植物大战僵尸课程设计报告二

一、课程设计的主要内容

在第一次课设的基础上，进行了如下拓展：

拓展植物种类：

双发射手：实现一次发射两个豌豆。

寒冰射手：发射冷冻豌豆，击中敌人造成伤害后附加减速效果。

坚果墙：生命值高，阻挡僵尸前进。

高坚果：生命值是坚果墙的两倍，并且能阻止撑杆僵尸越过。

窝瓜：但有僵尸出现在其攻击范围后，炸毁该地块的所有僵尸。并且在跳跃过程中处于无敌状态。

樱桃炸弹：炸掉周围3*3地块的所有僵尸。

大蒜：啃食过大蒜的僵尸会被驱赶到临近行。

南瓜头：种在其他植物上，以免被僵尸吃掉。

拓展僵尸种类：

路障僵尸：有较高的防御能力，只有在路障被打掉后本体才会收到伤害。

读报僵尸：报纸被打掉后，移动速度将会加快。报纸可以抵挡冷冻豌豆，只有当报纸打掉后才会收到减速效果影响。

撑杆僵尸：具有较高的移动速度，可以跳过遇到的第一个植物。

小丑僵尸：以一定的概率自爆，并炸毁周围3*3地块上的植物。在前面两个自爆概率为0，当走进第三格后，随着格数增加，自爆的概率也会增加，在到达最后一格的时候自爆概率为100，也就是说小丑僵尸无法进入屋内。

投石僵尸：用篮球远程攻击植物，待篮球投完后会缓慢开车碾压植物。先会开车碾压前三格的植物，到达第三格后，如果该行有植物，则会投射篮球攻击植物，待改行没有植物或者所有篮球投射完后，会继续开车碾压植物。

