# 项目经理带你 - 零基础手写坦克大战

# 课程介绍

## 1.1 讲师介绍

**奇牛学院Martin老师**



中文名： 王鹏程

个人简介： 企业邮局架构师，老板邮局的首席架构师，曾任职于多家上市公司。

十二年行业（C/C++）开发经验，一直专注于Linux服务器端高并发应用程序

和分布式大数据存储系统的开发和架构。

了解和研究过多款C/C++优秀开源软件（Nginx,TFS等）的框架和代码实现，

擅长在其基础上进行二次开发。

QQ: 2684436901

## 1.2 项目介绍

**2.1. 项目需求**

* 实现1款和经典的《90坦克大战》一样的游戏，任务是消灭敌对坦克，保护己方领地。防止敌方打破你的老窝围墙而把你的鹰打坏。



**2.2. 学习目标**

* 回顾经典，按照软件开发的标准流程，零基础开发第一个完整项目，快速提高编程能力和水平，真正理解软件开发的概念和基本流程！

**2.3. 授课方式**

* 手把手方式教大家完整的实现项目所有源码并进行讲解，不懂之处，随时可以咨询Martin老师答疑！QQ: 2684436901

# 项目准备

* 1. **环境安装**

请参考**《**项目经理带你-零基础学习C++**》 第6节 C++开发环境** 安装好开发工具vc2010 ,课程链接：<https://ke.qq.com/course/377567?tuin=cc02ada>

* 1. **图形库安装**

1. 下载easyx 图形库 网址：<https://www.easyx.cn/>
2. 安装
   1. **开发环境测试**

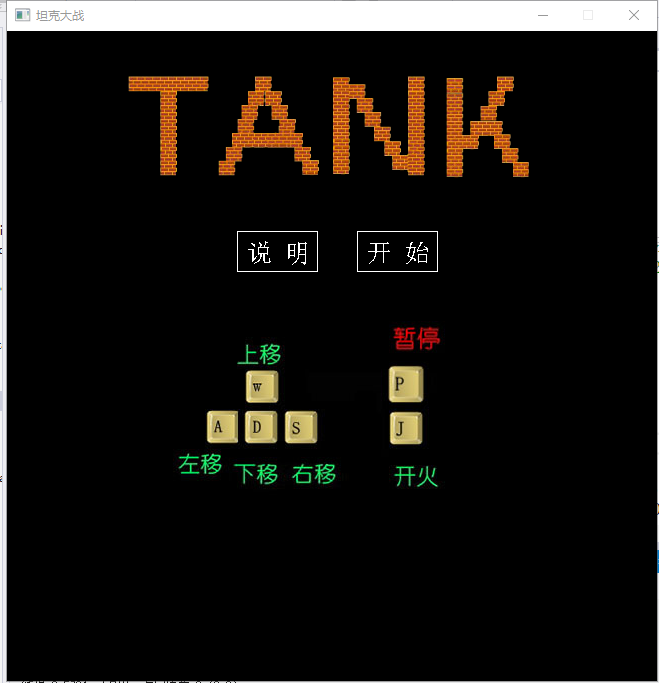
|  |
| --- |
| #include <graphics.h>  void main()  {  initgraph(666, 666); //定义画布大小 666\*666  system("pause");    } |

