

高级语言程序设计 2-2

实验报告

南开大学 计算机大类

姓名：石爽

学号：2312848

班级：0972

2024 年 5 月 14 日

目录

高级语言程序设计大作业实验报告	3
一. 作业题目	3
二. 开发软件	3
三. 课题要求	3
四. 主要流程	错误! 未定义书签。
1. 设计目标	3
1) 模仿设计四类游戏模型/属性:	3
2. 算法或公式	4
(1) 正三角形	错误! 未定义书签。
(2) 正方形	错误! 未定义书签。
(3) 正五边形	错误! 未定义书签。
(4) 正六边形	错误! 未定义书签。
(5) 圆形	错误! 未定义书签。
3. 单元测试	错误! 未定义书签。
五. 单元测试	错误! 未定义书签。
测试结果	错误! 未定义书签。
六. 收获	5
1. 虚析构造函数	错误! 未定义书签。
2. 单元测试	错误! 未定义书签。

高级语言程序设计大作业实验报告

一. 作业题目

坦克大战

二. 开发软件

Qt5.14.2

三. 课题要求

使用 C++ 语言完成一个图形化的小程序

四. 设计内容

1. 设计目标

1) 模仿设计四类游戏模型/属性:

- 游戏地形
 - ✧ 砖墙: 除四周围墙外均可被一级子弹打穿
 - ✧ 海水: 坦克和子弹均不可穿越
 - ✧ 铁墙: 可被二级子弹打穿
 - ✧ 森林: 坦克和子弹均可穿越, 但显示上会被森林覆盖
 - ✧ 基地: "★"表示基地中心, 被打中则游戏结束
- 游戏道具
 - ✧ 坦克: 玩家可以增加一条命
 - ✧ 炸弹: 全屏敌人遭到爆炸攻击, 全数毁灭
- 三种坦克属性:
 - ✧ 生命值: 即 HP, 表示坦克可以被子弹命中多少次
 - ✧ 攻击力: 一方面表现为子弹种类, 一级子弹只能打穿砖墙, 二级子弹可打穿铁墙; 另一方面表现为子弹射速, 即子弹的移动速度
 - ✧ 移动速度: 坦克的移动速度
- 五种类型的坦克:
 - ✧ 轻型坦克: 移动速度普通, 一级子弹, 子弹射速普通, 生命值为 1
 - ✧ 装甲车: 移动速度快, 一级子弹, 子弹射速普通, 生命值为 2
 - ✧ 反坦克炮: 移动速度普通, 二级子弹, 子弹射速快, 生命值为 2
 - ✧ 重型坦克: 移动速度普通, 一级子弹, 子弹射速普通, 生命值为 3
 - ✧ 玩家坦克: 移动速度普通, 一级子弹, 子弹射速普通, 生命值为 10

2) 实现基本游戏逻辑：

- 按照一定的策略刷出敌军坦克，敌军坦克随机移动并攻击。玩家控制的坦克在基地旁边刷出
- 玩家控制的坦克通过灵活移动保卫基地不被摧毁并消灭敌军坦克
- 游戏过程中在地图上会随机刷出道具，玩家可通过道具强化自己
- 失败条件：基地被摧毁或者玩家生命耗尽
- 胜利条件：消灭所有敌军坦克