拓展商店：

实现植物的购买具有冷却时间。

可以用键盘在商店界面进行上下左右的选择。

拓展界面：

实现5*7的地块。

更加每个地块的状态以不同的方式显示地块的内容，包括地块中的植物名称及血量，僵尸名称及血量，僵尸数目。

每个地块中可以有多个僵尸，并且地块中的僵尸具有一定的位置先后关系，排在最前的僵尸最先受到伤害。

二、课程设计的思路

1.和第一次课设一样，以状态机的方式实时对用户的输入做出反应，达到多线程并行，而不会阻碍其他功能的执行。

2.随着植物和僵尸种类的增加，考虑用继承实现代码复用，所有植物公用一个基类，所有僵尸以普通僵尸为基类，利用虚函数重写每种植物和僵尸的特殊的功能。

3.利用STL中的容器储存每个地块中的植物、僵尸以及子弹，方便遍历，移除和添加。

4.游戏主循环采用：绘制界面（draw），实时获得用户输入（getkey），更新每次循环后各个对象的状态和行为（update），以一定的逻辑执行相应的功能。

三、主要类的设计

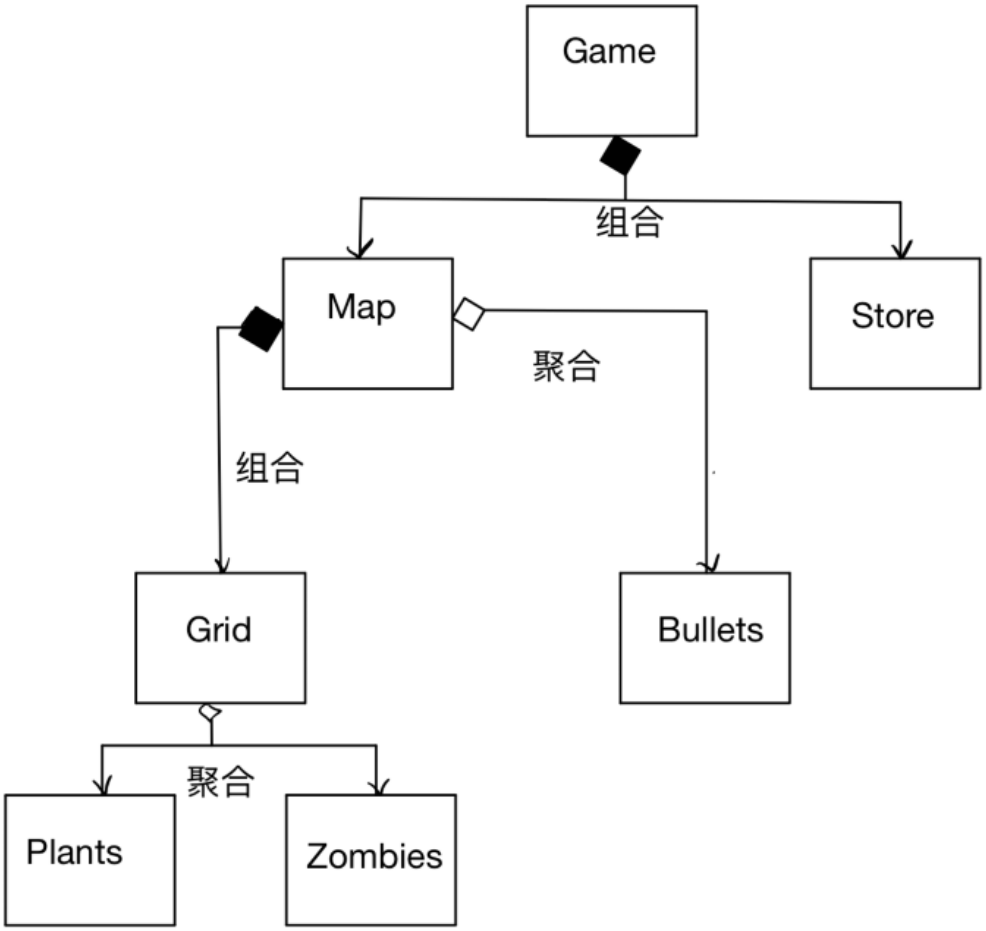
该项目包含Game类，Map类，Store类，Grid类，Plants类及其派生类，Zombies类及其派生类，Bullets类及其派生类。

1.类之间的关系

Game类与Store类和Map类是组合关系，Game类中含有一个Store成员对象和Map成员对象，三者同时产生同时消亡。

Map与Grid为组合关系，Map中有一个Grid类的二维数组，随着Map产生和消亡；Map类与Bullets为聚合关系，Map中有Bullets*类型的deque容器，用于储存临时产生的Bullets对象。

Grid与Plants和Zombies都是聚合关系，Grid用分别有储存Plants*的vector容器和Zombies*的deque容器。

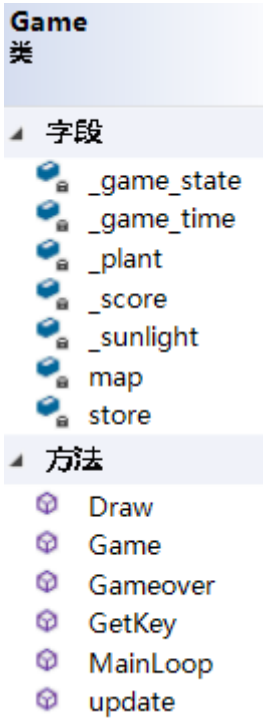


Plants为基类，分别派生出Peashooter、Sunflower、RepeatPeaShooter、SnowPeaShooter、WallNut、TallNut、Squash、CherryBomb、Garlic、Pumpkin类。

Zombies为基类，分别派生出ConheadZombie、PoleVaultingZombie、NewspaperZombie、JackInTheBoxZombie、CatapultZombie类。