# 项目启动

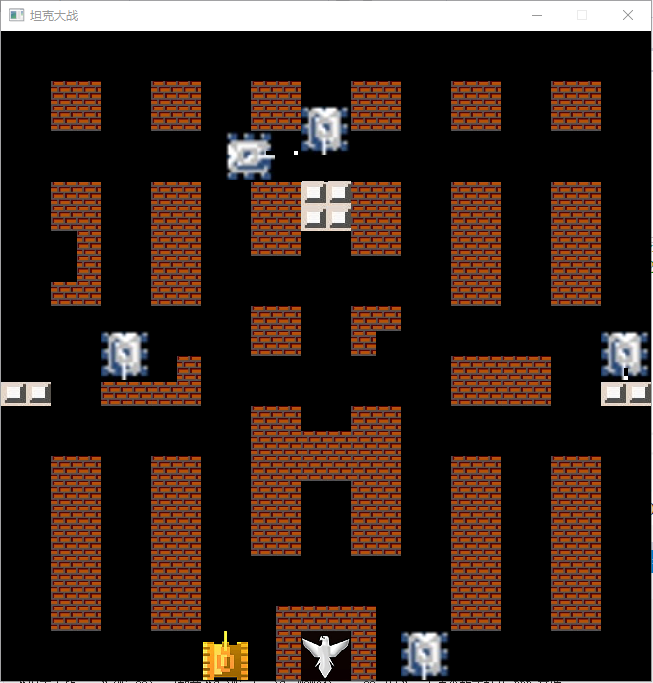
* 1. **模块划分**

（作用：1.化繁为简 2.适合团队协作 3.高质量代码）

****

****

开 始

****

游戏中

****

结 束

* 1. **资源准备**

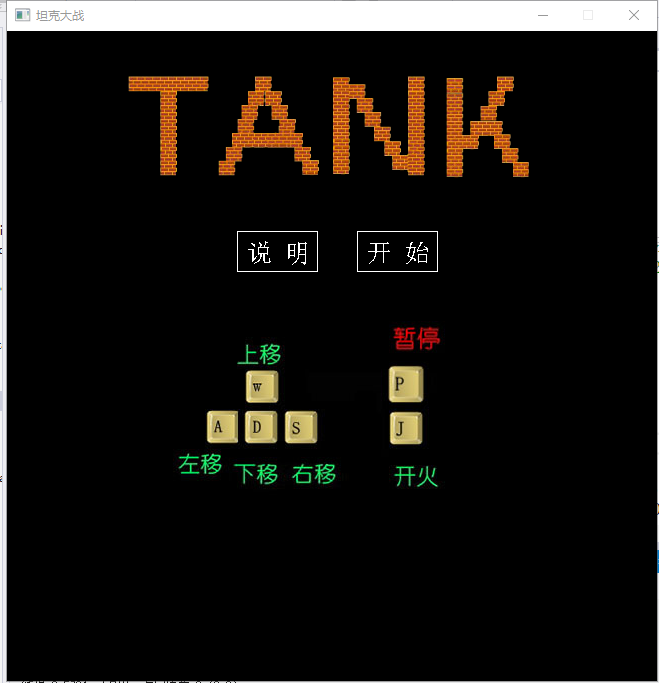
美工素材请从奇牛C/C++学习交流群文件夹下载

群号：875300321

下载路径: 群文件夹 => 坦克大战资源 => 坦克大战\_图片&音乐.rar

# 项目实现

* 1. **模块1 - 开始场景**

****

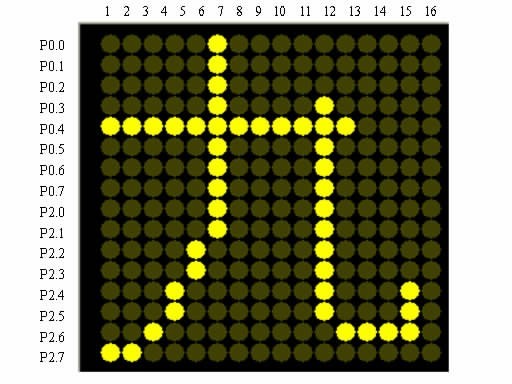
“搭台唱戏”要素：

* “戏台” - 绘图环境
* Logo - 美工图片，游戏标志
* 按钮 - 实现“说明”和“开始”导航
* 说明 - 美工图片，操作说明
  + 1. **戏台实现**

|  |
| --- |
| #include <graphics.h>  void main()  {  initgraph(650, 650); //定义画布大小 650\*650  system("pause");    } |

* **项目精讲 - 像素**

像素是整个图像中不可分割的单位或者是元素，每个像素近似一个小方块，这些小方块都有一个明确的位置和被分配的色彩数值（显示不同的颜色）



* **项目精讲 - 分辨率**

分辨率（屏幕分辨率）是屏幕[图像](https://baike.baidu.com/item/%E5%9B%BE%E5%83%8F/773234" \t "https://baike.baidu.com/item/_blank)的精密度，是指[显示器](https://baike.baidu.com/item/%E6%98%BE%E7%A4%BA%E5%99%A8/362722" \t "https://baike.baidu.com/item/_blank)所能显示的[像素](https://baike.baidu.com/item/%E5%83%8F%E7%B4%A0/95084" \t "https://baike.baidu.com/item/_blank)有多少，分辨率越大，单位面积内分布的像素点就越多，画面就越[精细](https://baike.baidu.com/item/%E7%B2%BE%E7%BB%86/1312083" \t "https://baike.baidu.com/item/_blank)

