****

课程设计报告书

**题目：坦克大战设计报告书**

**学 院 计算机科学与工程学院**

**专 业 网络工程**

**学生姓名 陈宇驰**

**学生学号 201866590050**

**指导教师 沃焱**

**课程编号 145241**

**课程学分 2**

**起始日期 2019年3月至2019年6月**

|  |  |
| --- | --- |
| 教  师  评  语 | 小组成员陈宇驰, 陈怡霏两位同学通过协调合作完成了《坦克大战》的开发与设计，组内共2名同学。  该组同学类的结构设计清晰合理，代码正确，功能丰富，还原度高，答辩过程能够准确回答问题。  陈宇驰同学完成了Console类, Map类, Bullet类的开发.  教师签名：  日期：2019年6月17日 |
| 成  绩  评  定 | 92分 |
| 备  注 |  |

**坦克大战**

1. 选题背景

经典90坦克大战是大家小时都在红白机上玩过的游戏, 年少的我们都曾深深的着迷于这款有趣的游戏. 游戏主要是玩家控制一辆坦克与敌方坦克作战, 直到消灭所有的敌方坦克. 游戏上手简单, 对战具有良好的互动性, 再加上游戏中的道具, 障碍物等元素, 给游戏增添了更多地趣味性, 让人欲罢不能.

经典的90坦克大战是在是难以超越的经典, 是我们这一代的童年记忆. 因此我们决定模仿它做一个基本的坦克大战. 游戏规则基本不变, 采取命令行界面.

待解决的问题：控制台的指定位置、形状、颜色输出；模拟子弹移动；控制台内的碰撞，按键检测。

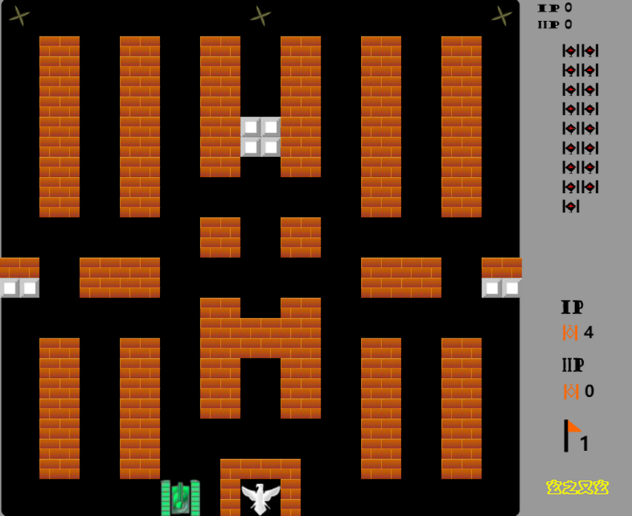
技术要求：尽可能模拟90坦克大战，不让用户感觉到控制台输入输出产生的差异。

指导思想：模拟，最大程度还原。

二、方案论证(设计理念)

设计原理：用控制台输出字符来模拟坦克大战的各种物体，道具。通过用户控制来推动游戏进程。

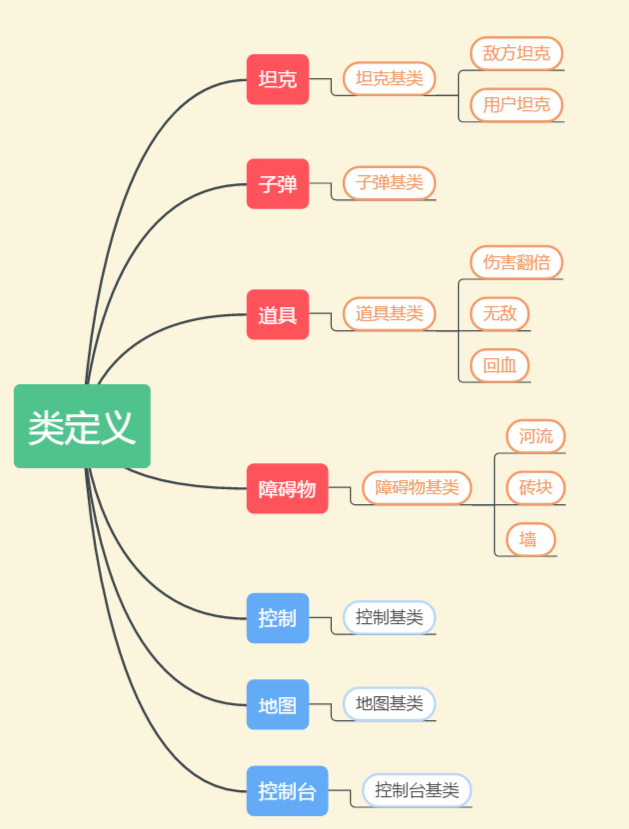
方案选择：使用一个地图二维数组来保存各种道具的情况，并通过该二维数组对游戏过程进行更新。且游戏消耗资源较少。