3) 实现当前局的状态栏

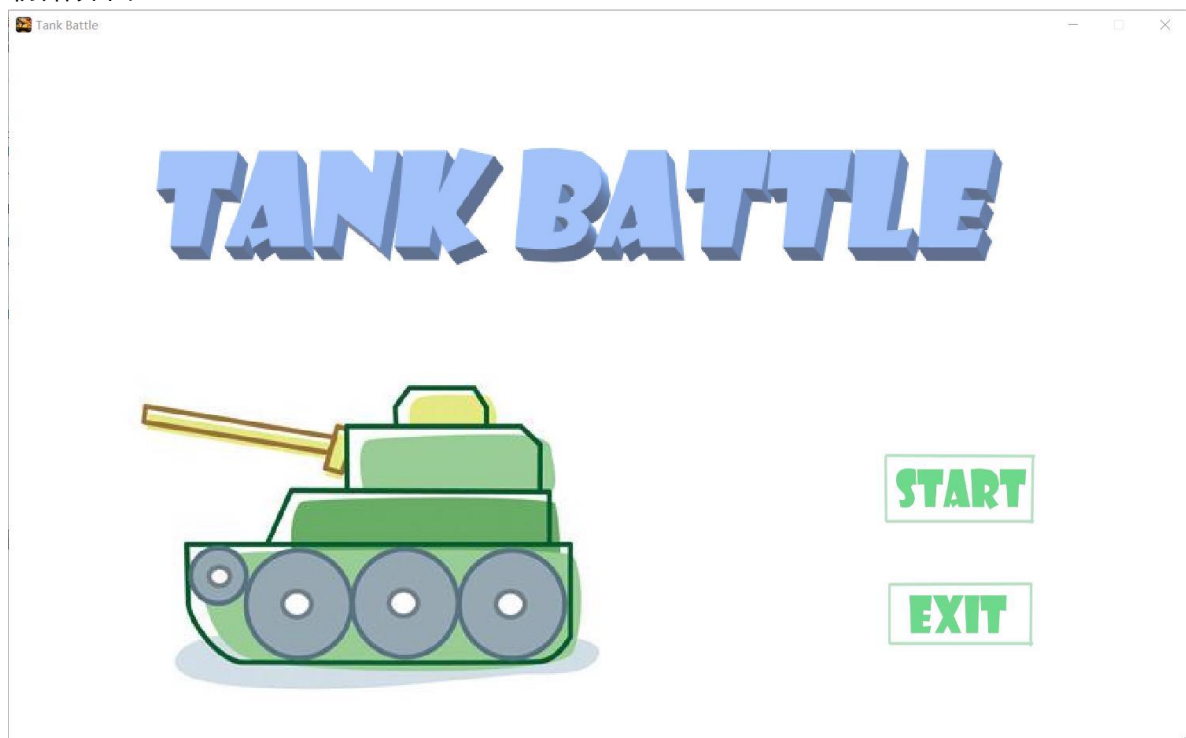
- 显示玩家生命值、敌军剩余坦克数量
- 游戏结束时显示：
 - ✧ 胜负：赢了显示“Win!”，输了显示”Game Over”;
 - ✧ 得分：根据玩家消灭的敌方坦克的种类和数量计算得分