如： 14英寸笔记本屏幕分辨率 1280 x 960 表示的意义是屏幕是由 1280 乘以 960 = 1228800 个像素点组成，其中宽占1280 个像素，高占960 像素

1280像素

960像素

* + 1. **显示LOGO**

|  |
| --- |
| //显示 logo  IMAGE logo\_img;  loadimage(&logo\_img, \_T("logo.bmp"), 433, 147);  putimage(110, 20, &logo\_img); |

戏台坐标系(650 x 650)：

X 坐标

坐标:(650, 0)

坐标:(0, 0)

Y 坐标

坐标:(650, 650)

坐标:(0, 650)

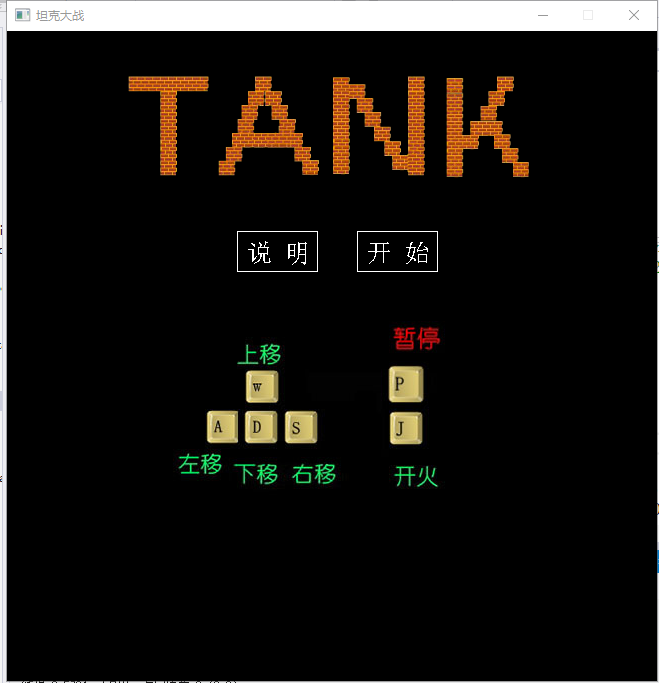
|  |
| --- |
| void menu(){  //显示 logo  IMAGE logo\_img;  loadimage(&logo\_img, \_T("logo.bmp"), 433, 147);  putimage(110, 20, &logo\_img);  //实现导航按钮  setlinecolor(WHITE);  setfillcolor(BLACK);  fillrectangle(230, 200, 310, 240);  settextstyle(25, 0, \_T("宋体"));  outtextxy(240, 210, \_T("说 明"));  fillrectangle(350, 200, 430, 240);  outtextxy(360, 210, \_T("开 始"));  MOUSEMSG mouse;  IMAGE illustrate\_img;  loadimage(&illustrate\_img, \_T("illustrate.jpg"), 300, 300);  while(1==1){  mouse=GetMouseMsg();  switch(mouse.uMsg){  case WM\_MOUSEMOVE:  if((mouse.x>230 && mouse.x<310) && (mouse.y>200 && mouse.y<240)){  putimage(150, 250, &illustrate\_img);  }else {  solidrectangle(150, 250, 450, 550);  }  break;  case WM\_LBUTTONDOWN:  if((mouse.x >350 && mouse.x<430) && (mouse.y>200 && mouse.y<240)){  cleardevice();  return ;  }  }  }  } |