特点：实现起来较为明朗，不易出错，简单易懂。****

图表 1 经典的坦克大战

对此游戏的不同元素分别用不同的类来表示，然后实现类之间的相互作用，由此达到模拟该游戏的水平。考虑到有一局游戏中会生成许多的敌方坦克，所以不让用户坦克与敌方坦克直接交互，而是通过地图数组进行间接的交互。

**总的类设计：**

如上图所示, 程序包含以下七个类: Barrier, Bullet, Console, Control, Map, Prop, Tank. 下面是类的详细介绍:

图表 2 类关系图

1. Tank: 派生出两个类, 分别表示玩家坦克和敌方坦克. 玩家坦克由自己控制行走射击, 敌方坦克由随机数控制行走和射击.
2. Bullet: 负责射击过程的执行.
3. Prop: 派生的无敌可以暂时使用户坦克不会受到伤害，回血可以增加用户坦克生命值一点，伤害翻倍则是用户的子弹伤害增加一倍。
4. Barrier: 派生出三个类。其中墙坦克和子弹都不能穿过。河流只有子弹能够穿过。砖块可以被子弹打烂，未打烂之前坦克不能穿过。
5. Control: 整合了开场动画和选择菜单, 包括重新开始游戏。
6. Map: 提供的二维数组方便了子弹, 障碍物与坦克的数据的交互。
7. Console: 对控制台的基础操作实现了封装，便于调用。

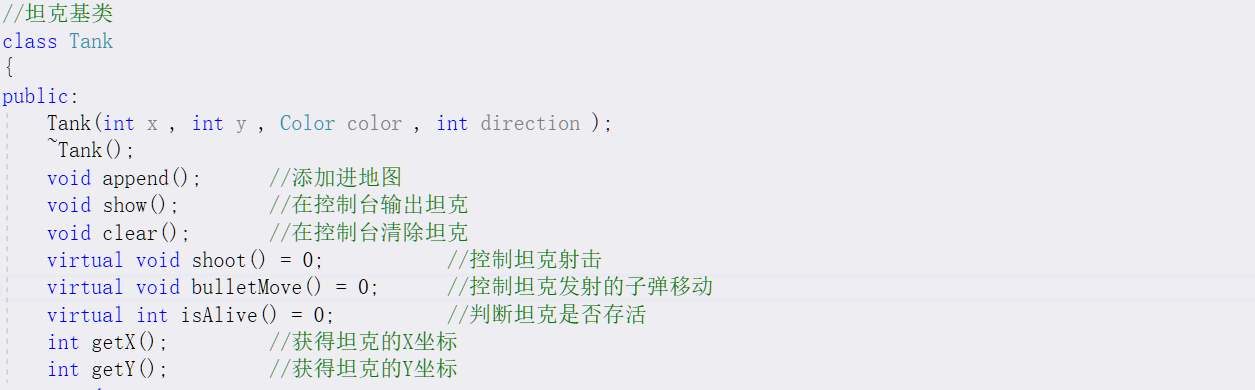
以面向对象的方式实现这些类方便不同类之间的交互, 也利于分工, 同时我们深刻的体会到了代码重用的好处.

三、过程论述

**我的类设计：**

坦克：

基类结构:

append函数将坦克的坐标存进地图的二维数组里面.

