

# 坦克大战个人 Report

--14 组 卢信虎

**Abstract** 经过前前后后 4 个月的努力(3 个月学习，巩固预备知识+1 个月的项目实现)，我们组的坦克大战游戏在大家齐心协力的工作下的到了很好的实现，已然是一部完成度极高的作品。

## 一， 工作梗概

Workload 声明：

坦克大战最终升级版的代码有 1867 行 (共 14 个函数)，我承担了其中 602 行的编写 (涵盖 8 个函数)，及其余代码的 debug 工作。

我在本组内负责的工作主要在于以下几个方面：

1°前期的游戏音效 (包括背景音，特效声，按钮声等) 的搜寻及插入；

2°编写了坦克的 type 函数，设计了多种坦克的型号；

3°编写了坦克的导弹发射函数 Jfire\_function；

4°使用指针型变量修改了 desk 函数，设计了游戏的信息显示界面；

5°修改了 tank 函数使得其结构更为优化，利于增加变量；

6°修改了 Move 函数和 fire 函数，starfire 使其结构适应坦克型号变量的插入；

7°debug 排除了界面显示信息残留问题和坦克移动过程中的错误。

## 二， 主要功能和亮点(my part)

### 1. 坦克 type 函数

真实世界中坦克的类型是多种多样的，考虑到这一点，在原来的 tank 函数中，

进行了彻底重构，使得坦克类型由原来的唯一确定变为可以由一个单独的函数 type 控制，这样一来，便可以随心所欲地绘制自己喜欢的坦克型号，使用一个整型变量 m 进行编号，并通过 switch 语句进行区别编写，期间需要注意 C++ 的控制台程序所能识别的 unicode 码的范围，有些图案是无法被识别的，如图是经过实验得到的可以被识别的字符，选取适合的进行绘制即可。