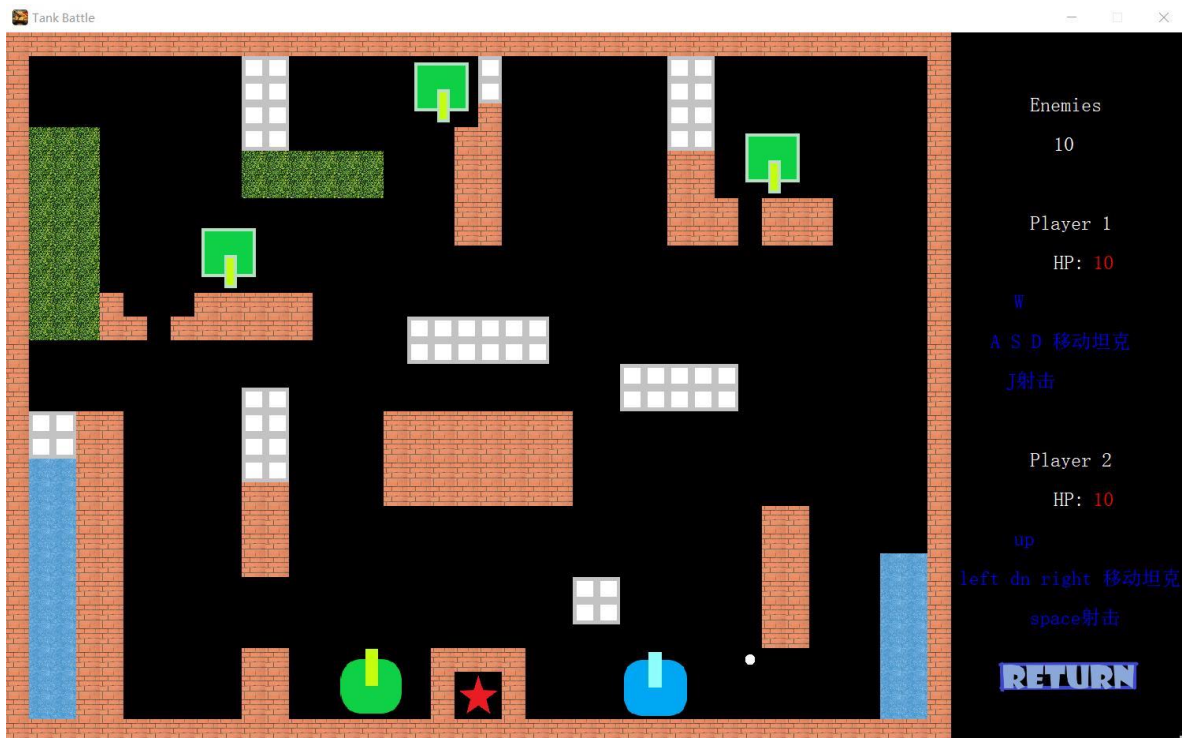
4) 支持双人模式

- 游戏开始时选择模式，游戏结束时显示计分板

2. 基本实现效果

初始界面：





五．设计思路

1.窗口交互和图形界面的实现

1) 窗口的实现:

设计了两个窗口类: MainWindow 和 subWindow 类

MainWindow 负责游戏初始界面, subWindow 负责游戏开始后的界面

窗口类可以在 ui 文件中可视化添加控件, 并且可以接收控件点击消息、键盘消息等, 可以通过 为窗口类添加相应事件处理函数来实现交互。

```
void keyPressEvent(QKeyEvent *ev); // 键盘事件处理函数  
  
void on_pushButton_clicked(); // pushbutton 被按下处理函数
```

2) 图片和文字的显示:

主要通过 QPainter 实现图片和文字的绘制, 在以下函数中实现:

```
void paintEvent(QPaintEvent *event);
```

关于地图的绘制: 这里将界面划分为一个个小格, 通过 map 数组记录每个小格需要显示的图片, 所以在函数 paintEvent 中只需读取 map 数组中的数据即可知道在相应位置应该绘制哪张图片。坦克, 子弹或地形发生改变时, 调用函数 update() 即可实现地图的更新。

3) 界面的切换:

subWindow 中会涉及多个界面, 如模式选择界面、游戏进行时的界面、游

戏结束后的计分板 界面。这里通过设置 flag 标志位让 paintEvent 函数知道应该读取哪些数据来绘制界面。例如： 游戏结束条件触发时设标志位 endflag 为 1，表示游戏结束，此时 paintEvent 函数读取玩家击败敌方坦克的数量和分数，并在相应位置显示。

2.游戏逻辑的控制

原来的控制台版本通过设计 Control 类来控制游戏的初始、进行和结束；但在 Qt 中因为键盘 消息、时钟信号都由窗口类捕捉，所以把游戏逻辑控制部分移到了 subwindow 类中。

1) 地图上的坦克、子弹变量：

在 subwindow 类的数据成员中存储着地图上现存的坦克指针（包括地方坦克和玩家坦克），而 Tank 类的数据成员中有其控制的子弹，通过遍历这些坦克及其控制的子弹并调用其 move 函数实现这些模型的运动。