图表 3 坦克基类结构

show函数根据数组中对应的坐标在控制台输出坦克的形状。

clear函数会清除坦克在控制台的输出。

shoot函数则会根据坦克当时的朝向射出对应方向的子弹。

Move函数会接受Console类中的按键检测函数判断用户按下的键来进行对应的移动。

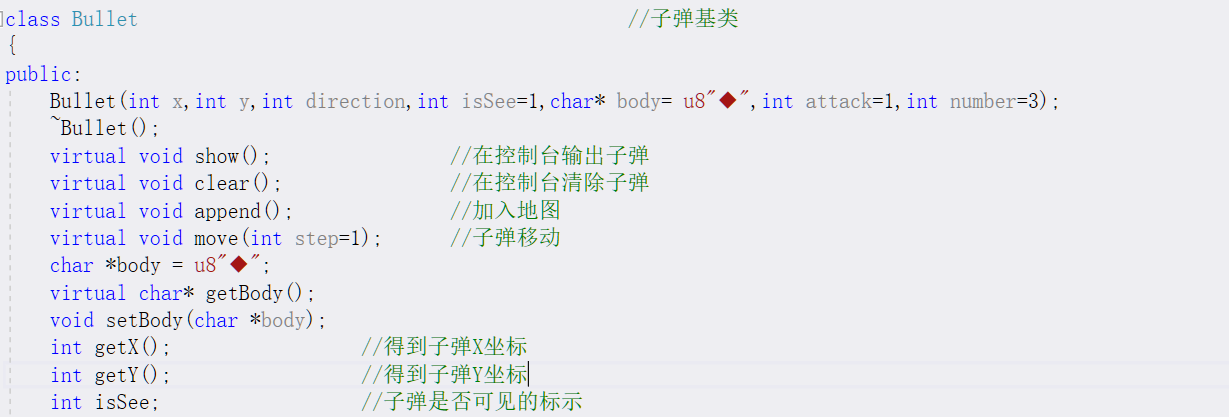
isAlive函数判断坦克是否死亡，死亡则游戏结束。

1. 用户坦克：

shoot: 如果用户按下空格键, 就会在坦克朝向的对应方向发射出一颗子弹.

move: 如果用户按下了上下左右其中一个键, 坦克就会朝对应方向移动. 清除自身原本的控制台输出并重新输出.

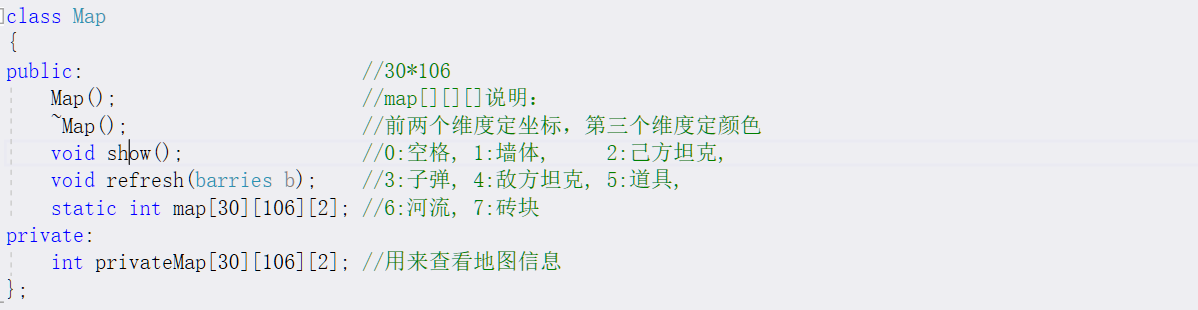
1. 敌方坦克：敌方坦克基本与用户坦克类似。只是敌方坦克的行走和射击是电脑自动操作。利用随机数来选取移动方向，每隔一定时间就发出一发子弹。