2.类的数据与操作

Game



数据：

map和store分别是两个成员对象，game通过不断向map和store发送消息完成游戏的运行。

_plant是一个结构体PlantCard的数据成员，负责从store中将玩家购买产生的临时植物对象传到map中对于grid中并种植。

_score则是当前游戏的分数，_sunlight则是当前游戏的阳光数。

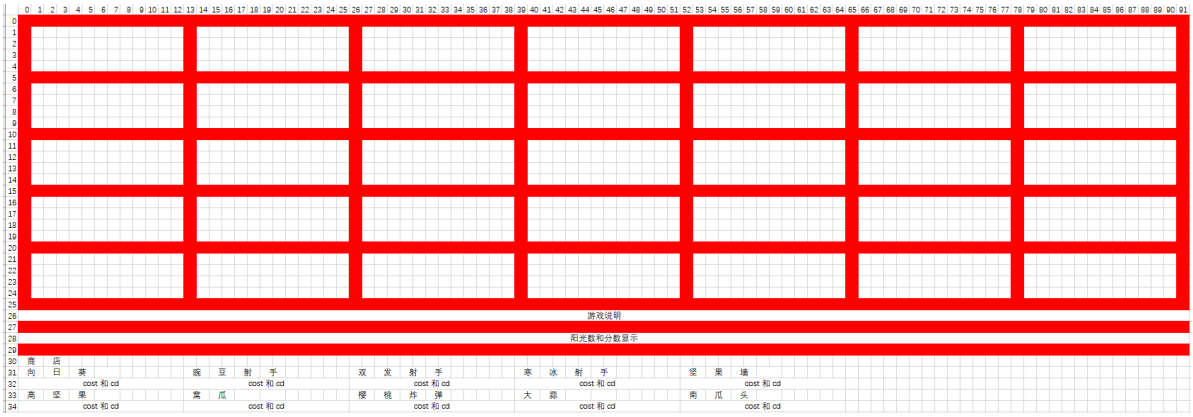
_game_state是一个自定义的枚举类型GameState数据成员，记录当前的游戏状态。

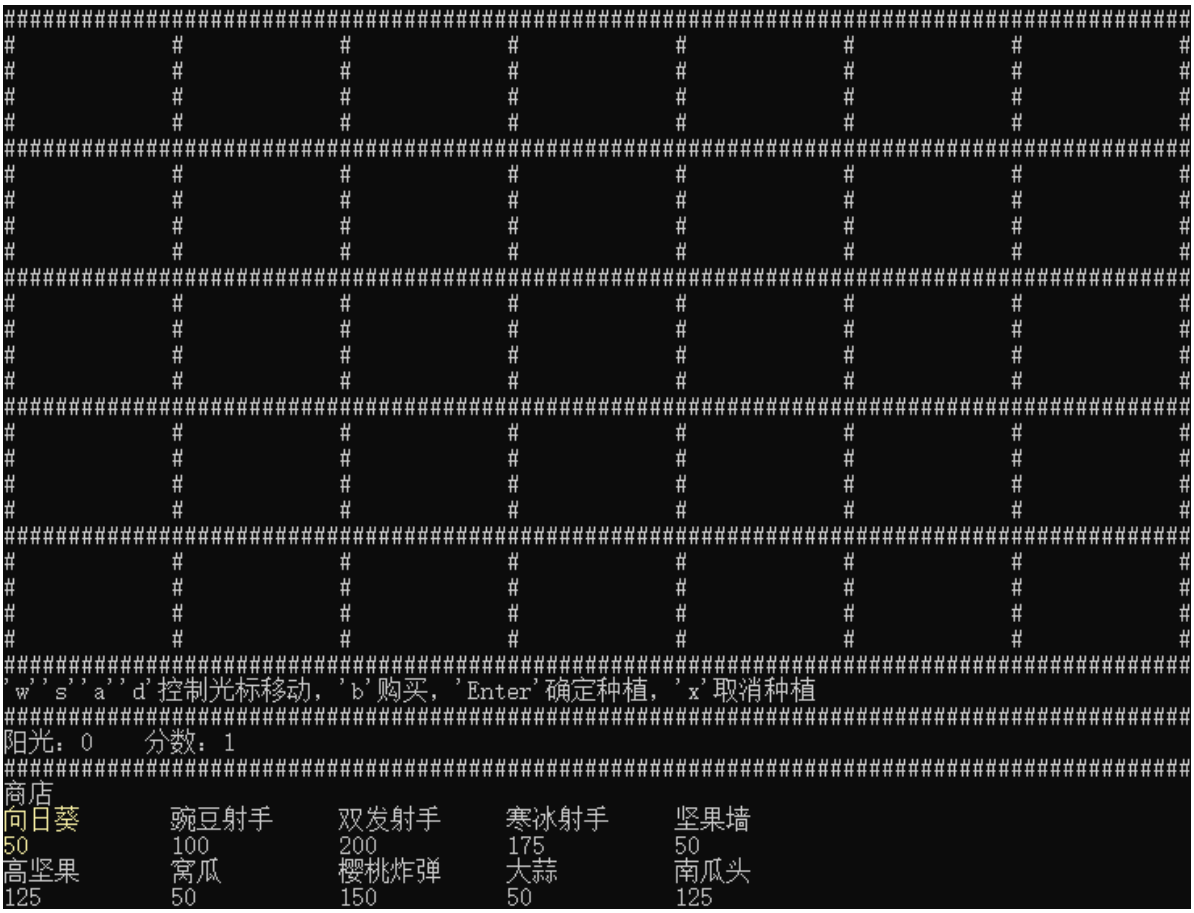
_game_time则是一个clock_t型的数据成员，用于记录游戏已经运行的时间，用于根据玩家的游戏时间进行不同的投放僵尸的方式，以不断增大游戏难度。

