程设大作业报告

小组成员：谢昱辉，陈昕宇

1 简介

本项目是基于Qt creator的小游戏《Isaac》，该小游戏根据面向对象的思想搭建，允许玩家在游戏中灵活地使用各种功能躲避敌怪、杀死敌怪并在无尽游戏中获得尽可能高的分数。本游戏提供了两种玩家的移动模式，和强大的杀怪技能e,q,r。值得注意的是，本游戏在性能优化、敌怪行为设计和技能效果上都进行了打磨以达到最好的游玩效果。

2 程序功能介绍

2.1 开始界面

开始界面是一个欢迎窗口。值得注意的是，在config.h中更改的版本号会反映在欢迎界面上。在开始界面，你可以选择两种移动方式，即仅使用上下左右键进行移动、和使用ad+上下左右的移动方式两种。后者允许斜向平移，故更简单。

2.2 主界面

选择后进入主界面，玩家通过之前选择的移动方式移动到“start”键上，此时下方英文提示玩家按下[space]开始游戏。下方英文给玩家介绍了基础的移动方式。在该界面，玩家可以选择退出。

2.3 战斗界面

2.3.1 主人公：普通攻击

主人公的普通攻击通过调整朝向来调整角度（与此同时发出的子弹角度也相应改变），每一颗子弹和敌人碰撞框相交后敌人-1血量。

2.3.2 主人公：e技能

玩家通过按下e释放e技能，主人公根据当前朝向在一定角度范围内发出一圈更快的特殊子弹。释放e技能之后，主人公进入短暂的e技能冷却。

2.3.3 主人公：元素晶球

主人公通过子弹消灭敌怪后，场上有一定概率会存留元素晶球。这是一种闪烁的、有时限的掉落物，被主人公拾取后会增加q技能充能，该充能比例在q技能图标上展示。

2.3.4 主人公：q技能

充能完成后，主人公可以按q释放q技能。q技能会对战场上的怪物进行时间暂停。怪物在q技能释放期间停止移动。主人公的射弹速度翻倍。玩家可以在此时间段内灵活采取策略，是尽快清理敌怪或是逃出包围圈？全都由你决定。Q技能结束，时间暂停结束，r技能获得充能。同时q技能进入冷却。

2.3.5 主人公：r技能

充能完成后，主人公可以按r释放r技能，使出毁天灭地的一击。主人公会模仿《崩坏：星穹铁道》中的希儿切割战场屏幕，并在一声枪响后抹杀屏幕上的所有敌怪。在危急时刻使用r技能以给你带来意料之外的转机！

2.3.6 主人公：冲刺

主人公按下shift以根据当前朝向进行一段冲刺。冲刺会消耗主人公一定的体力，体力条显示在主人公血条下方并会快速恢复。

2.3.7 敌怪：没头的怪物

Isaac战场中最普遍的敌怪。它血量为1（也即一枪就死），并有简陋的追踪机制，它会根据主人公当前位置来行进。

2.3.8 敌怪：有头的怪物

和上一种敌怪类似，但它血量为3

2.3.9 敌怪：苍蝇

和上一种敌怪类似，但它移速更快

2.3.10 敌怪：颤动

为了让敌怪更真实，它在每一个计时器tick会进行一个x、y轴上的随机微小位移。我们发现它可以很好地模拟怪物的颤动效果。

2.3.11 敌怪：冲刺的暗影

这是一种精英怪，不能被杀死，主人公所能做的就是躲避它。它拥有独特的计时器以控制其行为，并有一个state成员专门记录其状态，具体机制为：

在state==”wait”时，暗影进入冷却阶段，一段时间后，冷却阶段有一定概率结束，进入show阶段。

在state==”show”时，暗影进入路径展示阶段，一条红色框会揭示其即将冲刺的路径。主人公最好在此阶段内躲避。一小段时间后，暗影进入”dash“阶段。

在state==”dash“时，暗影以很快速度按照路径冲刺而过，对碰到的主人公造成5点伤害。冲刺结束后有2/3概率进入下一次”show”阶段，有1/3概率回到“wait”阶段，等待下一次冷却结束后被唤醒。

2.3.12 敌怪：冒血

敌怪被击中后会有冒血动画。

2.3.13 敌怪：血迹

敌怪被杀死后会留下方向随机、透明度随机的血迹，血迹在一定时间后清除。（问：玩家如何获得更好的打击体验？答：更多的血！）

2.4 结束界面

当玩家血量小于等于0后，进入结束界面，其会显示“game over”和玩家当前分数。玩家可以选择回到menu重新挑战或exit退出游戏。

2.5 icon

《Isaac》中怪物icon使用了以撒的结合wiki中的怪物icon。它们非常好用。

2.6 音效

《Isaac》中开始界面和战斗界面使用了《崩坏：星穹铁道》中的背景音乐，主界面和结束界面则使用了自己制作的背景音乐。

3 项目各模块与类设计细节

3.1 auxstructure类

auxstructure类里封装了一些对其他类很有用的对象。如一个可以计算增幅的VariableData类，一个用来刻画很多条装ui的TubeLikeData类，以及一个可用于管理各种事件的EventManager类。