如下图所示为 type 函数的一部分，用以绘制坦克“铁血雄心”的部分代码，

其中使用光标位置确定函数

gotoxy 确定绘制点坐标，再按照

几何关系绘制相应图案，值得注

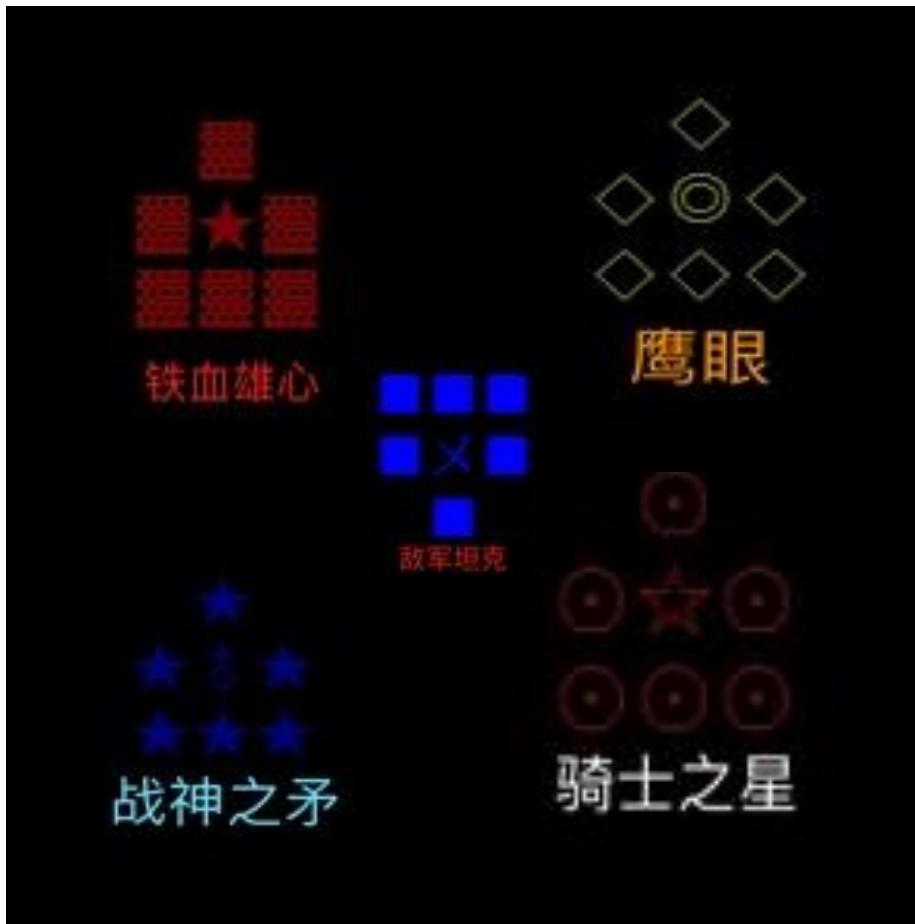
意的是，由于己方坦克供选颜色

固定 (color (n), n=

4 or 6 or 1)，需要通过条件语句

判断敌我坦克，进行区别绘制。

```
if (n == 4 || n == 6 || n == 1)
{
    printf("★");
    gotoxy(tank[0] - 1, tank[1]);
    printf("■");
    gotoxy(tank[0] + 1, tank[1]);
    printf("■");
    gotoxy(tank[0], tank[1] - 1);
    printf("■");
    gotoxy(tank[0], tank[1] + 1);
    printf("■");
}
else
{
    printf("×");
    gotoxy(tank[0] - 1, tank[1]);
    printf("■");
    gotoxy(tank[0] + 1, tank[1]);
    printf("■");
    gotoxy(tank[0], tank[1] - 1);
    printf("■");
    gotoxy(tank[0], tank[1] + 1);
    printf("■");
}
```



有了这个函数，游戏的趣味性被大大地提高。

## 2. 由 type 函数的设置衍生出的全体函数的结构调整

由于加入了坦克型号，开火函数（fire），移动函数（move），本阵星星开火函数（starfire）以及开场动画函数（brand）都需要进行结构上的调整，各自需添加一个整型变量 m，现已 brand（）开场函数的优化为例

修改前：

```
420 void brand()
421 {
422     int brandtank1[2] = { 10, 10 };
423     int brandtank2[2] = { 17, 4 };
424     int i;
425     PlaySound(TEXT("tankmove.wav"), NULL, SND_ASYNC | SND_NODEFAULT);
426     tank(brandtank1, 'd', 1);
427     tank(brandtank2, 's', 11);
428     Sleep(500);
429     for (i = 1; i <= 8; i++)
430     {
431         if (i == 8)
432         {
433             brandtank1[2] = Move(brandtank1, brandtank2, brandtank2, 'w', 'd', 1, 0);
434         }
435         else
436         {
437             PlaySound(TEXT("tankmove.wav"), NULL, SND_ASYNC | SND_NODEFAULT);
438             tank(brandtank1, 'd', t, c);
439             tank(brandtank2, 's', 1, 11);
440             Sleep(500);
441             for (i = 1; i <= 8; i++)
442             {
443                 if (i == 8)
444                 {
445                     brandtank1[2] = Move(brandtank1, brandtank2, brandtank2, 'w', 'd', c, t, 0);
446                 }
447                 else
448                 {
449                     brandtank1[2] = Move(brandtank1, brandtank2, brandtank2, 'd', 'd', c, t, 0);
450                 }
451                 Sleep(300);
452             }
453             PlaySound(NULL, NULL, NULL);
454             Sleep(100);
```

修改后：

除此之外，还通过修改 sleep (n) 中的 n 值，使得动画的进行与音效同步吻合

。

### 3. 导弹开火函数 (Jfire)

与空格开炮不同，导弹开火函数采用了按 J 发射的方式，通过一系列位置判定防止其打出战场边界，并且具有锁定特性，即：当判定炮弹发射路径必然经过敌方坦克时，全场静止，播放特殊音效—导弹出膛，随即导弹发射必定命中敌方

```
坦克,
658 int Jfire(int firetank[2], int tank1[2], int tank2[2], char c, char e1, char e2, int z, int t)
659 {
660     int n = 0, m = 8;
661     int p;
662     switch (c)
663     {
664     case 'w':
665         p = firetank[1] - 2;
666         if (p == 1)
667             ;
668         else
669         {
670             mcisendString(TEXT("play Sousou.wav wait"), 0, 0, 0);
671             for (p; p > 1; p--)
672             {
673                 gotoxy(firetank[0], p);
674                 color(m);
675                 printf("▲");
676                 Sleep(20);
677                 gotoxy(firetank[0], p);
678                 color(n);
679                 printf("■");
```

