南开大学 软件学院

姓名 谢卫超

学号 2313670

班级 四班

2025年5月14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 4](#_Toc198127351)

[**一.** **作业题目** 4](#_Toc198127352)

[**二.** **开发软件** 4](#_Toc198127353)

[**三.** **课题要求** 4](#_Toc198127354)

[**四.** **主要流程** 4](#_Toc198127355)

[**1 . 设计思想** 4](#_Toc198127356)

[采用全局对象化的思路，将所有组件封装为对象。每一个对象的基本方法为draw方法，通过数组存储对象位置，通过对象之间距离判断交互。 4](#_Toc198127357)

[**2.** **整体流程** 4](#_Toc198127358)

[**类图：** 4](#_Toc198127359)

[5](#_Toc198127360)

[6](#_Toc198127361)

[**五.** **收获** 8](#_Toc198127362)

高级语言程序设计大作业实验报告

1. **作业题目**

imSheep（我是羊）

玩法介绍：

1. 通过在地图上放置草，引导两只羊分别进入随机生成的两个栅栏区域内，以达到通关目的。
2. **开发软件**

Visual Studio 2022

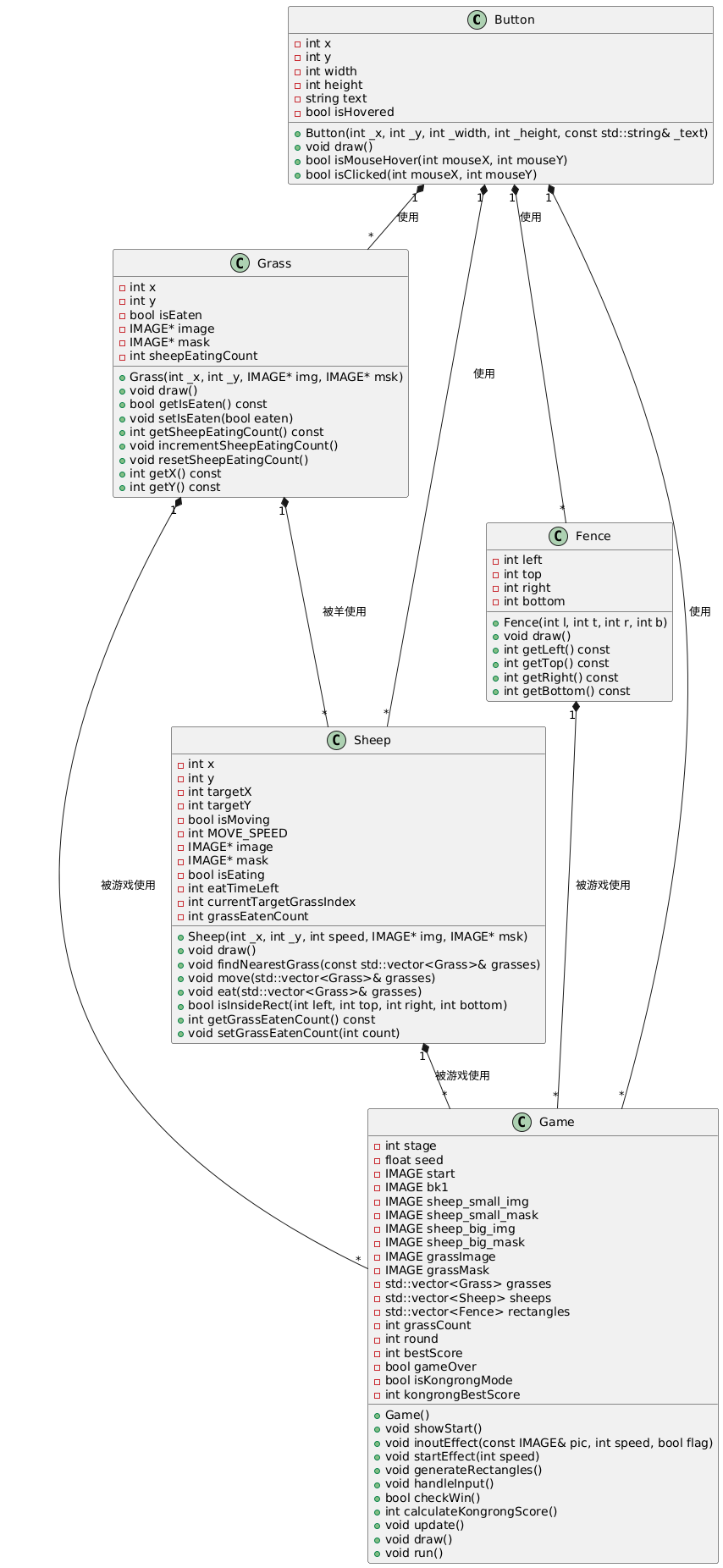
1. **课题要求**
2. 面向对象。
3. 可视化。
4. **主要流程**