3.2 virtualbutton类

virtualbutton类封装了一个方便使用的按钮类，在virtualbutton中实现了按钮的初始化和绘制。之后的普通按钮有了统一的实现方式。

3.3 config头文件

在config.h文件中，我们放入了所有游戏内的可调整参数以方便改变与查找。它们包括：游戏配置数据、地图配置数据、主人公配置数据、子弹配置数据、

敌怪配置数据、奖励相关数据、动图播放数据、元素晶球相关数据、以及各个ui和icon所用到的图片地址。

3.4 enemy类

enemy1,enemy2,enemy3,enemy4类均公有继承了enemy类，里面封装了敌人初始化和坐标更新的函数。其中，enemy4类控制暗影的行动，为了实现更特殊的功能，在重写了坐标更新函数之外还封装了决定其等待冲刺、展示路径和冲刺以上三种状态转移的函数。其内部维护了一个独特的recorder来实现状态转换。我们接下来的更新目标是让每个enemy都有一个recorder来帮助其进行状态转换。这在一些更复杂的敌人上可能会有用。

3.5 hero类

在hero类里封装了hero的各个状态、初始参数、和进行各个行为的事件管理器（它们是一个EventManager对象）。Hero类中包含了hero初始化、移动和正常释放子弹的函数，也包含了hero释放q,e,r技能所必要的条件判断和释放函数。另外、为了保证hero在旋转过程中的平滑流畅，在hero中封装了转向的函数。

3.6 bullet类

由于在hero中已经把子弹发射时的状态转换写好了。bullet类中只包括子弹初始化和日常位置更新的函数。

3.7 bloodtrail类

用来控制敌人死后的血迹。包括初始化和状态更新（主要是要实现越来越浅的效果）。

3.8 energy类

包括了元素晶球初始化、闪烁和随时间消失的函数

3.9 movevector类

将按键交互和对应效果连接在一起。使用了一个数组来管理每一个按键是否被按下。避免了按键同时按下时彼此冲突的情况，同时使得英雄可以斜向移动。

3.10 audiothread类

起初是尝试手写代码用来管理音乐的播放进程以debug，但是最后使用别的方式解决了bug，并没有用上这个类。

3.11 startscene类

简单的开始界面，实现了上述的功能

3.12 mainscene类

复杂的主程序类，所有需要被统一管理的函数和对象都在这里实现。mainscene里面主要的函数都与游戏的基本运行逻辑有关。包括每10毫秒一次的游戏场景绘制和各单位状态的刷新与位置的计算。同时，mainscene也负责实时读取键盘信息，来控制英雄移动和射击。

1) **MainScene函数** 初始化界面

2) **mainLogic函数** 把各个分支情况和对应的界面连起来

3) **initScene函数** 按钮实现和各资源加载，窗口和按钮事件被初始化，第一波敌怪初始化

4）**enemyToScene函数** 若有空闲状态的敌人对象，让它们重新回到场上

5）**updatePosition函数** 统一管理敌人更新、子弹更新、能量更新、血迹更新。同时根据玩家的键盘输入采取不同的措施（q,e,r和shift）。一个复杂的变换被用于计算主人公的角度变化和位移（包括了冲刺和非冲刺的情况）。

6）**updatePosition4welcome函数** 开始界面的逻辑管理（包括主人公位置更新）

7）**updatePosition4result函数** 结束界面的逻辑管理（包括主人公位置更新）

8）**paintEvent函数** 总的paint逻辑

9）**paintCutScene函数** 画r技能的切割屏幕

10）**paintWelcomeScene函数** 画欢迎界面该有的东西

11）**paintResultScene函数** 画结束界面该有的东西

12）**paintInGameScene函数** 画主界面该有的东西

13）**resetScene函数** 重置

14）**playGame函数** 以一个固定帧率刷新，执行下属的各个函数

15）**welCome函数** 欢迎界面的主函数

16）**gameOver函数** 结束界面的主函数

17）**keyPressEvent函数** 按键事件，和之前开的按键数组有关

18）**keyReleaseEvent函数** 松键事件，和之前开的按键数组有关

19）**killAll函数** r技能全部消灭函数

20）**collisionDetection函数** 碰撞框检测和对应的碰撞后事件函数

21）**collisionDetection4welcome函数** 欢迎界面独有的碰撞检测及事件触发

22）**collisionDetection4result函数** 结束界面独有的碰撞检测及事件触发

23）**InfoIcon函数** 字体初始化

24）**paint函数** 画笔初始化

25）**paintInfoComponent函数** 绘制信息

26）**paintHostileObject函数** 绘制血迹和敌人

27）**paintFriendlyObject函数** 绘制子弹、英雄、元素晶球、体力和血条

28）**paintDebug函数** 绘制debug信息

29）**paintMask函数** 绘制r技能蒙版

4 反思与展望

反思方面，我们发现从零开始实现我们的宏伟想法是很困难的，但一步一步增加主体玩法、不停push版本可以让我们每一次实现一个小功能、然后慢慢充实。在7月8日截止时，我们的程序已迭代了4个大版本，拥有日积月累实现的丰富功能，这是让人十分可喜的成果。在程序设计上，我们遵循“先写出来，再改良”的办法，这让我们不至于在第一个新版本想的太多以至于无从下手。

展望方面，我们计划在未来的某个版本进行一次大更新，将游戏主体玩法更改为roguelike式。这需要我们在之后的各个版本小更新下实现更好的类封装、加入更丰富的敌怪和更多样化的主人公行为。另外，奖励机制、特殊事件设置、boss设置和最关键的——roguelike游戏的探索树机制还需要搭建。我们期待一个逐层深入、循序渐进、奖励机制和提升树合理的isaac能够带给玩家更好的游戏体验。

5 成员分工

陈昕宇：程序编写和测试、程序玩法设计、ui和icon设计、qt报告撰写
谢昱辉：程序编写和测试、程序玩法设计、ui和icon设计、qt视频录制
特别感谢：李浩石