操作：

Draw函数会分别调用map和store的绘图函数，实现在命令行上输出游戏界面。

游戏界面设置如下：





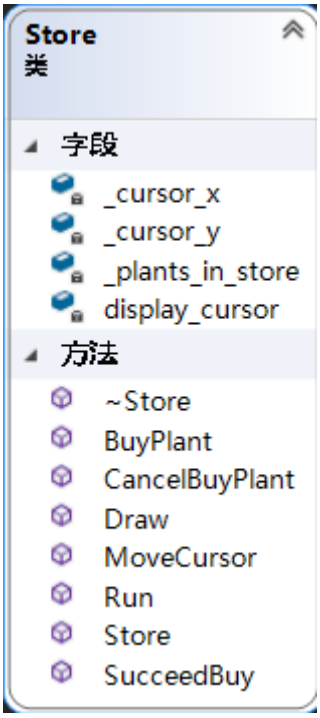
GameOver函数用于在_game_state变为GAMEOVER时执行跳出游戏主循环，打印最终游戏结果。

GetKey函数用于获得用户输入，并且采用状态机的方式根据不同的游戏状态进行相应的反应。

MainLoop函数是游戏的主循环。

update函数里面负责更新各个对象的状态，执行相应的功能，例如move，attack，function等等

Store



数据：

(_cursor_x, _cursor_y)代表当前商店中光标的位置；

_plants_in_store是一个PlantCard类型的数组，里面存放在这商店里植物的信息，包括CD，cost等等；

注：程序里所有count结尾的数据成员都是clock_t类型的数据，用于计时，实现每隔一定的时间执行特定的功能。

操作：

BuyPlant函数是购买植物函数，如果用户的阳光数足够，则会更加光标所在地生产对于植物的动态对象，然后传递到game中的_plant里；

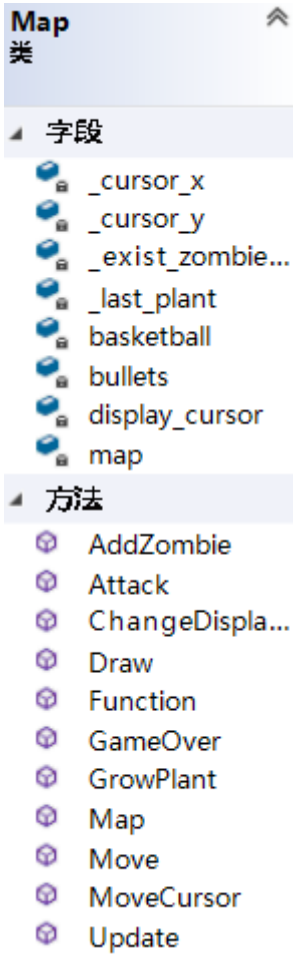
CancelBuyPlant函数是购买取消函数，如果用户取消购买，则会调用此函数销毁之前创建的临时对象。

