计算机程序设计课程实践项目报告

C#语言版Roll a Ball

**完成人： 邱慕凡 PB20061360**

**曾彬城 PB20000158**

**徐笑阳 PB20000155**

**完成时间：2020年12月30日**

摘 要

本项目

关键词：

目 录

[1. 需求分析 3](#_Toc24442708)

[1.1 项目特点 3](#_Toc24442709)

[1.2 需求分析 3](#_Toc24442710)

[1.2.1 功能分析 3](#_Toc24442711)

[1.2.2 性能分析 3](#_Toc24442712)

[2. 概要设计 3](#_Toc24442713)

[2.1 系统总体设计 3](#_Toc24442714)

[2.2 主要数据结构 4](#_Toc24442715)

[2.3 模块定义与接口 4](#_Toc24442716)

[2.4 开发语言和开发工具 4](#_Toc24442717)

[3. 系统详细设计 5](#_Toc24442718)

[3.1 主体功能设计 5](#_Toc24442719)

[3.1.1 主函数 5](#_Toc24442720)

[3.1.2 菜单模块 5](#_Toc24442721)

[3.1.3 打印模块 5](#_Toc24442722)

[3.1.4 判断模块 5](#_Toc24442723)

[3.1.5 游戏逻辑模块 5](#_Toc24442724)

[3.2 其他功能设计 6](#_Toc24442725)

[3.2.1 其他模块 6](#_Toc24442726)

[4. 系统调试 6](#_Toc24442727)

[5. 系统运行 7](#_Toc24442728)

[6. 结论 7](#_Toc24442729)

[7. 参考文献 8](#_Toc24442730)

# 项目概述

## 项目特点

本游戏是2D横版积分游戏。受到2D横版过关游戏《超级马里奥》和《猫里奥》的启发，开发组结合了《超级马里奥》的操作方式与《猫里奥》富有特色的关卡设计，继承《正常的大冒险》的游戏风格，加以开发者对游戏方式与内容的创新，设计出这个以获得高分为目标的游戏。本游戏基于Unity开发环境，可以在Windows与MacOS上运行，游戏操作简单方便，界面丰富多彩，性能与稳定性极佳，可以激发玩家与开发者对计算机学习的兴趣，提升玩家的手脑协调的能力，是一个非常有趣的小游戏。

## 项目设计

在开发的初期，开发组对编程枯燥的界面没有感到太大的兴趣，但是Unity独具一格的可视化编程方法可以大大提高开发组的兴趣。它基于C#的面向对象编程理念也可以帮助开发组管理代码，扩充编程思维，从而更深入地了解计算机，也能方便开发组实现自己的想法， 更高效地开发出更高质量的游戏。为保证游戏质量，本游戏专门有一位开发人员不参与游戏性的开发，只负责测试游戏与提供游戏所需素材。同时开发组使用了Photoshop绘制了素材与概念稿，使游戏UI明了、简洁。

### 功能分析

1. 本游戏是单机游戏，游戏目标是玩家控制小球绕过障碍物，利用机关、躲避怪物、获得高分。
2. 本游戏有部分素材从Unity的Asset Store里选取恰当的素材并且进行适当加工，剩余素材为原创素材。这些素材使界面简洁清晰，一目了然。
3. 本游戏的关卡设计合理，有专门的关卡设计师负责。
4. 本游戏允许二段跳，提升了玩家的可操作空间与游戏趣味。