* + 1. **导航按钮实现**
* 使用矩形绘制函数和文字输出函数实现按钮显示效果

附：EasyX 帮助文档

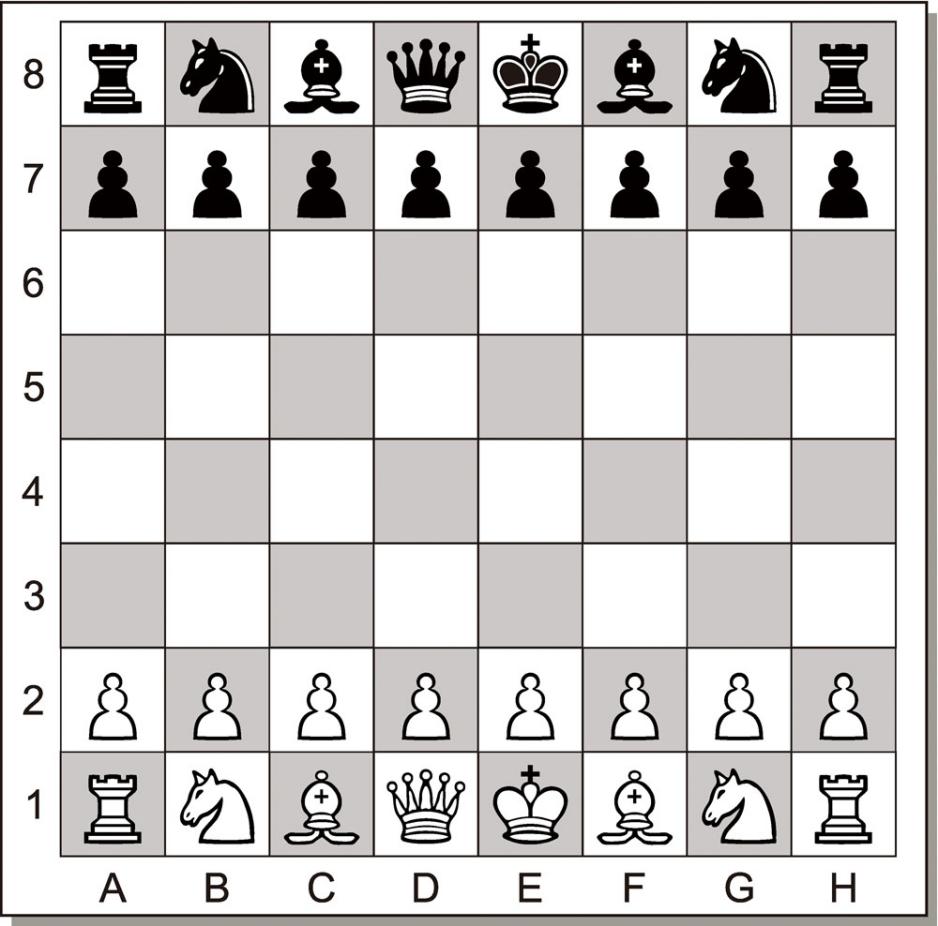
群号：875300321

下载路径: 群文件夹 => 坦克大战资源 => EasyX\_Help.chm

****

* 使用鼠标事件实现导航效果

1. 当鼠标移到 “说明” 按钮时，显示操作说明，当鼠标离开隐藏
2. 当鼠标点击“开始”按钮时，就进入 **游戏场景**
   1. **模块2 - 游戏场景**
      1. **地图初始化**

****

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **名称** | **王** | **后** | **车** | **象** | **马** | **兵** |
| **角色** | **国王** | **皇后** | **战车** | **主教** | **骑士** | **禁卫军** |

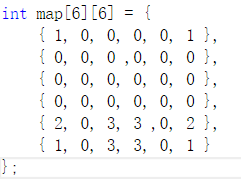
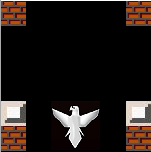
**地图表示:**