SucceedBuy函数是购买成功函数时会调用的函数，用于扣除对应的阳光数。

Draw函数是绘制商店函数，如果植物卡片处于冷却状态，会显示已冷却的时间。

Run函数是每次循环都会调用的函数，表示商店的运行，在此函数中实现了商店植物的冷却刷新和定时产生自然光。

Map



数据：

(_cursor_x, _cursor_y)代表商店里光标的坐标；

_exist_zombie_lines是一个bool型的数组，用于标记草坪每行是否有僵尸，用于豌豆射手是否需要发射豌豆的判断。

_last_plant是一个int型的数组，用于标记草坪每行最后面一个植物的位置，-1代表该行没有植物，用于判断投石僵尸是否需要投石的判断。

basketball和bullets分别是deque容器，用于投石僵尸投射的篮球和储存植物发出的豌豆子弹。

display_cursor用于标记当前游戏是否需要显示草坪中的光标。

map则是一个Grid类的数组，代表草坪的5*7的格子。

操作：

Draw函数，遍历map，绘制草坪地图；

MoveCursor函数用于根据用户的输入移动草坪的光标；

GrowPlant函数用于判断是否可以种植植物，如果可以种植则将植物种植到光标对应的Grid中。

AddZombie函数实现在地图里产生僵尸。

Move函数实现僵尸和子弹的移动。

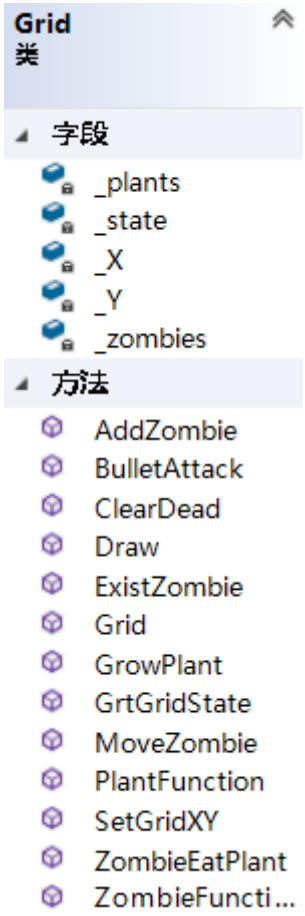
update函数用于更新_exist_zombie_lines和_last_plant和清除地图中HP归为0的僵尸和植物，减少bug的产生。