约定同一时刻存在的敌方坦克数量最多为 3，当有敌方坦克被摧毁并且有备用坦克时，刷新新的敌方坦克。玩家被击中时退回其出生地。

2) 坦克、子弹的移动：

通过设置几个时间间隔不同的时钟 QTimer，并将时钟信号与槽函数..._move 连接，每隔一段时间调用坦克或子弹的 move 函数，实现坦克和子弹移速的控制。

3) 整体逻辑的控制：

最最主要的游戏逻辑控制在 subwindow 中的 bullet_move 函数中，因为坦克的消亡、玩家坦克血量的减少、基地被摧毁等与游戏结束触发条件相关的事件均由子弹引发。

而当基地被摧毁、现存地方坦克数量为 0、玩家生命值耗尽时游戏结束，设 endflag 为 1，paintEvent 函数会显示胜负情况，计算并显示玩家得分情况。

3.地图的设计

通过 int 型的二维数组 map 存储地图上的模型信息，不同的数字表示占用当前位置的不同游戏模型，其中

1~10 表示地形：空地，砖墙，铁墙，基地，海洋，森林等

10~20 表示敌方坦克的 id

20~30 表示玩家坦克的 id

30~40 表示敌方坦克射出的子弹 id，减去 20 即可得敌方坦克 id

40~50 表示玩家坦克射出的子弹 id，减去 20 即可得玩家坦克 id

（以上数字区间中不一定所有数都被用到）

通过这样的区间约定可以有效判断子弹或坦克移动时遇到的是哪一类游戏模型并作出相应的反应

4.坦克的设计

坦克可以分为两类：敌方坦克和玩家坦克，这两类坦克有很多共同属性和功能，但又有不同之处（如：敌方坦克通过一定策略自动移动射击，玩家坦克通过键盘输入移动射所以设计了 `Tank` 基类来包含它们的共同属性和功能，在其派生类 `Enemy` 和 `Player` 中对不同之处做各自的调整。

●坦克的移动

通过 `void Tank::basic_move()`；函数实现。根据坦克目前的方向决定下一刻的坦克坐标，将之前坦克所在位置区域对应的 `map[i]`置零，并打印空格；根据目前坦克 位置打印坦克模型。

```
void print_tank();//在地图上打印坦克
```

```
void delete_tank();//使坦克在地图上消失
```

●坦克的射击

通过 `void Tank::shoot()`；实现，约定一个坦克在同一时刻只能拥有一颗子弹，所以只有当坦克的数据成员 `bullet` 为 `NULL` 时，才可以调用 `shoot` 函数产生新子弹，并根据坦克 `id` 决定子弹 `id`，达到可通过子弹 `id` 知道其坦克 `id` 的目的。

●坦克的属性

坦克的移速，其射出的子弹的移速都通过调节其 `sleeptime` 来控制，`sleeptime` 越大，调用其 `move` 函数的时间间隔越长，表现为其移动速度越慢。

