИ убегает от них.**

**Метод draw() отвечает за отображение персонажа ИИ на экране, используя текстуры изображения в зависимости от его текущего направления движения.**

**Метод playerTextur() загружает текстуры для персонажа ИИ из файла изображения.**

**Класс Music отвечает за воспроизведение звукового файла в формате WAV. Вот краткое описание его методов:**

**Метод playMusic(String filepath, boolean loop) запускает воспроизведение музыкального файла, указанного по пути filepath. Если параметр loop установлен в true, то воспроизведение будет зациклено. Он также обрабатывает исключения, которые могут возникнуть при попытке воспроизведения.**

**Метод stopMusic() останавливает воспроизведение текущего музыкального файла и закрывает аудиопоток, если он был открыт.**

**Метод setVolume(float x) устанавливает громкость воспроизведения музыки. Параметр x должен находиться в диапазоне от 0 до 1, где 0 - это минимальная громкость, а 1 - максимальная. Этот метод также проверяет и корректирует значение x в допустимом диапазоне и устанавливает соответствующее значение громкости.**

## Интерфейс приложения

Интерфейс приложения представляет собой игровое меню, выполненное в виде графического окна с фоновым изображением и тремя основными кнопками для управления игрой. Вот описание интерфейса:

Основные элементы интерфейса

Фоновое изображение:

На заднем плане отображается изображение, загружаемое из файла menu.jpg.

Фоновое изображение охватывает всю область панели и задает атмосферу меню игры.

Кнопка Start Game:

Кнопка с надписью "Start Game".

При нажатии на эту кнопку происходит переход к игровому режиму с управлением игроком.

Кнопка Bot play:

Кнопка с надписью "Bot play".

При нажатии на эту кнопку происходит переход к игровому режиму с управлением ботом.

Кнопка Exit:

Кнопка с надписью "Exit".

При нажатии на эту кнопку приложение закрывается.



### Рисунок 9. Панель с выбором

А сама игра копирует графику оригинальной игры.

### Рисунок 10. Игровое поле

# **3.** **Результаты работы**

### В результате выполнения курсового проекта было разработано приложение, которое реализует игру «Bomberman» и бота проходящего ее. В данном приложении реализованы основные механики игры, такие как размещение бомб, разрушение стен . Игра предоставляет несколько уровней сложности и генерируемые игровые поля. Бот, встроенный в игру, обладает алгоритмом, который позволяет ему перемещаться по игровому полю, избегать взрывов и размещать бомбы для достижения победы.