我的程序的设计思路是oop的方法。先创建一个sprite的对象（其中有位置，速度，长宽，生命，图像等最基本的属性），然后用类继承的方式创建bullet,player,enemy三个对象，对各自加上其所需要的额外属性与额外函数。这样做的好处是使代码结构更加清晰与更强的可扩展性（直接在类中添加需要的功能）。随后用这些基本对象模拟射击游戏的过程：在每帧中首先用deal计算各对象的状态，然后draw当前画面中的所有对象。

下面逐一阐述基本要求的实现过程：

1. 游戏界面由一个显示画面的窗体呈现。

在示例文件中已提供

1. 提供一个状态栏，实时显示分数等信息。

通过一个drawtext函数实现将文字转成图片并显示在特定位置的任务。调用此函数显示player1，player2的分数，生命，和超级武器的情况

1. 用户控制一架飞机，在画面中可上、下、左、右方向移动（分别由键盘的四个方向键控制，或由 WASD 四个 按键控制）。飞机也可向左上、右上、左下、右下四个斜方向移动，例如，当用户同时按下“↑”键与“←”键时， 飞机向左前方移动。用户控制的飞机不能移动超出画面范围。

用keyevent函数来接受键盘的按键操作，逐渐给予飞机加速度，判断飞机速度是否达到预定速度，若达到，则已预定速度移动。用move来判断是否还在按键，实现放开按键时逐渐减速。用new\_pos函数测试飞机越界，如果越界，就把它拉回来。

1. 敌机持续出现在游戏画面上方，按照某种轨迹向下方移动。敌机会发射子弹攻击用户控制的飞机，且用户控制的飞机与敌机相撞时也会受到损坏。

用count的变量记录游戏进行的时间，每刷新一帧就加1，当count%120==0时向vector<enemy>中添加一个enemy对象，从而实现每2秒出现一架敌机，用随机数提供enemy的位置 pointd（randomnumber，0）。enemy有一个构造函数，这样就能在每次添加敌机时自动赋初值属性，其中就比如有速度PointD（0，2），就能直线向下移动。敌机与我机都有一个vector<fire>，记录发射子弹的位置，敌机每秒向bullets\_enemy中添加一个bullet对象，有初速，初始位置就是fire的pos（也就是敌机的pos），向下移动攻击我机。对相撞损坏功能的介绍留到下一章节介绍。

1. 用户可以通过某按键控制发射无限制的子弹攻击敌机。

首先通过判断space或’q’键是否按下来确定是否发射子弹，只要player i按下，就向bullets\_player[i]中添加一个bullet对象。下面介绍collide和collide2函数，这两个都是检测碰撞的函数，但由于我设定我机和敌机碰撞，我机子弹击中敌机用原始图片尺寸，敌机子弹击中我机用较小的尺寸，所以分了两种collide，collide对应前一种，collide2对应后一种。怎么检测碰撞？就根据两个图片的pos和width，length判断有无重合。

如果我机射的子弹打中对方，就移除这个子弹，对方生命-1（总生命为5）。如果敌机射的子弹击中我方，就移除这个子弹，我机生命-1。如果两机相撞就移除敌机，我方生命-1。当敌方生命到0（即!enemy.alive()）时，就移除该敌机。我方若被击中或碰撞会损毁，图片透明度设置为128，然后会有2秒无敌时间（即碰撞不会造成伤害）。

1. 用户控制的飞机在受到伤害后会损毁，共有若干次复活机会。

上一章节已阐释。复活机会也就是上面说的无敌时间

1. 游戏进行到一定程度后，不再有敌机出现，游戏结束，此时显示用户的游戏相关数据，例如消灭敌机数，己方损毁次数等，并按一定规则为用户的表现打分。

当2个飞机的life都为0或者时间达到1分钟时，就停止游戏（不再deal，draw）,显示game over的字样并显示player1的消灭敌机数，损毁次数，根据公式给分，5秒之后显示player2的上述3样表现。

对选做内容的实现方式：

我选做了3个项目

1. 超级武器（5%）：具有一个使用次数很有限的特殊技能，由一按键控制，可对画面中的所有敌机造成大规模 伤害。

按下‘c’ 或 ‘x’ 可以释放超级武器，我预设每个玩家使用次数为1，还保留了一个cooling\_down 的变量（可以以后增加超级武器时设置冷却时间）。当释放超级武器时，可以清除场上所有敌机与敌方子弹，直接用bullets\_enemy.clear();enemies.clear();

1. 双人模式（10%）：实现双人游戏模式，可供两个玩家各操作一架飞机，协同作战。

这个功能相当简单。我用了vector<player> 来添加我机，然后将对player[0]的操作同样的也应用于player[1] 即可。注意点：分开算分，所有操作都需要分开计算。

\*可能有些键盘会没法同时操作两架飞机

3. 添加音效（10%）：自行研究 SDL2 的使用，在游戏中添加音效，要求音效对于游戏中的撞击、中弹等事件有 响应。

首先要在头文件中添加#include <SDL2/SDL\_mixer.h>,然后在链接选项中还要添加-lSDL2\_mixer，这样就可以用mixer中的函数了。Mixer的使用和image一样，都要首先load音乐/图像，然后再播放/绘画。我对游戏中从头到尾都在放的音乐作为music播放（因为只能同时播放一个music），用Mix\_PlayMusic函数，对游戏中的音效作为chunk，在自己创建的频道中播放。对事件的相应就把播放音乐的函数放在判断事件发生的if语句里即可。最后程序结束时将mix\_music\*，mix\_chunk\*释放即可

除了这些之外的功能实现：

1. 在敌机/我机中弹或两机碰撞时会有爆炸特效

实现方法是创建一个explosion对象，建立一个vector<explosion>，在检测到碰撞或中弹时就添加一个元素，初始位置为敌机当前位置。设置显示时间，在每一帧递减，直到0就移除这个对象。

在大作业完成过程中遇到的困惑以及解决方式

1. 我以前了解到oop是在python中，那个写起来十分方便，然而到了c++中我写到类会有一些语法问题不太明白，比如public，private，构造函数，析构函数等等。我就自己跳着翻书预习这些内容。另外我对于一个项目中.h,.cpp文件的编译不是很清楚，一开始写就有一些编译错误，后来经助教提示我把它全放在了hpp文件中。
2. 第二个困惑是对于sdl2\_image和sdl2\_mixer的使用。我原来是在每次添加对象的时候都loadimage，每次需要播放音效的时候都loadwav，这样会造成重复load，而我当时没有注意，就造成了内存泄漏。当时助教提醒我关注loadimage，我改掉了之后还是出错。后来我不断研究才发现是音效的问题。我意识到image 和mixer是相通的，都是先用load函数给mix\_chunk\*/image\*赋值 ，然后再用play/draw函数（以前面的指针为参数），也就是说只用load一次即可，最后将这些指针所指的清空。
3. 起初我对游戏运行的结构不太清楚，也就是initialize,work,deal 这些函数不太懂，我后来把我以前写过的pygame重新翻出来看了一遍，才知道了一帧之内要干的事。

对于这种稍具规模的程序开发有何新的体会：

写这种程序首先要确定结构，大体上的initialize,work,deal,draw这些函数要干什么都要弄懂。然后大致用oop还是过程要想清楚，（我一开始虽然想做oop，但是助教给的示例文件都是过程就懒得改，后来写的越来越乱才想到改就特别麻烦）。然后我是从示例文件一步步添加功能：射子弹，有敌机，碰撞，生命等等，不能一蹴而就。最后开发这种程序发布前一定要debug（就像我以为前80分做好了之后还发现了两个bug）。