Отчёт о проекте: "3D Labyrinth with Shadows and Textures"

1. Общее описание проекта

Проект "3D Labyrinth with Shadows and Textures" представляет собой интерактивную игру-лабиринт с трёхмерной графикой, реализованную с использованием библиотеки OpenGL и FreeGLUT. Игроку предлагается пройти лабиринт от стартовой точки до выхода, управляя персонажем с видом от первого лица. Игра поддерживает выбор сложности (Easy, Medium, Hard), отображение мини-карты, тени, текстуры стен и пола, а также масштабируемый интерфейс. Основная цель проекта — создание функциональной и визуально привлекательной игры с акцентом на оптимизацию и масштабируемость.

Проект разработан с использованием объектно-ориентированного программирования (ООП) на языке C++, что обеспечивает модульность, читаемость и возможность дальнейшего расширения функционала.

2. Структура проекта

Проект состоит из следующих файлов, разделённых по функциональным модулям:

2.1. Исходные файлы (.cpp)

- **main.cpp**:

- Точка входа в программу. Создаёт экземпляр класса Game, инициализирует его и запускает основной цикл GLUT.

- **Game.cpp**:
  - Управляет состоянием игры (GameState: MENU, PLAYING, WIN), размерами окна и основными коллбэками GLUT (отрисовка, изменение размера, ввод с клавиатуры и мыши).

- **Renderer.cpp**:
  - Отвечает за всю графическую часть: рендеринг 3D-сцены, меню, экрана победы, теней, текстур и мини-карты.

- **Maze.cpp**:
  - Загружает лабиринт из PNG-файлов, определяет начальную позицию игрока и выход, генерирует стены.

- **Player.cpp**:
  - Управляет движением игрока (позиция, угол обзора), обработкой столкновений и условием победы.

- **InputHandler.cpp**:
  - Обрабатывает ввод пользователя (клавиатура и мышь), взаимодействие с интерфейсом и перезапуск игры.

2.2. Заголовочные файлы (.h)

- **Game.h**:
    - Описывает класс Game с состоянием игры, размерами окна и методами управления.

- **Renderer.h**:
    - Определяет класс Renderer с методами рендеринга и статическими переменными для текстур и света.

- **Maze.h**:
    - Определяет класс Maze с методами загрузки лабиринта и управления позицией игрока.

- **Player.h**:
    - Описывает класс Player с методами управления движением и проверкой столкновений.

- **InputHandler.h**:
    - Определяет класс InputHandler с методами обработки ввода.

2.3. Ресурсы

- **PNG-файлы**:
    - maze_easy.png, maze_medium.png, maze_hard.png — карты лабиринтов (черные пиксели — стены, белые — старт и выход).
    - wall_texture.png, floor_texture.png — текстуры для стен и пола.

- **stb_image.h**:
    - Библиотека для загрузки изображений в формате PNG.

3. Использованные технологии

Проект построен с использованием следующих технологий:

- **Язык программирования**: C++ (стандарт C++11).
- **Графическая библиотека**: OpenGL (версия совместимая с FreeGLUT).
- **Библиотека управления окнами и вводом**: FreeGLUT (взаимодействие с окном, рендеринг, обработка событий).
- **Библиотека загрузки изображений**: STB Image (stb_image.h) для чтения PNG-файлов.
- **Объектно-ориентированное программирование (ООП)**:
    - Классы с чётким разделением (Single Responsibility Principle).
    - Использование паттерна Singleton для Maze (единственный экземпляр лабиринта).

4. Инструменты разработки

- **Компилятор**: GCC (MinGW) — использовался для компиляции и линковки проекта на Windows.

- **Команда компиляции**:

```
g++ src/main.cpp src/Game.cpp src/Renderer.cpp src/Maze.cpp
src/Player.cpp src/InputHandler.cpp -o main -lopengl32 -lglu32 -
lfreeglut
```

- **Редактор кода**: Любой текстовый редактор (например, Visual Studio Code, Notepad++), предполагается для редактирования файлов.

5. Описание функциональности

5.1. Основные возможности

- **Режимы игры**:
    - Меню выбора сложности (Easy, Medium, Hard).
    - Игровой процесс с видом от первого лица.
    - Экран победы с опциями "Start again?", "Exit", "Go back to the menu".

- **Управление**:
    - W/A/S/D или стрелки — движение вперёд, влево, назад, вправо.
    - Q/E — поворот влево/вправо.
    - M — включение/выключение мини-карты.
    - Мышь — выбор опций в меню и на экране победы.

- **Графика**:
    - 3D-рендеринг лабиринта с текстурами стен и пола.
    - Тени, реализованные через stencil buffer (дополнительный буфер, соответствующий размеру выводимого кадра) и shadow volumes (область трехмерного пространства, полностью заполненную тенью).
    - Мини-карта в левом нижнем углу (масштабируемая, с отступом 1 пиксель от низа).

5.2. Масштабируемость

- Интерфейс (меню, экран победы, мини-карта) адаптируется к изменению размеров окна:
    - Координаты кнопок и текста вычисляются как процент от windowWidth и windowHeight.
    - Мини-карта масштабируется пропорционально ширине окна с сохранением соотношения сторон лабиринта.

5.3. Оптимизации

- **Singleton для лабиринта**: Один экземпляр класса Maze, что минимизирует затраты памяти.

- **Статические методы**: Использование статических методов в Renderer, Player, и InputHandler для упрощения доступа и избежание ненужных экземпляров.

- **Повторное использование текстур**: Текстуры загружаются один раз и хранятся в статических переменных Renderer::wallTexture и Renderer::floorTexture.

- **Эффективная загрузка PNG**: Использование STB Image с однократным определением (#define STB_IMAGE_IMPLEMENTATION) в Maze.cpp.

6. Оптимизации и улучшения

- **Разделение логики**:
    - Код разделён на модули (Game, Renderer, Maze, Player, InputHandler), что упрощает поддержку и расширение.

- **Сброс состояния**:
    - При перезапуске лабиринта через "Start again?" позиция игрока сбрасывается с использованием сохранённых координат (startX, startZ) из Maze, а мини-карта выключается (setMiniMapShown(false)).

- **Масштабируемость интерфейса**:
    - Все элементы интерфейса (кнопки, текст, мини-карта) используют относительные координаты, что позволяет корректно работать при любом размере окна.

- **Обработка ошибок**:
    - При загрузке PNG-файлов и текстур выводятся сообщения об ошибках в консоль, что упрощает диагностику.

7. Потенциальные улучшения

- **Динамическая загрузка ресурсов**: Поддержка загрузки новых лабиринтов и текстур без изменения кода.

- **Конфигурационный файл**: Параметры (размер окна, пути к файлам) можно вынести в отдельный файл.

- **Анимации**: Добавить плавное движение игрока или эффекты освещения.

- **Кроссплатформенность**: Адаптация путей к файлам для работы на Linux/MacOS.

8. Выводы

Проект "3D Labyrinth with Shadows and Textures" демонстрирует успешную реализацию интерактивной 3D-игры с использованием OpenGL и FreeGLUT. Он включает современные подходы к разработке (ООП, модульность, масштабируемость), оптимизирован для производительности и предоставляет гибкость для дальнейших улучшений. Все ключевые требования — выбор сложности, управление, рендеринг с тенями и текстурами, масштабируемый интерфейс — выполнены.