**高级语言程序设计大作业实验报告**

南开大学 计算机学院

姓名：林科融

学号：2413121

班级：计算机卓越班

2025 年 5 月 13 日

高级语言程序设计

实验报告

**目录**

* 高级语言程序设计大作业实验报告 1
* 一.作业题目
* 二.开发软件
* 三.课题要求
* 四.主要流程
  + 1． 整体流程
  + 2． 算法或公式
  + 3． 单元测试
* 五。单元测试
* 六。收获
  + 1． 技术难点与解决
  + 2． 开发过程中的收获

**高级语言程序设计大作业实验报告**

**一。作业题目**

基于 Unity 引擎的 i wanna 风格小游戏开发

**二。开发软件**

Unity 2022.3.21f1c1（可根据实际使用版本填写）

**三。课题要求**

1. 运用 C# 语言进行开发，符合面向对象编程规范。
2. 确保游戏具备完整的功能，包括但不限于角色移动、关卡设计、碰撞检测等。
3. 对游戏核心功能进行单元测试，保证功能的正确性和稳定性。

**四。主要流程**

**1． 整体流程**

游戏采用 MVC（模型 - 视图 - 控制器）架构模式进行开发。

* **模型层**：负责存储游戏数据，如角色的生命值、得分、关卡进度等。创建PlayerModel类，用于管理角色相关数据；LevelModel类管理关卡数据，包括关卡布局、障碍物位置等。
* **视图层**：通过 Unity 的场景搭建和 UI 系统展示游戏画面。利用 Unity 的预制体功能创建角色、障碍物、背景等游戏元素，并使用 Canvas 搭建游戏 UI，显示生命值、得分等信息。
* **控制器层**：处理游戏逻辑和用户输入。编写PlayerController类，监听玩家的输入操作（如键盘的上下左右按键），控制角色的移动；GameController类管理游戏的整体流程，包括关卡的切换、碰撞检测等。  
  在角色移动方面，通过获取玩家输入，修改角色的Transform组件的位置属性实现移动效果。例如，当玩家按下 “→” 键时，角色向前移动，代码大致如下：

csharp

if (Input.GetKey(KeyCode.W))

{

transform.Translate(Vector3.forward \* speed \* Time.deltaTime);

}

在关卡设计上，将每个关卡的地图数据存储在文本文件中，在游戏加载时读取文件，根据数据生成相应的关卡布局，包括放置障碍物、设置目标点等。

**2． 算法或公式**

* **碰撞检测算法**：利用 Unity 内置的碰撞检测系统，通过OnCollisionEnter和OnTriggerEnter函数实现。例如，当角色与障碍物发生碰撞时，减少角色生命值，代码大致如下：

csharp

private void OnCollisionEnter(Collision collision)

{

if (collision.gameObject.CompareTag("Obstacle"))

{

playerModel.DecreaseHealth(1);

}

}

* **得分计算**：玩家每通过一个关卡或完成特定任务获得一定分数，得分计算公式为：总得分 = 基础得分 + 关卡得分 + 额外奖励得分。其中，基础得分根据玩家完成游戏的时间计算，时间越短得分越高；关卡得分根据关卡难度设定；额外奖励得分则根据玩家在游戏中的特殊表现（如收集隐藏道具）获得。

**3． 单元测试**

针对角色移动、碰撞检测和得分计算等核心功能进行单元测试。

* **角色移动测试**：创建测试场景，设置角色初始位置，模拟玩家输入操作，检查角色是否按照预期移动到指定位置。
* **碰撞检测测试**：在测试场景中放置角色和障碍物，验证角色与障碍物碰撞时生命值是否正确减少。
* **得分计算测试**：模拟玩家完成不同任务的情况，检查得分计算是否正确。

**五。单元测试**

**测试案例定义**

| **输入** | **输出** | **目的** |
| --- | --- | --- |
| 模拟按下 “→” 键 | 角色向前移动指定距离 | 测试角色移动功能 |
| 使角色与障碍物碰撞 | 角色死亡 | 测试碰撞检测功能 |
| 模拟玩家完成关卡 | 感谢游玩 | 测试结束功能 |

**测试结果**

在最初的测试中，角色移动测试出现问题，角色移动速度不稳定。经分析，是因为在计算移动距离时未考虑Time.deltaTime，导致移动速度受游戏帧率影响。修改代码后重新测试，所有测试用例均通过，游戏核心功能运行稳定。

**六。收获**

**1． 技术难点与解决**

在开发过程中，遇到了碰撞检测不准确的问题。由于碰撞体的参数设置不合理，导致有时角色与障碍物重叠却未触发碰撞事件。通过仔细调整碰撞体的大小、形状和位置，并参考 Unity 官方文档中关于碰撞检测的相关内容，最终解决了该问题。

**2． 开发过程中的收获**

通过本次项目开发，对 Unity 引擎和 C# 语言有了更深入的理解和掌握。学会了如何使用 Unity 的各种组件和工具进行游戏开发，如预制体、Animator、UI 系统等。同时，在面向对象编程方面也有了很大的提升，能够更好地设计和组织代码结构，提高代码的可维护性和扩展性。此外，单元测试的实践让我认识到其在保证代码质量方面的重要性，能够帮助快速定位和解决代码中的问题。