5.子弹的设计

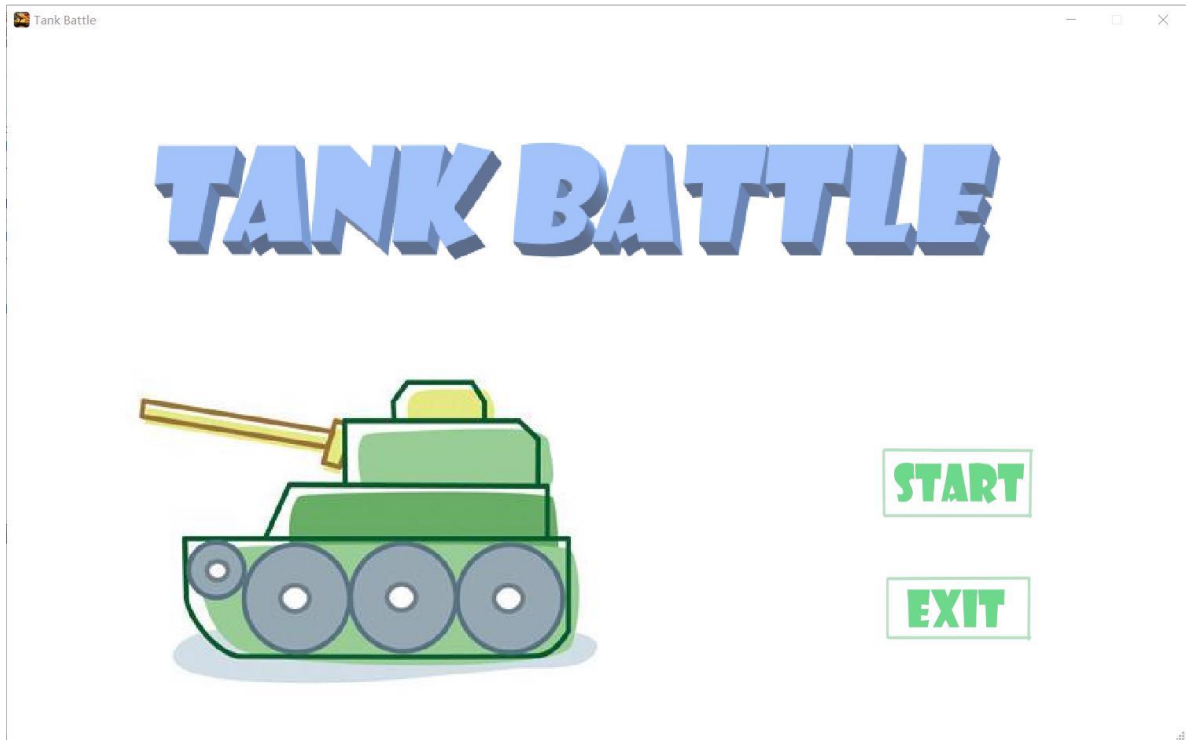
●子弹的移动

子弹一直沿坦克射出子弹时的方向移动，直至碰到障碍物时子弹消亡。

如果是地形障碍，根据地形种类和子弹种类决定该地形障碍是否消失；如果是坦克，将其 `id` 记录在 `bullet` 类的数据成员 `hurt_id` 中，坦克生命值减一，生命值为 0 时 该坦克死亡；如果是子弹，也将其 `id` 记录在 `hurt_id` 中，并令该子弹消亡。