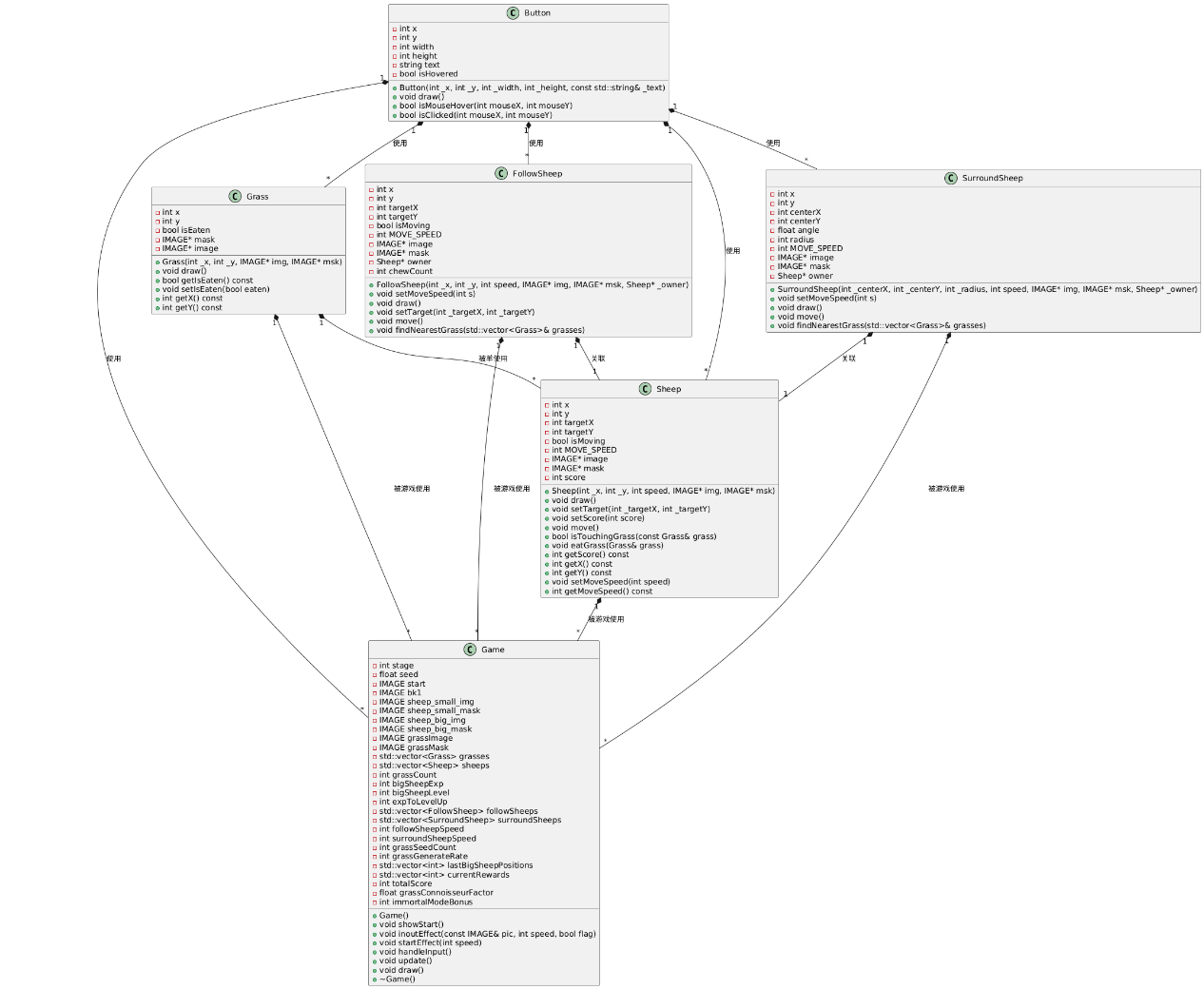
**1 . 设计思想**

采用全局对象化的思路，将所有组件封装为对象。每一个对象的基本方法为draw方法，通过数组存储对象位置，通过对象之间距离判断交互。

1. **整体流程**

**类图：**

****

****

实现思路：

游戏初始化：在Game类的构造函数中，初始化游戏的各个元素。设置游戏的阶段stage为 0，初始化种子seed，加载游戏所需的图像资源，如开始界面图像、背景图像、羊的图像及其遮罩、草的图像及其遮罩等。创建两个羊的实例，分别设置其初始位置、移动速度、图像和遮罩。

按钮类Button：定义了Button类，用于表示游戏中的按钮。按钮具有位置（x、y）、大小（width、height）、文本内容（text）和是否悬停（isHovered）等属性。draw方法用于绘制按钮，根据鼠标是否悬停改变按钮的填充颜色。isMouseHover方法用于判断鼠标是否在按钮上，isClicked方法用于判断按钮是否被点击。

草类Grass：Grass类表示游戏中的草。草具有位置（x、y）、是否被吃掉（isEaten）、图像（image）、遮罩（mask）、羊吃的次数（sheepEatingCount）等属性。draw方法用于绘制草，根据isEaten的值决定是否绘制。getIsEaten方法用于获取草是否被吃掉的状态，setIsEaten方法用于设置草的被吃状态，incrementSheepEatingCount方法用于增加羊吃的次数，resetSheepEatingCount方法用于重置羊吃的次数。

