

1 Анализ задачи

1.1 Постановка задачи

Описание предметной области

Разрабатываемая игра *NeverLend* относится к жанру 2D-платформеров, вдохновленных классическими играми, такими как *Super Mario*. Однако в отличие от традиционных представителей жанра, игра предлагает уникальную атмосферу, где главный герой - кот, исследующий загадочный мир.

Основные задачи, подлежащие автоматизации:

- Управление персонажем – перемещение, прыжки, взаимодействие;
- Обработка коллизий между;
- Генерация и загрузка уровней;
- Реализация игровой логики;
- Обработка событий (ввод с клавиатуры, столкновения, сбор бонусов).

В настоящее время подобные задачи решаются с помощью готовых игровых движков (Unity, Godot) или вручную на языках программирования (C++, Python с использованием библиотек, таких как SFML, Pygame). Данный проект реализуется на C++ с SFML, что позволяет глубже изучить основы разработки игр без использования готовых решений.

Периодичность использования программы

Игра предназначена для разового или периодического использования в развлекательных целях. Пользователи могут запускать ее неограниченное количество раз, проходить уровни и улучшать свои результаты.

Преимущества новой разработки

Упрощенный геймплей по сравнению с современными сложными платформерами. Уникальная стилистика и атмосфера. Возможность модификации и расширения (добавление новых уровней, врагов, механик).

Анализ существующих аналогов

Игра	Особенности	Отличия от <i>NeverLend</i>
<i>Super Mario</i>	Классический платформер, линейные уровни, враги с разными паттернами атак.	В <i>NeverLend</i> враги атакуют только при столкновении, более мрачная атмосфера.
<i>Celeste</i>	Сложный платформер с механиками лазания и дашинга.	<i>NeverLend</i> проще в управлении и ориентирована на исследование.
<i>Hollow Knight</i>	Карты с нелинейным миром и боями.	<i>NeverLend</i> линейна и фокусируется на платформинге, а не на сражениях.

Выходная информация

- Игровое поле – графический интерфейс с персонажем, врагами;
- Меню – старт, пауза, завершение игры;
- Экран победы/поражения;

Входная информация

- Управление – клавиатура: стрелки/WASD для движения;
- Конфигурация уровней – загружается из файлов или программно;
- Настройки графики и звука;

Постоянная информация

- Спрайты и текстуры;
- Звуковые эффекты и музыка;
- Параметры уровней;

Требования к функциям

- Загрузка и отображение графики;
- Обработка пользовательского ввода;
- Физика перемещения и гравитации;
- Система коллизий.
- Логика врагов;
- Система подсчета жизней;
- Сохранение/загрузка прогресса;

1.2 Инструменты разработки

Выбор среды разработки

– Язык программирования: C++ (высокая производительность, низкоуровневый контроль).

– Библиотека: SFML (Simple and Fast Multimedia Library) - предоставляет удобный API для работы с графикой, звуком, вводом.

– Среда разработки: Visual Studio Code.

Требования к аппаратному и программному обеспечению

Параметр	Минимальные требования	Рекомендуемые требования
ОС	Windows 7 / Linux	Windows 10/ 11 / Linux
Процессор	1.5 ГГц	2.4 ГГц+
ОЗУ	512 МБ	2 ГБ+
Видеокарта	Интегрированная	Дискретная (OpenGL 3.3+)

Параметр	Минимальные требования	Рекомендуемые требования
Место на диске	100 МБ	500 МБ (для дополнительного контента)

1.3 Требования к приложению

Ограничения

- Игра не поддерживает сетевой режим;
- Уровни загружаются статически;
- Нет редактора уровней, встроенного в игру;

Требования к интерфейсу

- Интуитивное управление;
- Четкие визуальные подсказки;

Средства защиты

- Отсутствует необходимость в сложной защите, так как проект не является коммерческим, а разрабатывается лишь в целях защиты курсовой;
- Возможна простая проверка целостности файлов;

Вывод: Разрабатываемая игра соответствует современным требованиям к 2D-платформерам и предоставляет пользователям простой, но увлекательный геймплей. Использование C++ и SFML обеспечивает хорошую производительность и гибкость в разработке.