使用二维数组

* 游戏道具显示（墙、老鹰、我方坦克、敌方坦克、子弹）
* 便于程序控制敌方坦克前进，控制子弹移动和判断子弹击中目标等

**道具表示:**

可消除墙为1，不可消除墙为2，老鹰(3,4),敌方坦克 100 - 109,我方坦克200

**代码实现**

|  |
| --- |
| //定义地图数组  int map[26][26] = {  { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },  { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },  { 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0 },  { 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0 },  { 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0 },  { 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0 },  { 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 2, 2, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0 },  { 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 2, 2, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0 },  { 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0 },  { 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0 },  { 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0 },  { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },  { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },  { 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1 },  { 2, 2, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 2, 2 },  { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },  { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },  { 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0 },  { 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0 },  { 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0 },  { 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0 },  { 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0 },  { 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0 },  { 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0 },  { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 3, 3, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 },  { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 3, 3, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 }  };  void menu();  void init\_map();  void init\_map\_2(int \*map, int rows, int cols);  int main(void){  //搭建舞台  initgraph(650,650);  //开始场景，显示菜单  menu();  //初始化地图  init\_map\_2(&map[0][0], 26, 26);  system("pause");  }  /\*  初始化地图，可消除墙为1，不可消除墙为 2，老鹰 (3 ,4)  \*/  void init\_map(){  int i, j;  IMAGE img\_home, img\_wall\_1, img\_wall\_2;  loadimage(&img\_home, \_T("home.jpg"), 50, 50);// 老鹰  loadimage(&img\_wall\_1, \_T("wall1.jpg"), 25, 25);//不可消除的墙  loadimage(&img\_wall\_2, \_T("wall2.jpg"), 25, 25);//可消除的墙  for(i=0; i<26; i++){  for(j=0; j<26; j++){  if(map[i][j] == 1){  putimage(25\*j, 25\*i, &img\_wall\_2);  }else if(map[i][j] == 2){  putimage(25\*j, 25\*i, &img\_wall\_1);  }else if(map[i][j] == 3){  putimage(25\*j, 25\*i, &img\_home);  map[i][j] = 4;  map[i][j+1] = 4;  map[i+1][j] = 4;  map[i+1][j+1] = 4;  }  }  }  }  //逼格更高、兼容性更好的初始化地图函数  void init\_map\_2(int \*map, int rows, int cols){  int i, j;  IMAGE img\_home, img\_wall\_1, img\_wall\_2;  loadimage(&img\_home, \_T("home.jpg"), 50, 50);// 老鹰  loadimage(&img\_wall\_1, \_T("wall1.jpg"), 25, 25);//不可消除的墙  loadimage(&img\_wall\_2, \_T("wall2.jpg"), 25, 25);//可消除的墙  for(i=0; i<rows; i++){  for(j=0; j<cols; j++){  if(\*(map+cols\*i+j) == 1){  putimage(25\*j, 25\*i, &img\_wall\_2);  }else if(\*(map+cols\*i+j) == 2){  putimage(25\*j, 25\*i, &img\_wall\_1);  }else if(\*(map+cols\*i+j) == 3){  putimage(25\*j, 25\*i, &img\_home);  //以下代码需要考虑数组越界，该如何处理？  \*(map+cols\*i+j) = 4;  \*(map+cols\*i+(j+1)) = 4;  \*(map+cols\*(i+1)+j) = 4;  \*(map+cols\*(i+1)+(j+1)) = 4;  }  }  }  } |

* + 1. **我方坦克实现**

**坦克结构体定义**

|  |
| --- |
| enum DIRECTION{  UP,  DOWN,  LEFT,  RIGHT  };  //坦克结构体  struct tank\_s{  int x; //坦克在地图数组中所在列  int y; //坦克在地图数组中所在的行  DIRECTION direction; //坦克的方向，上、下、左、右  int live; //是否生存 1-活着 0-挂了  }; |

**我方坦克显示**

|  |
| --- |
| tank\_s my\_tank;  IMAGE my\_tank\_img[4];  //加载我方坦克的图片  loadimage(&my\_tank\_img[UP],\_T("tank\_up.jpg"),50,50);  loadimage(&my\_tank\_img[DOWN],\_T("tank\_down.jpg"),50,50);  loadimage(&my\_tank\_img[LEFT],\_T("tank\_left.jpg"),50,50);  loadimage(&my\_tank\_img[RIGHT],\_T("tank\_right.jpg"),50,50);  //设定我方坦克的出场的位置  my\_tank.x = 8;  my\_tank.y = 24;  my\_tank.live = 1;  my\_tank.direction = UP;  map[my\_tank.y][my\_tank.x] =200;  map[my\_tank.y][my\_tank.x+1] =200;  map[my\_tank.y+1][my\_tank.x] =200;  map[my\_tank.y+1][my\_tank.x+1] =200;  putimage(my\_tank.x \* 25, my\_tank.y \* 25, &my\_tank\_img[my\_tank.direction]); |