GameOver函数用于判断游戏是否接受；

Attack函数实现僵尸攻击植物和子弹打中植物或者僵尸

Funtion函数实现让僵尸和植物执行各自特有的功能。

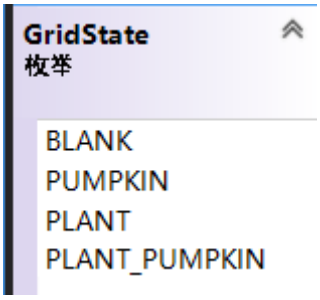
Grid



数据：

(_X, _Y)代表格子左上角在命令行中的坐标，用于draw时的定位；

_state代表格子的状态，类型为GridState类型的枚举型，用于实现植物南瓜头可以种植在其他植物上 的功能



_plant和_zombie则是储存格子中植物和僵尸的容器；

操作：

AddZombie函数实现在该格子中增加僵尸

BulletAttack函数则会实现子弹打中目标，并且如果目标死亡，会消亡目标并移除容器，目标可以是植物或者僵尸

ClearDead函数用于按照需求清除格子里的僵尸、植物，也int型的参数表示需求；

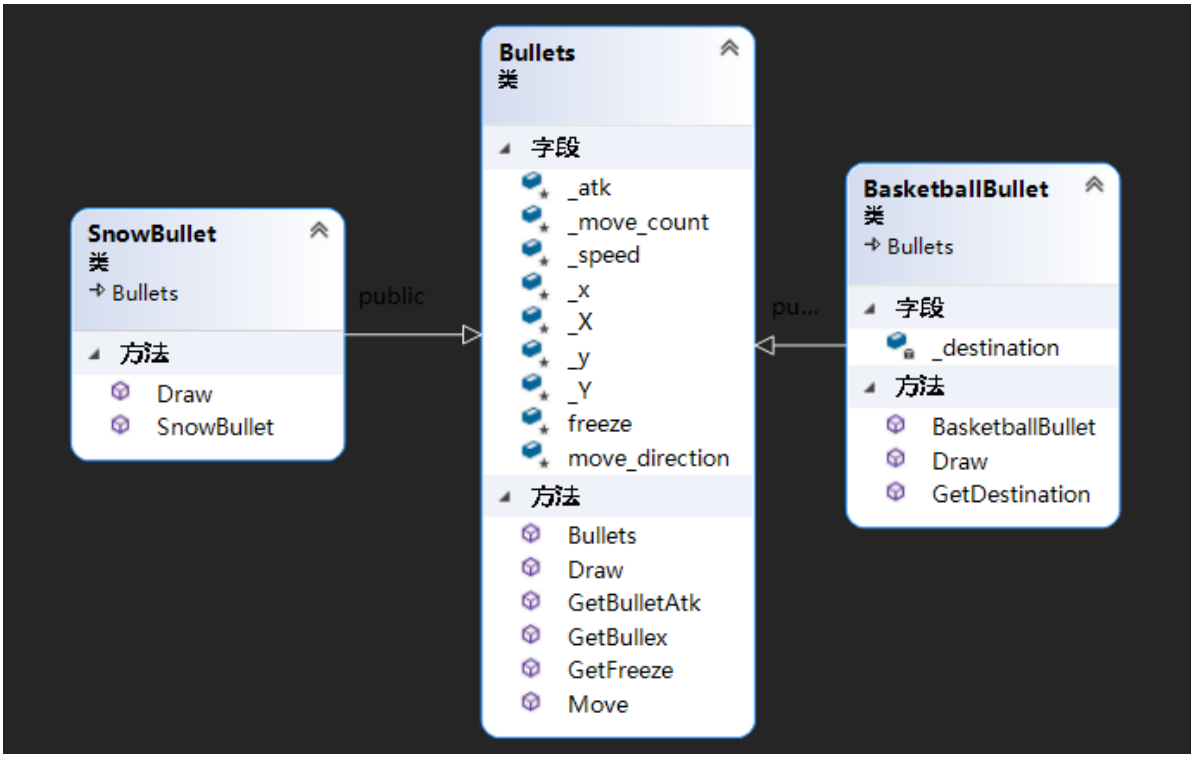
GrowPlant函数用于种植植物

MoveZombie函数遍历格子中的僵尸执行其Move函数，如果可以移动则对僵尸进行移动，包括向前移动和啃食大蒜后的换路操作

ZombieEatPlant实现僵尸啃食植物的操作；

ZombieFunction和PlantFunction用于遍历格子中的僵尸和植物调用他们的Function函数执行相应的功能，Grid会对Function返回的枚举型变量分别执行：该处产生的自爆、该处被窝瓜压瘪、该处发射了子弹等操作。

Bullets



Bullets本身表示普通的豌豆子弹，通过继承实现冷冻子弹和篮球子弹。

每种子弹的伤害均为20每颗，移动速度为0.1秒/字节。

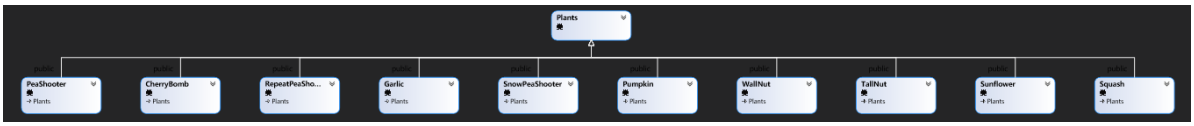
freeze表示子弹是否具有冷冻效果，当僵尸被具有冷冻效果的子弹打中后，会开启减速效果；