其中导弹形状根据发炮瞬间坦克朝向分别采用不同指向的三角形。增加了此函数之后，让火力输出方式多样化，增强了己方坦克的战力。

#### 4. 使用指针设计的全新游戏信息显示界面

当游戏开始前要求玩家填写姓名，选择游戏难易度，选择坦克型号，选择坦克颜色的时候，想到这些汉字，数字信息如果能显示在游戏界面的旁边，则会极大地增加游戏的信息化，于是修改初始化界面函数 (desk () → desk ( , , )，如下图所示) 申请指针变量记录下各型号关卡信息，使用 switch 语句在关卡等选择完毕后，于相应位置输出。

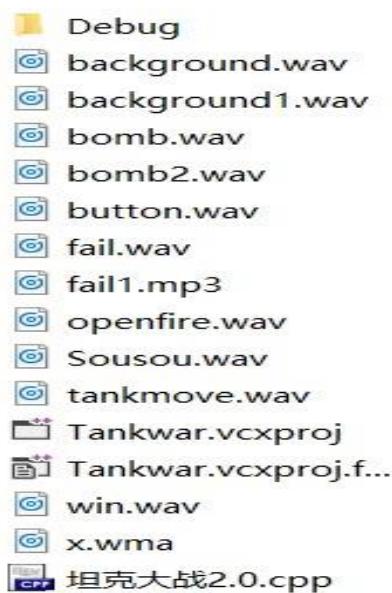
```
void desk(char name[10], const char *difficulty, const char *type)//初始化函数(初始化围墙、显示信息, )  
{  
    gotoxy(N + 3, 4);  
    color(4);  
    printf("难度: %s", difficulty);  
    gotoxy(N + 3, 5);  
    color(4);  
    printf("坦克型号: %s", type);  
  
    switch (m)  
    {  
        case 1:  
            type[0] = "铁血雄心";  
            break;  
        case 2:  
            type[0] = "鹰眼";  
            break;  
        case 3:  
            type[0] = "战神之矛";  
            break;  
        case 4:  
            type[0] = "骑士之星";  
            break;  
        case 0:  
            return 0;  
        default:  
    }  
}
```

(在游戏运行函数 (game) 中，记录  
相应信息)



## 5. 游戏音效

一个游戏的成功之关键很大程度上取决于音视觉效果的好坏。如何用 Win32 控制台实现较高的游戏体验也是颇费了一番功夫,在大量的试听筛选工作后,以下是我从互联网上搜索到的十分切合游戏运营环境的 music files,



其中包括游戏的背景音乐，开场音，结束音，履带声，发炮声，爆炸声，按纽音开场动画，结束动画音效等。使用 playsound 或 `mciSendString` 函数在恰当的语句间插入即可。区别在于 playsound 函数无需等到音乐文件播放结束再运行下一语句，且可以被 mci 函数播放的音乐覆盖，适合于背景音乐的调用，而 `mciSendString` 函数需等到音乐文件播放完毕再执行下一行命令，适用于导弹发射等各类简短音效文件的调用。以下是相关代码实例：

```
662     switch (c)
663     {
664     case 'w':
665         p = firetank[1] - 2;
666         if (p == 1)
667             ;
668         else
669         [
670             mciSendString(TEXT("play Sousou.wav wait"), 0, 0, 0);
671             for (p; p > 1; p--)
672             {
673                 gotoxy(firetank[0], p);
674                 color(m);
675                 printf("%c",火炮图标);

}
PlaySound(NULL, NULL, NULL);
Sleep(100);
fire(brandtank1, brandtank2, brandtank2, 'w', 's', 's', 3);
Sleep(1000);
PlaySound(TEXT("background1.wav"), NULL, SND_ASYNC | SND_NODEFAULT);
color(15);
for (i = 1; i <= 9; i++)//T
{
    if (i <= 5)
```