羊类Sheep：Sheep类表示游戏中的羊。羊具有位置（x、y）、目标位置（targetX、targetY）、是否移动（isMoving）、移动速度（MOVE\_SPEED）、图像（image）、遮罩（mask）、是否在吃（isEaten）、剩余吃的时间（eatTimeLeft）、当前目标草的索引（currentTargetGrassIndex）、吃掉的草的数量（grassEatenCount）等属性。draw方法用于绘制羊。findNearestGrass方法用于找到最近的未被吃掉的草并设置目标位置。move方法用于移动羊，根据目标位置和当前位置计算移动的距离和方向。eat方法用于让羊吃草，根据羊吃的次数减少剩余吃的时间，当时间为 0 时，设置草为被吃掉状态。isInsideRect方法用于判断羊是否在指定的矩形区域内，getGrassEatenCount方法用于获取羊吃掉的草的数量，setGrassEatenCount方法用于设置羊吃掉的草的数量。

栅栏类Fence：Fence类表示游戏中的栅栏。栅栏具有位置（left、top、right、bottom）属性。draw方法用于绘制栅栏，设置线条样式和颜色，绘制矩形。

游戏逻辑处理：handleInput方法用于处理鼠标输入。根据游戏的阶段进行不同的操作，如在游戏开始阶段，点击鼠标左键开始游戏；在游戏进行阶段，点击鼠标左键放置草；在游戏结束阶段，点击 “重新来过” 按钮重新开始游戏，点击 “切换模式” 按钮切换游戏模式。checkWin方法用于检查游戏是否胜利，判断两只羊是否分别在两个矩形区域内。calculateKongrongScore方法用于计算孔融让梨模式下的得分，根据大羊和小羊吃掉的草的数量计算得分。