 子弹：

图表 4 子弹基类结构

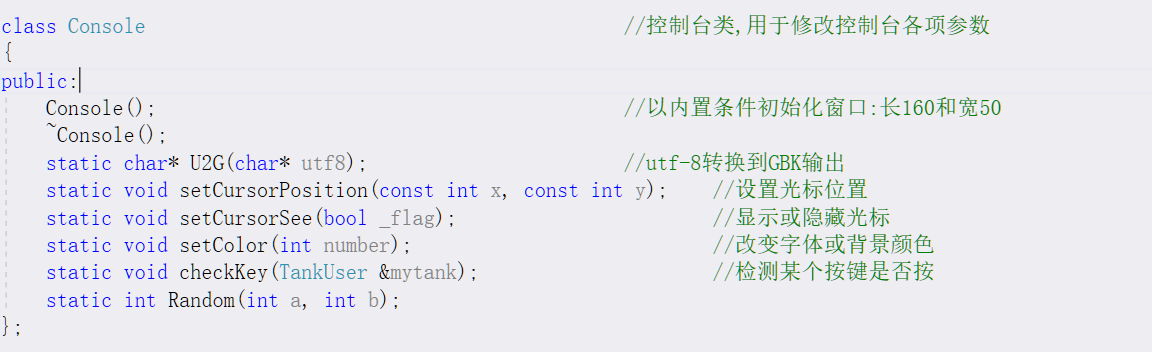
子弹的clear，show，append函数和坦克的作用一样。

move: 会根据沿途的数据做出对应反映。例如，前面是一辆坦克，如果是敌对坦克，子弹就会把数据插入敌对坦克的数组内部，敌对坦克检测到自己内部有子弹就会减血。如果是友军坦克，子弹就会直接消失，不造成伤害。