**热键控制(aswd )**

|  |
| --- |
| /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \*实现游戏场景  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  void play(){  tank\_s my\_tank;  IMAGE my\_tank\_img[4];  int key;  //加载我方坦克的图片  loadimage(&my\_tank\_img[UP],\_T("tank\_up.jpg"),50,50);  loadimage(&my\_tank\_img[DOWN],\_T("tank\_down.jpg"),50,50);  loadimage(&my\_tank\_img[LEFT],\_T("tank\_left.jpg"),50,50);  loadimage(&my\_tank\_img[RIGHT],\_T("tank\_right.jpg"),50,50);  //设定我方坦克的出场的位置  my\_tank.x = 8;  my\_tank.y = 24;  my\_tank.live = 1;  my\_tank.direction = UP;  set\_prop\_map(my\_tank.x, my\_tank.y, 200);  putimage(my\_tank.x \* 25, my\_tank.y \* 25, &my\_tank\_img[my\_tank.direction]);  while(1){  if(\_kbhit()){  key = \_getch();  switch(key){  case 'a': //左  if((my\_tank.x-1)>=0 && map[my\_tank.y][my\_tank.x-1] ==0 &&map[my\_tank.y+1][my\_tank.x-1] ==0){//左边是空地  my\_tank.direction = LEFT;  tank\_walk(&my\_tank, LEFT, &my\_tank\_img[my\_tank.direction]);  }  break;  case 'w': //上  if((my\_tank.y-1)>=0 && map[my\_tank.y-1][my\_tank.x] ==0 &&map[my\_tank.y-1][my\_tank.x+1] ==0){//上边是空地  my\_tank.direction = UP;  tank\_walk(&my\_tank, UP, &my\_tank\_img[my\_tank.direction]);  }  break;  case 's': //下  if((my\_tank.y+2)<=25 && map[my\_tank.y+2][my\_tank.x] ==0 &&map[my\_tank.y+2][my\_tank.x+1] ==0){//下边是空地  my\_tank.direction = DOWN;  tank\_walk(&my\_tank, DOWN, &my\_tank\_img[my\_tank.direction]);  }  break;  case 'd': //右  if((my\_tank.x+2)<=25 && map[my\_tank.y][my\_tank.x+2] ==0 &&map[my\_tank.y+1][my\_tank.x+2] ==0){//右边是空地  my\_tank.direction = RIGHT;  tank\_walk(&my\_tank, RIGHT,&my\_tank\_img[my\_tank.direction]);  }  break;  case 'j': //开火  break;  case 'p': //暂停  system("pause");  break;  default: //其他键盘输入无须处理  break;  }  }  Sleep(10);  }  } |

**坦克移动**

|  |
| --- |
| void set\_prop\_map(int x, int y, int val){  map[y][x] = val;  map[y][x+1] = val;  map[y+1][x] = val;  map[y+1][x+1] = val;  }  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \*控制坦克按相应的方向前进一步  \*返回值：失败 - 0 成功 -1  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  int tank\_walk(tank\_s \*tank, DIRECTION direction, IMAGE \*img){  int new\_x = tank->x;  int new\_y = tank->y;  if(direction == UP){  new\_y -= 1;  }else if(direction == DOWN){  new\_y += 1;  }else if(direction == LEFT){  new\_x -= 1;  }else if(direction == RIGHT){  new\_x += 1;  }else {  return 0; //无效的方向  }  set\_prop\_map(tank->x, tank->y, 0);  setfillcolor(BLACK);  solidrectangle(tank->x\*25,tank->y\*25, tank->x\*25+50, tank->y\*25+50);  set\_prop\_map(new\_x, new\_y, 200);  tank->x = new\_x;  tank->y = new\_y;  putimage(tank->x \* 25, tank->y \* 25, img);  return 1;  } |

* + 1. **子弹飞行控制实现**

**子弹结构体定义**

|  |
| --- |
| //子弹结构体  struct bullet\_s{  int pos\_x; //子弹在“戏台”上的横坐标  int pos\_y; //子弹在“戏台”上的纵坐标  DIRECTION direction; //子弹方向  int status; //子弹是否存在  }; |

**子弹热键控制 (j)**

|  |
| --- |
| if(my\_bullet.status == 0){  if(my\_tank.direction == UP){  my\_bullet.pos\_x = my\_tank.x \* 25 + 23;  my\_bullet.pos\_y = my\_tank.y \* 25 -3;  }else if(my\_tank.direction == LEFT){  my\_bullet.pos\_x = my\_tank.x \* 25 -3;  my\_bullet.pos\_y = my\_tank.y \* 25 +23;  }else if(my\_tank.direction == DOWN){  my\_bullet.pos\_x = my\_tank.x \* 25 + 23;  my\_bullet.pos\_y = my\_tank.y \* 25 + 50;  }else if(my\_tank.direction == RIGHT){  my\_bullet.pos\_x = my\_tank.x \* 25 + 50;  my\_bullet.pos\_y = my\_tank.y \* 25 + 23;  }  my\_bullet.direction = my\_tank.direction;  my\_bullet.status = 1;  } |