### 性能分析

1. 游戏画面流畅，性能优良，无卡顿
2. 游戏关卡设计富有趣味，锻炼玩家心理素质，耐玩性极高
3. 本游戏代码开源，不盈利，旨在为玩家与开发者提供一个学习平台

# 系统设计

## 系统总体设计

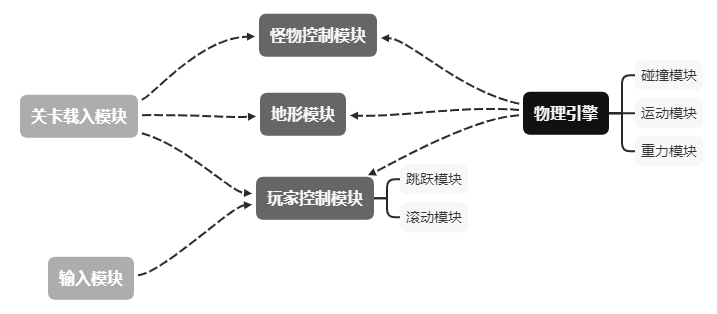
小游戏有简单的菜单选项，用来选择相应的功能，包括：

1. 开始游戏
2. 游戏操作说明
3. 游戏的规则说明
4. 选择关卡
5. 选择是否允许用来降低游戏难度的操作

在游戏中应体现的功能包括：

1. 游戏的暂停和开始
2. 游戏的得分纪录和得分的实时更新
3. 游戏操作说明
4. 游戏的规则说明
5. 退出游戏

该项目包含关卡载入、输入、怪物控制、地形、玩家控制、模块与物理引擎。其关系图如下：



其中输入模块控制键盘与鼠标的输入，关卡载入模块将数据库中的关卡数据读入程序，玩家控制模块控制玩家可进行的操作，包括滚动与跳跃两种操作，地形模块控制玩家与地形障碍物的互动，怪物控制模块控制怪物与玩家的互动，而物理引擎是整个游戏的基石，它可以控制碰撞、运动与重力，使游戏看起来更自然，更符合玩家的直观感受，优化游戏体验。

## 主要数据结构

使用结构体类型定义：

struct Tetris{

int x;

int y;

int number;//产生方块的个数

int type;

int flag;//方块类型序号

int next;//在随机函数中表示下一个要下落的方块序号

int scord;//表示游戏过程分数的表示

int speed;//表示游戏过程中速度的变化

int level;//表示游戏的难易程度

};

## 模块设计

### 菜单模块

该模块通过按钮来进入不同的场景，具体规则如下：

1. Play按钮，用来进入游戏的主界面level01.unity
2. Help按钮，用来进入帮助界面HelpPanel
3. Quit按钮，用来退出游戏

### 判断模块

该模块主要功能是逻辑判断功能模块，判断的内容主要是：

1. 游戏是否结束：判断游戏是否结束，如果玩家被怪物杀死或者触碰致死的障碍物。
2. 是否触发机关：判断机关是否被触发，如果玩家或者怪物碰到机关触发器，那么机关就会被触发。
3. 角色是否碰到障碍物：判断玩家或者怪物是否碰到障碍物，如果碰到则触发反弹。如果玩家下方没有障碍物则玩家受到向下的重力加速度。
4. 怪物是否被杀死：判断玩家是否从上方击中可被杀死的怪物从而判断该怪物是否被杀死。
5. 判断是否得分：通过判断小球是否击中可得分物体来判断玩家是否得分。

### 游戏逻辑模块

该模块是游戏的主要模块，游戏需要有逻辑规则来保证趣味性的，该模块主要包括的功能函数有：

1. 保证左右键的作用，移动、加速与变换。
2. 保证分数的更新和相应游戏模式的变化。
3. 保证游戏规则。
4. 保证怪物在玩家可到达的地方并不断移动。
5. 保证物体与物体之间不穿模。
6. 判断玩家是否可以进行跳跃或二段跳。
7. 定时清理游戏中过多的可收集物品与怪物。
8. 定时自动生成一定数量的怪物和可收集物品。

### 动画模块

该模块是游戏的动画模块，游戏需要动画的润色才能达到更好的视觉效果，给玩家更好的体验：

1. 怪物动画动画：为怪物的移动与死亡添加动画，为玩家提供即使的视觉反馈。
2. 玩家死亡动画：为玩家的死亡添加动画，轻快的画风也可以降低玩家心中的沮丧情绪。

### 声音模块

该模块是游戏的声音模块，声音不仅可以渲染游戏的气氛，也可以给玩家提供信息反馈：

1. 背景音乐：为游戏添加轻快的背景音乐，给玩家游戏的快节奏体验，也能增加玩家心中的喜悦。
2. 怪物击杀音效：为怪物的死亡添加声音，给玩家提供反馈的同时也能宣泄玩家对怪物的恨意。
3. 玩家死亡音效：短暂的死亡音效，告诉玩家游戏结束了。