 地图：

图表 5 地图类结构

地图保存着一个全局二维数组，便于进行数据交互。任意类可以进行自由的读取。

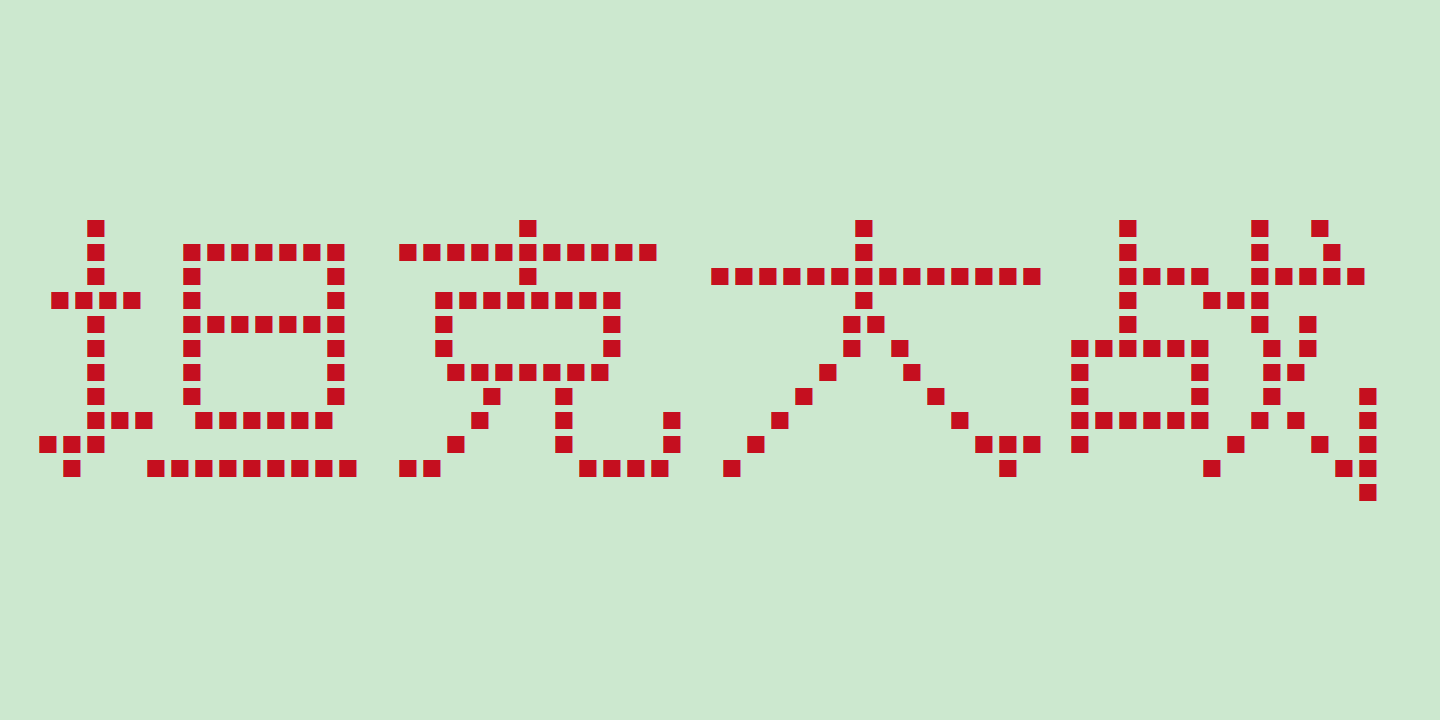
 控制台：

图表 6 控制台类结构

实现了一些对控制台操作的封装，例如把光标移动到指定位置、转换utf-8字符输出到控制台、还有改变字体颜色或者背景色。有利于减小重复代码量，优化组织结构。

四、结果分析

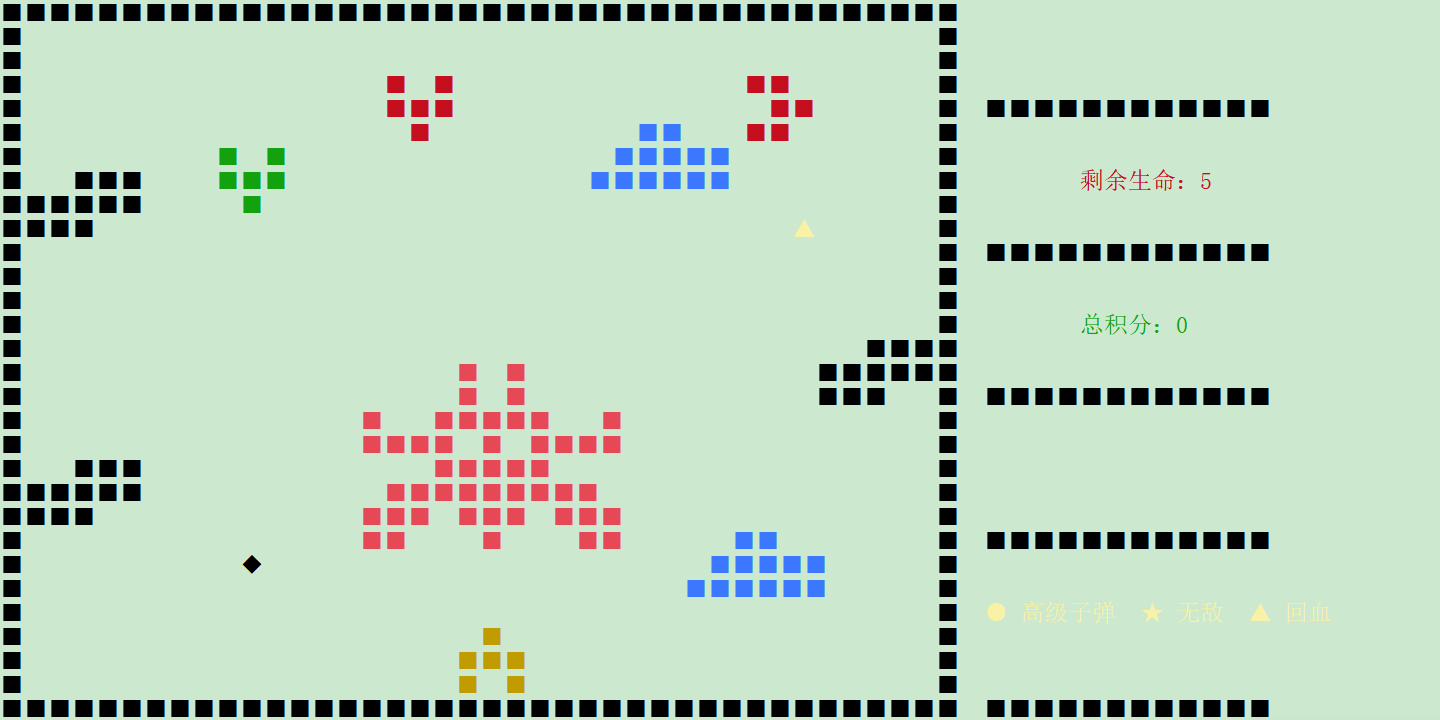
基本完成预定任务, 以下是游戏过程展示:



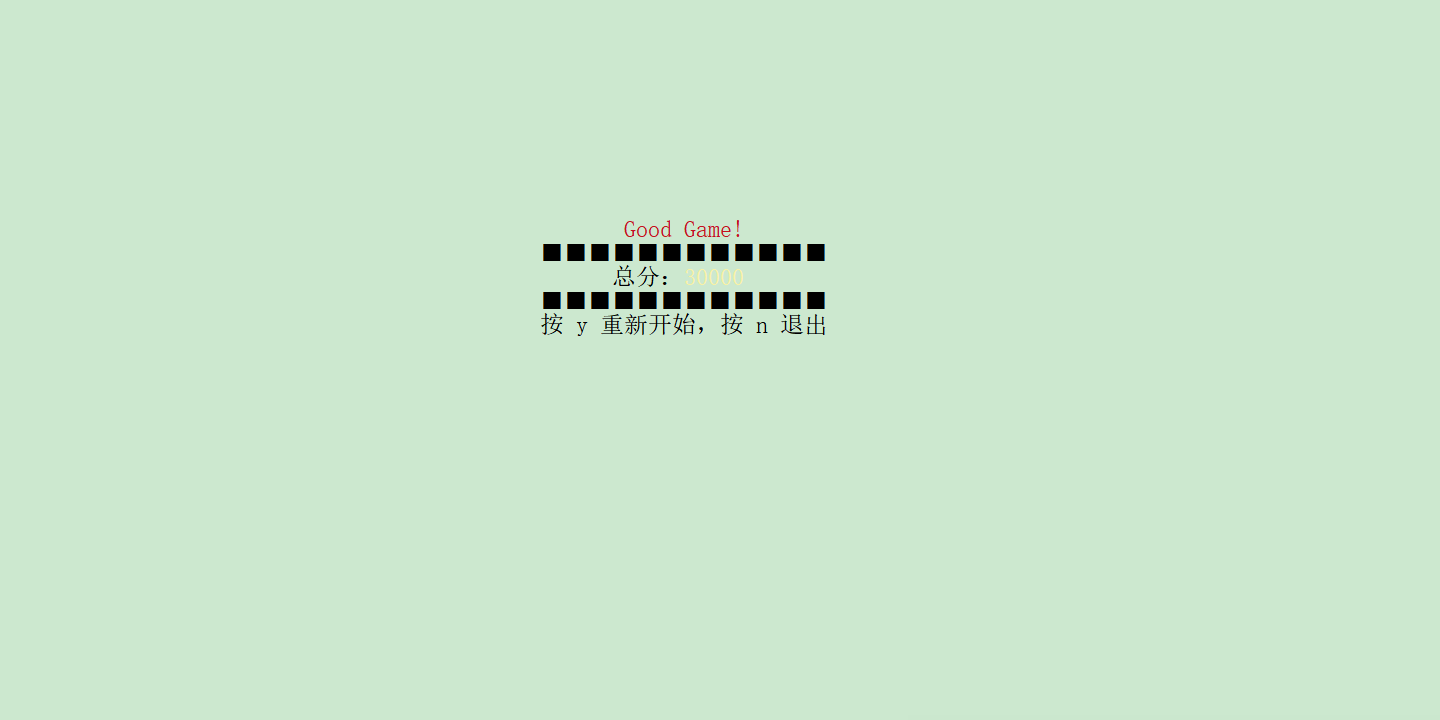
图表 7 开场动画



图表 8 开始界面



图表 9 对战界面



图表 10 结束界面

五、课程设计总结

1.项目完成过程中发现一个天马行空的思维会有助于解决一些看起来难以解决的问题--比如调试。由于地图二维数组比较大，如果在断点中追踪30\*80的二维数组，一行一行的找过去看非常不方便。后来发现可以直接在控制台不间断的输出地图保存的二维数组，由此可以及时观察地图的变化，给调试工作带来了很大的提升。

2.子弹击中坦克的判断一直是项目完成过程中的一个难点，随着项目的进行，不断的修改子弹的运动函数导致最后子弹的运动函数臃肿不堪，而且难以完成既定任务。还是最后狠下心重新写了一遍子弹的运动函数，在这之后子弹的运动才符合预期。这提醒了我，有必要时刻注意精简自己的代码,如果代码变得越来越臃肿，最后肯定会带来意想不到的奇妙后果。

参考文献

［1］《C++程序设计基础（上）》——周霭如 林伟健 编著

［2］《C++ Primer plus》——Stephen Prata著 张海龙 袁国忠 译