**子弹运行和碰撞检测**

|  |
| --- |
| void bullet\_action(bullet\_s \*bullet){  int x,y,x1,y1; //子弹目前所在的二维数组中的坐标  x = bullet->pos\_x/25;  y = bullet->pos\_y/25;  //1.擦除上一次绘制的子弹  setfillcolor(BLACK);  solidrectangle(bullet->pos\_x, bullet->pos\_y, bullet->pos\_x+3, bullet->pos\_y+3);  //2.根据方向计算子弹在“戏台”上的坐标  if(bullet->direction == UP){  bullet->pos\_y -= 2;  x1 = x+1;  y1 = y;  }else if(bullet->direction == DOWN){  bullet->pos\_y += 2;  x1 = x+1;  y1 = y;  }else if(bullet->direction == LEFT){  bullet->pos\_x -= 2;  x1 = x;  y1 = y+1;  }else if(bullet->direction == RIGHT){  bullet->pos\_x += 2;  x1 = x;  y1 = y+1;  }else{  return;  }  if(bullet->pos\_x<0 || bullet->pos\_x>650 || bullet->pos\_y<0 || bullet->pos\_y>650){  bullet->status = 0;  return;  }  //碰撞检测  if(map[y][x] == 4 || map[y1][x1] == 4){  return ;  }  if(map[y][x]== 1){//子弹击中可消除的墙  map[y][x]= 0;  bullet->status = 0;  setfillcolor(BLACK);  solidrectangle(x\*25, y\*25, x\*25+25, y\*25+25);  }else if(map[y][x]== 2){  bullet->status = 0;  }  if(map[y1][x1]== 1){//子弹击中可消除的墙  map[y1][x1]= 0;  bullet->status = 0;  setfillcolor(BLACK);  solidrectangle(x1\*25, y1\*25, x1\*25+25, y1\*25+25);  }else if(map[y1][x1]== 2){  bullet->status = 0;  }  //3.重新绘制子弹  if(bullet->status == 1){  setfillcolor(WHITE);  solidrectangle(bullet->pos\_x, bullet->pos\_y, bullet->pos\_x+3, bullet->pos\_y+3);  }  } |

* + 1. **敌方坦克实现**

**坦克出场**

1. 首先出场3台坦克，然后每隔小段时间出场一辆坦克

|  |
| --- |
| tank\_s enemy\_tank[ENEMY\_NUM]; //敌方坦克  bullet\_s enemy\_bullet[ENEMY\_NUM];//敌方坦克发射的子弹  IMAGE enemy\_tank\_img[4];  //加载敌方坦克的图片  loadimage(&enemy\_tank\_img[UP],\_T("enemy\_tank\_up.jpg"),50,50);  loadimage(&enemy\_tank\_img[DOWN],\_T("enemy\_tank\_down.jpg"),50,50);  loadimage(&enemy\_tank\_img[LEFT],\_T("enemy\_tank\_left.jpg"),50,50);  loadimage(&enemy\_tank\_img[RIGHT],\_T("enemy\_tank\_right.jpg"),50,50);  //设置敌方坦克出场的位置  for(int i=0; i<ENEMY\_NUM; i++){  if(i%3 == 0){  enemy\_tank[i].x = 0;  }else if(i%3 == 1){  enemy\_tank[i].x = 12;  }else if(i%3 == 2){  enemy\_tank[i].x = 24;  }  enemy\_tank[i].direction = DOWN;  enemy\_tank[i].y = 0;  enemy\_tank[i].live = 1;  set\_prop\_map(enemy\_tank[i].x, enemy\_tank[i].y, 100+i);  enemy\_bullet[i].status = 0;  }  //前3辆坦克闪亮登场  tank\_walk(&enemy\_tank[0], DOWN, &enemy\_tank\_img[DOWN], 0);  tank\_walk(&enemy\_tank[1], DOWN, &enemy\_tank\_img[DOWN], 0);  tank\_walk(&enemy\_tank[2], DOWN, &enemy\_tank\_img[DOWN], 0); |

**选路实现**

**炮击实现**

* 1. **模块3 - 结束场景**