## 开发语言和开发工具

本系统采用C#语言作为开发语言，C#语言的主要特点如下：

(1) 语言简洁、紧凑，使用方便。

(2) 运算符和数据类型丰富，具有现代语言的各种数据结构。

(3) 具有结构化的语句。

(4) 便于进行面向对象的程序设计，方便了游戏的实现与多人合作。

(5) 生成的目标代码质量高，程序执行效率高。

(6) 用C#语言编写的程序可移植性比较好，基本上不做任何修改就能运用于各种型号的操作系统中。

(7) 与Unity兼容，方便可视化编程。

(8) 拥有强大的库，简化了许多低效冗余的操作。

因为C#语言拥有各种优点，所以本系统使用C#语言作为开发语言，简化了编写过程，简化了系统界面。

使用了Unity库。该库提供绝大多数基础运算功能与图形界面的实现。

# 系统测试

## 系统运行

游戏使用指南：

本游戏的开始界面有3个按钮：Play, Help, Quit。Play可以开始一新的游戏，Help可以进入帮助界面，Quit可以退出应用。本游戏的游戏目标是在死亡前尽可能获得多的分数。在游戏中，你的角色是一个球。你可以按左(右)方向键来向左(右)移动或者按空格键跳跃，但是不能越过障碍物。本游戏允许二段跳。游戏中的非玩家控制角色包括樱桃和怪物。吃掉樱桃可以获得分数，从上方接触青蛙怪物可以杀死它，但是从侧面或者下方接触青蛙怪物，或者以任何形式接触其他怪物会被怪物杀死。

测试员评价：

在项目完成之时，本游戏的操作手感已比最初版本好很多，开始时的bug与不合理之处已全部被修复。本游戏在配置较差的现代机器上即可做到丝毫无卡顿，游戏帧数高，视觉效果极佳。

## 系统测试

本程序的资源占用

CPU占用率：3%-7% (Intel® Core™ i7 4700MQ单核)

内存：80M-95M

GPU：30%-40% (NVIDIA GeForce GT 730M)

由于C#的语言特性，本程序不会发生内存泄漏错误

经过了大量的系统测试，我们做出了如下修改：

1. 修复了天花板可以刷新小球的二段跳
2. 修复了小球落地时立即起跳可以跳得过高
3. 在空中按左右键原改变加速度，但是这严重加大了操作难度。故将按左右键改为改变速度。
4. 综合游戏难度与创新性的考虑，将原本的过关游戏改为在有限的空间内收集分数的游戏。
5. 为了增加游戏难度，移除了石球与弹性球
6. 增大了怪物的体积，大幅度增加了玩家杀死怪物的难度
7. 增加了帮助界面，为帮助新玩家更快适应这个游戏
8. 修改了怪物的初始速度方向，这很大程度上优化了任意地图上的游戏体验

/\*

下面是对系统测试的一些介绍，你应参考它来进行自己的测试，而不是复制粘贴它。

系统测试的目的是为了发现错误而执行程序的过程，成功的测试是发现了至今尚未发现的错误的测试。

测试的目的就是希望能以最少的人力和事件发现潜在的各种错误和缺陷，我们应该根据开发个阶段的需求。设计等文档或程序的内部结构精心设计测试实例，井利用这些实例来运行程序，以便发现错误的过程。

系统测试的原则应该遵循以下基本规则。

(1) 应该尽早并且不断地进行测试。测试不是在系统开发完之后才进行的，由于原始问题的复杂性、开发阶段的多样性以及参加人员的细条等因素，在开发过程中的各个阶段都有可能出现错误，因此，测试应该贯穿在开发的各个阶段，我们应该尽早纠正作物，消除隐患。

(2) 测试工作应该避免由本人来进行，一方面，开发人员往往不愿否认自己的工作，总认为自己开发的软件过程序没有错误，另一方面，开发人员往往很难发现自己的错误，具有局限性，所以我们应该找其他人来帮忙检查程序。这样会变得更客观，更有效。

(3) 设计测试方案时，不仅要确定输入数据，而且要根据系统功能来确定预期的输出效果和结果。将实际输出结果和预期结果相比较就能发现测试的对象是否正确，是否有一些隐患。

(4) 在设计测试用例时，不仅要设计有效合理的输入条件，还要包含不合理、失效的输入条件。测试的时候，人们往往习惯按照合理的、正常的情况下进行测试，从而忽略了对异常、不合理，意想不到的情况进行测试，而这些情况可能都是隐患，可能会造成系统的崩溃。

(5) 在测试程序时，不仅要检验程序是否做了该做的事情，还要检验程序是否做了不该做的事情。多余的工作会带来副作用，影响程序的效果，有时会带来潜在的危害或者错误。

(6) 严格按照测试计划来进行，避免测试的随意性。测试计划应该包括测试内容、进度安排、测试环境、测试工具和测试资料等。严格按照测试计划可以保证进度，使各方面都得以协调进行。