六．功能介绍

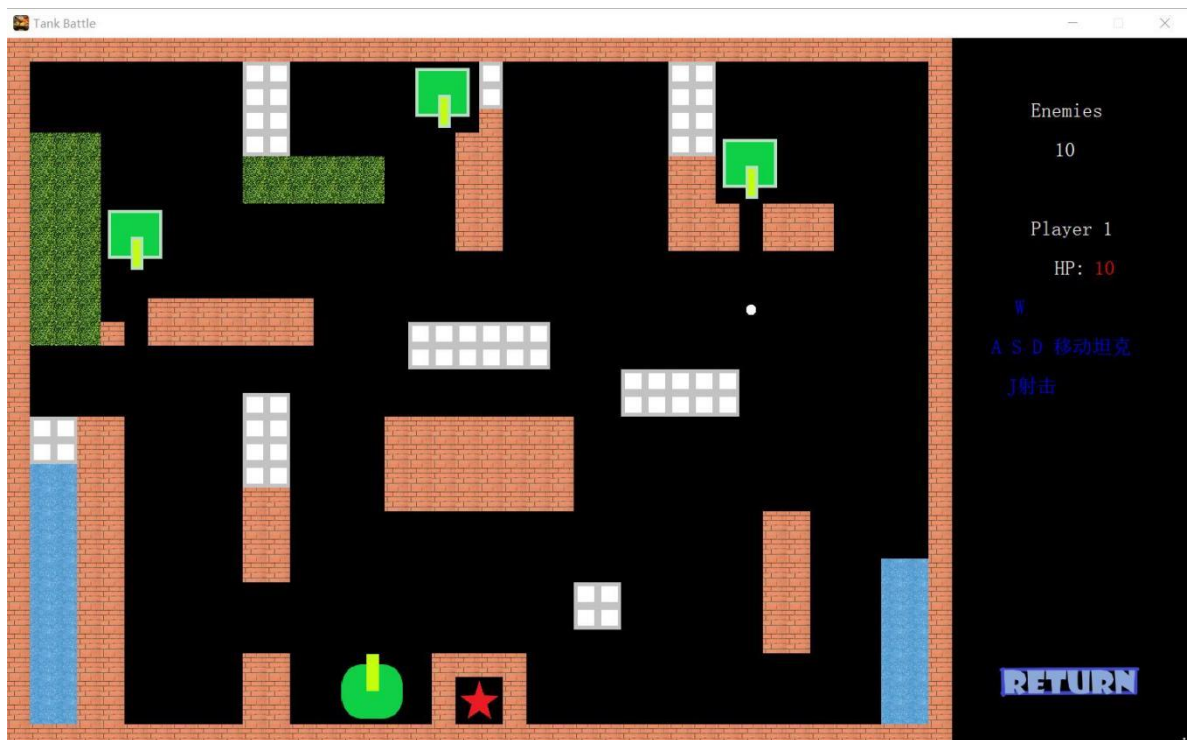
游戏初始界面：



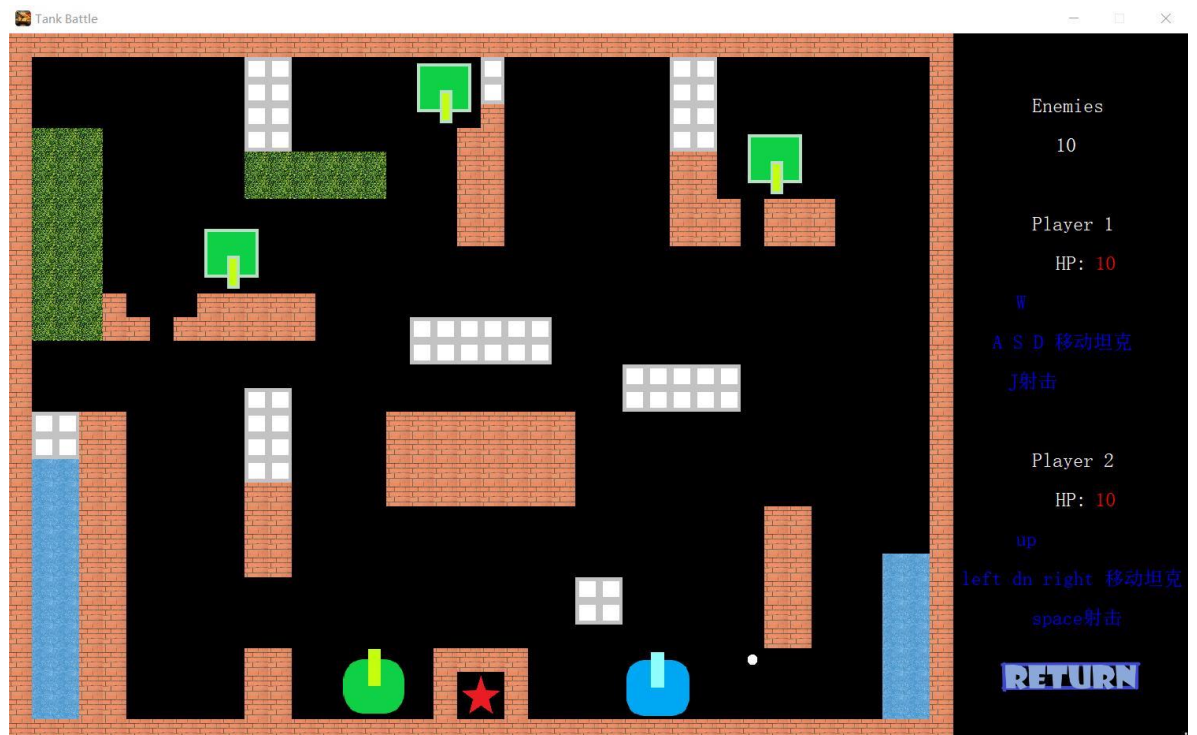
鼠标左键点击"START"键进入模式选择：（或点击“EXIT”键退出程序）



鼠标左键点击"1 PLAYER"键进入单人模式：