游戏更新：update方法用于更新游戏状态。在游戏进行阶段，让羊寻找最近的草并移动、吃草，根据草是否被吃掉移除被吃掉的草。在栅栏挑战模式下，检查游戏是否胜利，更新草的数量和回合数；在孔融让梨模式下，根据得分更新最高得分。

游戏绘制：draw方法用于绘制游戏画面。根据游戏的阶段绘制不同的内容，如在游戏开始阶段绘制开始界面和提示信息；在游戏进行阶段绘制背景、草、羊、栅栏等元素，显示游戏信息；在游戏结束阶段绘制游戏失败信息和 “重新来过” 按钮、“切换模式” 按钮。

游戏运行：在main函数中创建Game对象并调用run方法，开始游戏循环。在循环中，根据游戏的状态进行输入处理、更新和绘制操作，控制游戏的帧率。

1. **收获**

**一、**

深入理解了如何通过类来封装数据和行为，比如 Button 类封装了按钮的位置、大小、文本以及鼠标交互相关的方法。各个类之间通过对象的创建和方法调用进行协作，像 Game 类使用 Sheep、Grass 等类的对象来构建游戏世界，这种面向对象的编程方式提高了代码的可维护性和可扩展性。

掌握了 easyx 图形库的基本操作，包括图像的加载（如加载羊、草、背景等图像）、图形的绘制（绘制按钮、栅栏等）以及双缓冲绘图技术（通过 BeginBatchDraw 和 EndBatchDraw 减少画面闪烁）。了解了如何设置绘图的颜色、字体等属性，使界面更加美观。

使用 vector 来管理游戏中的对象集合，如 std::vector<Grass> 用于存储草的对象，std::vector<Sheep> 存储羊的对象。通过 vector 的 emplace\_back 方法添加元素，erase 方法移除元素，提升了对数据结构的实际应用能力。

学会了如何通过 peekmessage 和 WM\_LBUTTONDOWN 等消息来处理鼠标左键点击事件，以及如何判断鼠标是否悬停在按钮上（如 Button 类的 isMouseHover 方法），实现了游戏的交互功能。

**二、**

Game 类中的 stage 变量用于管理游戏的不同阶段（开始界面、游戏进行、升级奖励等），合理地控制游戏流程。在不同阶段执行相应的操作，如在开始阶段显示开始界面，游戏进行阶段处理游戏逻辑和绘制游戏画面。

羊的移动、寻找草、吃草等行为都有详细的逻辑设计。羊会寻找最近的草并移动到草的位置，当到达草的位置后会吃草，通过判断草是否被吃掉以及羊的咀嚼时间来更新游戏状态。这种行为逻辑增加了游戏的趣味性和真实性。

草的生成采用随机位置生成，并根据一定的规则（如 grassGenerateRate 变量控制生成频率）进行生成。同时，通过判断草是否被吃掉来管理草的生命周期，当草被羊吃掉后从 grasses 容器中移除。

游戏中的羊通过吃草获得经验值，当经验值达到一定值后可以升级。升级后会有不同的奖励选项，如领养新的羊、增加羊的移动速度、增加草籽数量等，丰富了游戏的玩法和体验。

**三、**

在实现羊与草的碰撞检测（如 Sheep 类的 isTouchingGrass 方法）时，需要考虑羊和草的位置以及大小，通过计算中心坐标和边界来判断是否发生碰撞。在调试过程中，通过打印坐标和距离等信息来排查碰撞检测不准确的问题。

在设计游戏时，需要平衡各个元素，如羊的移动速度、草的生成速度、升级所需的经验值等。通过不断调整参数（如 MOVE\_SPEED、grassGenerateRate、expToLevelUp 等）来优化游戏的平衡，使游戏既具有挑战性又不会过于困难或简单。

在使用 IMAGE\* 指针时，需要注意内存的分配和释放。虽然代码中没有显式的内存释放操作（依赖于系统自动管理），但在更复杂的场景下，需要考虑对象的生命周期和内存管理，避免内存泄漏。