高级程序设计 **2020 春**

课程设计 2

指导老师：陈家骏 黄书剑 姓名：王晨渊, 学号：181220057

1 编译运行环境

使用了 VS2019 进行编译运行。请将 windows 控制台放大到最大以改善使用体验。

2 操作与规则

2.1 操作

仅实现了基本要求。玩家出生在屏幕下方。WASD 操控移动，J 发射炮弹。

2.2 规则

每击杀一辆坦克积 1 分，敌人的内讧击杀同样积 1 分。玩家共两条生命。

共 3 辆敌军坦克。轻型坦克生命为 1，速度为 3，为蓝色。中型坦克生命为 2，速度为 2，为红色。重型坦克生命为 3，速度为 1，为绿色。玩家为轻型坦克。

每一点生命可以承受一发炮弹，生命规律后死亡。

屏幕下发红色星星为基地，任意炮弹命中星星会导致游戏失败。

胜利条件为所有敌人死亡。失败条件为基地被击中或玩家控制的坦克生命为 0。

3 设计思路

3.1 核心思路

本质上，程序中有一个画板，画板中的每个像素点都存储了对应的属性信息。所谓的 tank 炮弹等等只是一个个管理员，用于管理对应的一些像素点。游戏的交互本质上是这些管理员的交互。

3.2 代码结构

common 中定义了项目中所有需要的宏和一些全局变量，此外包括了一些特殊函数的实现，如将输入映射到 UP 等宏的映射函数。

Screen 类型对象实现输入、交互、更新功能。敌人的行为利用随机数，复用 Screen 的输入模块实现。Screen 的成员对象包括子弹管理类 (Bullets),item* 指针等等。

Bullet, Tank 都继承 Item 类型。Item 是一个纯虚类。由于设计的失误，他们对应的方向等等信息接口实际上并不完全相容。为了保证编译通过，因此编写了一些冗余代码。

Bullets 是 Bullet 类型对象的管理员，将系统中可能的所有 bullet 类型对象都设置为其成员对象。

Vector2D 类是坐标类型的一个包装，直接复用了上次贪吃蛇的设计。

Block 是像素点，保存了单个像素的信息，如颜色、类型等。自认为一个比较脏的设计是，像素的信息中，包含了一个指针，用于指向管理这个像素的管理员。这样明显破坏了数据的封装，但是实际上带来了极大的便捷，可以迅速索引到对应的管理员从而完成一些更新。

LightTank,MediumTank,HeavyTank 仅仅是对 Tank 的一些成员的信息进行了修改，如修改速度、血量。虽然继承自 Tank，但是实际上没有本质区别。

游戏的进行采取了传统的批处理系统的模式，以 Sreen 为核心进行。print_text 和 print_graphic 是真实的输出函数，用于输出图像和游戏的统计信息。

4 尚未解决的问题

4.1 虚基类设置不合理

Item 作为虚基类，其两个派生类 bullet 和 tank 需要的接口明显不同，同时程序中所有的指针都是 Item*，直接导致了程序中塞了很多的无意义代码用来保证编译通过。解决办法是彻底重构调整结构，同样由于时间问题没有进行。

4.2 ”画板”设置不合理

画板数组应当直接设置为全局变量，而非私有成员。设置不合理导致程序中存储了大量完全一样的画板指针，造成了极大的信息冗余。由于时间来不及，没有进行优化。

4.3 刷新速率过慢，有明显屏闪

我通过关闭 cout 的与 stdio 的协同来加快速度，但是很遗憾问题没有完全解决。我认为可能需要一些移动光标的库函数，仅刷新改变的部分才有效。但是由于时间有限没有更新。