(7) 妥善保存测试计划、测试用例，作为软件文档的组成部分，为了以后维护的方便做了铺垫。

(8) 测试用例都是精心设计出来的，可以为重新测试或者追加测试提供方便。当纠正错误、系统功能扩充后，都需要重新进行测试，而这些工作重复性很高，可以利用以前的测试用例，或者在其基础上进行修改，然后在进行测试，这样科技简化系统的维护时间并且提高了维护的效率。

在系统调试过程中，我们可以利用一组简单的数据对程序进行调试，来观察各函数是否有出错现象。然后用复杂数据对程序进行调试。然后我们要对系统的各项能力进行测试，看看有没有漏洞或者死循环现象。

数据测试时，如果程序出现错误，要首先找到出错的函数，对函数中各个变量进行观察，有时变量很多，循环多重不要急，对各个数据的观察，对程序的调试有很大的帮助。你也可以先对几个变量熟悉后，在多重循环里，确定正确的数据，将其跳过调试，对有疑问的数据循环进行仔细观察。虽然这是比较省力气的活，但对复杂的数据错误并不是很有效，所以仔细观察每个数据的变化对调试的成功与否有决定性作用。

测试时，根本的错误井不一定在运行错误的函数内，或许在前面的函数内已经奠定了这些错误数据的基础，需要反过头来，对所有的的函数进行一项项的调试！从程序运行的第一个函数开始调试。按照程序运行的步骤下去。一个一个地确定函数的正确性，每确定一个函数，在以后的调试中可以减少对该函数的调试次数，或者直接跳到下一个函数。所谓的各个击破便是如此。对头文件中的各个函数的返回值和它所要传入的参变量及其功能更是需要注意。

调试时，需要对错误缩小范围，从整个程序到指定函数，再到指定区域，再到指定行，在对程序缩小范围或对编译没通过的程序缩小错误范围时可以用”/\*\*/”来进行调试，确定正常范围，缩小错误范围。

\*/

# 结论和体会

本游戏开发组经过以上各章节的功能，已经基本完成了Roll a Ball游戏的开发，Roll a Ball是一个趣味性和操作性很强的游戏。本游戏代码结构清晰，便于合作开发。界面友好，基本操作简单，操作空间极大。综合了玩家的视听感官，特制的UI设计与完善的帮助界面使得本游戏对新玩家极其友好。由于时间关系，许多同学们的建议未能实现，但是这并不妨碍本游戏是一个创新性极高、体验极佳、系统完备的游戏。

开发组在开发过程中学习到了很多。第一点，也是最重要的一点，团队协作非常重要。本次作业的团队分工合理，一位同学负责主要代码，一位同学负责关卡设计，一位同学负责素材收集以及测试。这三项工作绝非易事，工作难度相当，工作方向也完全不同。这样的分工大大提高了我们的效率。这次制作也有一些工作分配不合理的地方。比如一位组员制作的许多素材与另一位组员的冲突了，导致其中有大量的工作冗余。不过这更加深了我们对团队协作重要性的理解。第二点，虽然我们有强大的Unity引擎支撑，但是我们遇到的bug数量如此之多，开始时操作不方便之处与想象中差距那么大。2D横版游戏看似简单，但是其中的碰撞检测对延时要求非常短，碰撞检测的精确程度要非常高，不然违背玩家直觉的现象会在很大程度上败坏游戏体验。再者，设计师不得不为了游戏性在物理常识方面做出妥协，即空中转向方面。大量的游戏实践表明在空中加入合适的转向是非常有必要的。第三点，2D横版游戏的关卡设计难度是非常大的。曾经我们玩《超级马里奥》与《猫里奥》的时候对其中部分关卡设计的不合理之处有过许多不满，但是最初的2D横版过关模式的关卡设计就给设计师带来了许多困扰，最后不得不通过改变游戏形式来平衡游戏难度。最后一点，经过了这次游戏制作，我深刻理解到设计一个直观简约的UI是一项非常困难的工作。网上的素材使海量的，但是真正适合我们游戏的素材却凤毛麟角。在大量的搜索工作后，我不得不在Photoshop中独立绘制属于自己的游戏素材。经过了大量的工作，我们也仅仅制作了主界面、帮助界面与小部分游戏素材。

这次的课程实践让我们学到了更多团队协作的技能，也让我明白了前人的工作极其不容易，我们不能总用苛责的眼光看待那些极其优秀的作品。总而言之，这次实践让我们获得了如此多的课本上学不到的宝贵经验。