### 三，收获与感悟

#### 实现 project 过程：

- 1, 小组合作很重要，一个团队只有每个人都出力这个团队的项目才能够精彩；
- 2, 在这种教学模式下（先集中教学，再分小组做 project），既能很好地学习接受 C 语言的全套理论，又能有一个长期的时间去实践所学知识，是十分科学的学习模式。
- 3, 有一个长期实践去实现一个 project 的最大好处是我们能够在相当长的一段时间内对语言各个板块的知识十分熟悉---

因为要用，所以趋使我们去学；因为能用，所以促进我们所学。

## C 语言学习本身：

- 4, 很多新型的语言如， C++, Java , C# , J# , perl, python 等等都是衍生自 C 语言。掌握了 C 语言，可以说你就掌握了很多门语言。
- 5, Prof. A.T. 和徐昊老师上课的时候一定要认真听，老师讲的肯定是很重要的，错过了就是一大笔损失，认真听讲才可以提高学习效率。
- 6, 要学好 C 语言就要先懂得最基本的语法知识，看课本是必需的。我觉得看不懂也没关系，尽力去理解就好了，在对知识有了一个大致的了解过后，就要上机实践。学习 C 语言一定要动手，只看不做，眼高手低是不行的。最开始可以打书上的例题，熟悉程序，慢慢的开始试着编程。
- 7, 在编程之前，要把自己的想法写在纸上，如果是简单一点的程序不需要这样，如果程序比较复杂，就写下来，这样可以让思路更加清晰。
- 8, C 的基础知识学习以及在后期 project 的实现过程中，很好地掌握了各式各样函数的调用语法，能够熟练运用条件语句，循环语句及其嵌套实现各种功能，明白了许多同类或相似函数间的优缺点以及适用情况，如：  
if 跟 switch----  
若条件分支是多个而且条件的值是整数或是一个字符值时就会选

SWITCH 而不会选 IF。因为如果条件分支太多时要用 IF 语句，这样一定会出现 IF 的嵌套，而 “else if” 的嵌套越多时程序的开销就会越大，这样对整个程序的运行效率就大为降低。而 SWITCH 就不同，它只要比较一次就可以找出条件的结果，比起嵌套 IF 它的效率就大大的提高了很多。

不过 SWITCH 也有它的约束条件，就是它的条件值一定要是一个整型数或是一个字符值，所以碰到它不能解决的问题时我们也会通常使用 IF 语句，毕竟 IF 语句它使用起来也比较方便用的范围也比较广。

指针数组和数组指针---

指针数组，首先它是一个数组，数组的每个元素都是指针，可以理解为“存储指针的数组”的简称；数组指针，首先它是一个指针，它指向一个数组，在 32 位系统下永远只占 4 个字节，至于它指向的数组有多少个字节，并不知道，可以理解为“指向数组的指针”。如 int \*t1[5] 与 int (\*t2)[5]，首先，“[]” 的优先级比 “\*” 的优先级高，所以首先 t1 与 “[5]” 结合构成一个数组 t1[5]，int \* 为修饰数组的内容，所以数组元素是指向 int 类型的指针，所以这个是指针数组；“()” 的优先级比 “[5]” 的优先级高，“\*” 与 t2 结合构成一个指针变量，int 修饰数组的内容，即数组的每个元素，数组这里并没有名字，是个匿名数组，现在清楚了 t2 是一个指针，它指向一个包含 5 个 int 类型数据的数组，即为数组指针。

## 四，意见与建议

本学期 C 语言的课程圆满结束了，在老师的辛勤教导和自己的刻苦学习中，收获颇多，除此之外还有些许对于课程的建议：

- 1, 可以适当的增加课堂测试甚至是期末测试，结合 project，综合评定成绩，这样可以给学的好的同学以展示自己的机会，也能起到鞭策大家好好学习编程的作用。
- 2, 在讲授完基础知识后，可在开学后每星期的课程里增设更多小的项目或小的作业，从而使我们更连续地掌握相关知识，不容易忘。
- 3, 中英双语教学的模式很棒希望唐班的学弟学妹们也可以享受到这种优质的教育资源！

最 终 版 游 戏 讲 解 演 示 视 频 :