篮球子弹中新定义的_destination表示篮球子弹目标植物所在坐标。

操作

Draw函数会根据不同子弹以不同颜色绘制子弹，绿色为普通豌豆子弹，蓝色为冷冻豌豆子弹，红色为篮球子弹。

Plants



通过从Plants基类实现十种植物

step_over表示植物能否被翻越，高坚果是false，其余植物均为true；

invincibility表示植物是否处于无敌状态，处于此状态的植物不会扣血，主要是用于实现窝瓜锁定目标跳起后的无敌时间；

Function函数用于执行植物特定的功能，对于有需要的植物可以重写此函数

Sunflower类

重写Function函数实现定时增加游戏阳光数

PeaShooter

重写Function函数实现发射单发豌豆

class RepeatPeaShooter :public Plants

重写Function函数实现连续发射两枚豌豆

SnowPeaShooter

重写Function函数实现发射冷冻豌豆

Squash

重写Function函数实现压瘪僵尸，锁定顺序依次为：后一格，当前格，前一格

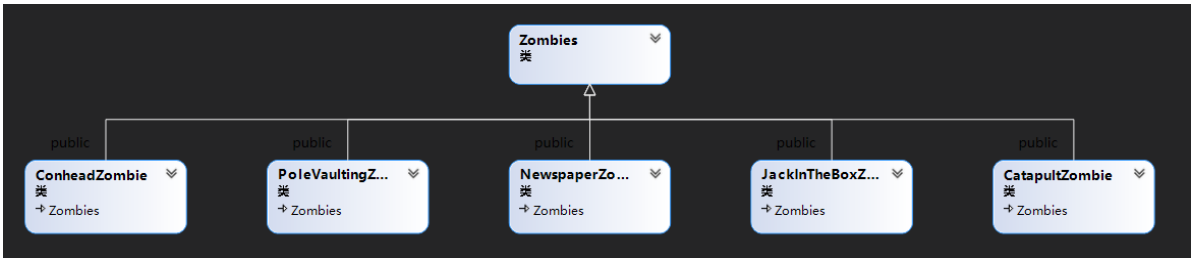
CherryBomb

重写Function函数实现自爆，炸掉附件3*3格内的僵尸

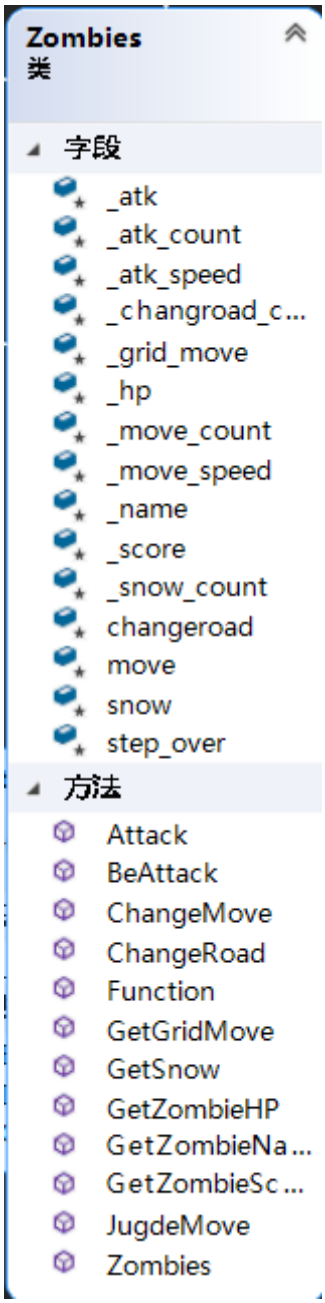
Garlic

重写BeAte函数，实现僵尸啃咬植物后进入换路状态

Zombies



Zombies类代表普通僵尸，其余5种僵尸以此为基类继承得到



_grid_move代表僵尸在格子中移动的距离，当子弹判定时，会最先判断打到最前面的僵尸，即_grid_move值最大的僵尸；

move代表僵尸是否可以移动；

changroad代表僵尸是否需要换路，及中了大蒜的效果；

snow代表僵尸是否处于减速状态，减速状态的移速和攻击速度都会下降；

step_over表示僵尸是否可以翻越植物，只有撑杆僵尸为true；

ConheadZombie

重写Function函数，实现只有路障打掉后本体才会受到伤害；

PoleVaultingZombie

重写JugdeMove函数实现遇到的第一个植物，翻越后速度下降；

NewspaperZombie

重写BeAttack函数，实现报纸可以抵挡冷冻豌豆的减速效果，实现报纸打掉后的狂暴状态；

JackInTheBoxZombie

从写Function函数，实现离庭院越近爆炸几率越大。

CatapultZombie

重写ChangRoad函数，实现投石僵尸不受大蒜影响

重写Function函数，实现投石僵尸发射篮球；