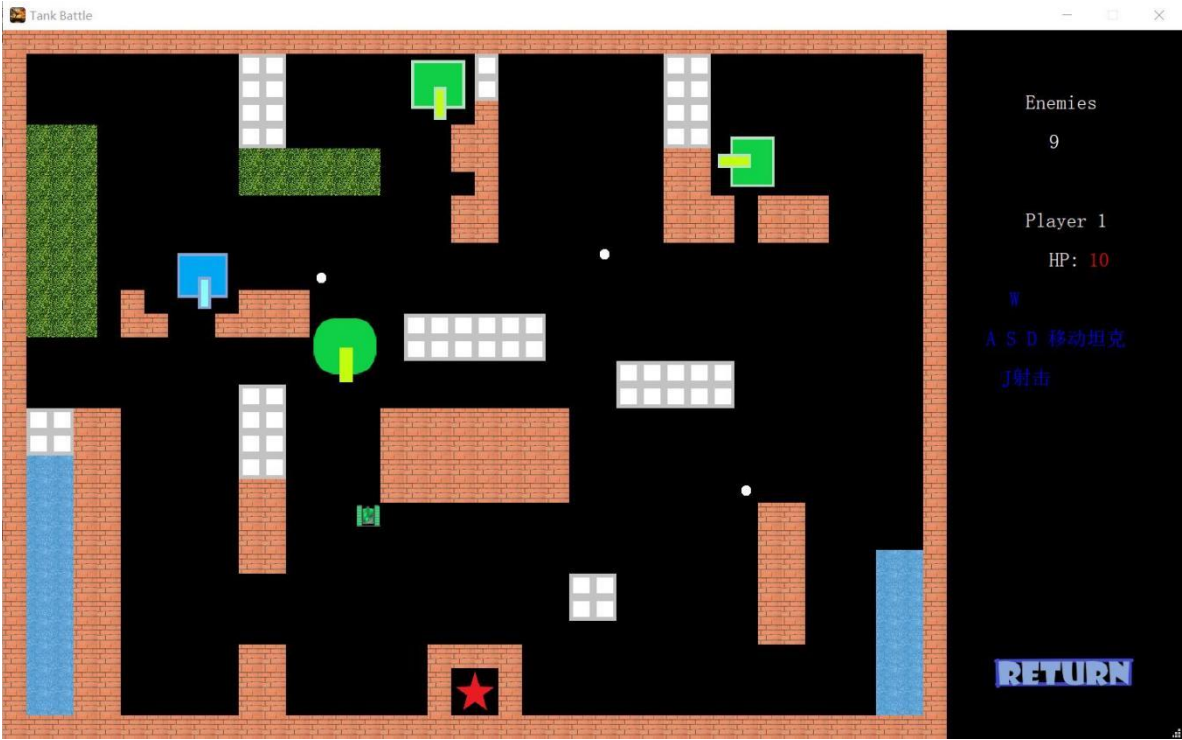


点击"2 PLAYER"键进入双人模式:



注： 点击右下方的“RETURN”键可返回模式选择界面

游戏过程中可掉落游戏道具（以坦克道具为例）：



游戏结束后显示胜负和得分，（胜：WIN! 负：GAME OVER），点击“RETURN”键重新开始。

