潘峰立的贪吃蛇介绍：

（这个是第三次上传，和第二次相比修改了一点小bug——我的有一个overpass判断的参数名字搞混了所以可能出现特殊食物和bricks在一块的情况）

需要声明的是，我这个蛇的界面是在那个钟的基础上重写，利用实例中的画图函数纯画出来的蛇，不像其他有些同学一样在ui界面上写（私以为那样比较麻烦）但是我还是设计了一个ui的开始界面，让它更像是一个游戏  
 通过点击欢迎界面的singlemode可以进入单人模式  
 通过window.cpp里面的槽函数让这里的已经实例化的SnakeWindow窗口clock的show函数展示出来（这里的clock是因为在时钟的例子上修改的缘故，保留下来以做纪念）。请看到window.cpp文件，在show之前我还有一个  
restart函数（具体的在test.h里面SnakeWindow类里面实现）。这个函数是为了让我们在关闭窗口后重新打开能够重新开始一个新的游戏——函数内容大概就是把所有的变量数据初始化。——因为我没有能够找到自带的重开函数所以才自己写了这个。

贪吃蛇的具体的实现请看test.h文件  
 （这里的文件名有点抽象，因为这本来只是我的一个副本，然而父本却莫名其妙地崩了，所以这个就成了最后的正品）  
 这里设计了这么几个类:  
 Point:因为qt自带的QPoint不好用 就重写了一遍  
 Snake:蛇的本体，实际上是一个Point数组，此外还有他的生存状态（用于之后判断游戏结束）。此外还包含了所需要的函数比如生长，移动，等等。Resetposition是用于多人游戏里面多条蛇的定制初始化，reset就是迎合上面所讲的restart函数把蛇里面的数据初始化。

Food：就是食物类，包含了颜色还有坐标——由于设定上同时只有一个food所以就只有一个点的坐标，它也有存在状态，用于判断有没有吃到进而产生新的food。函数没什么好说的，其中changeposition函数是随机重新生成新的food。

Bricks：就是要求的随机生成砖块，类似于蛇，本体也是一个Point数组，也和food有点像，x，y表示的是新生成的砖块的坐标——因为已经生成的砖块在我这个项目里面不用做任何处理。同样地也有changeposition和reset函数。这里多了一个addBricks，就和蛇的grow差不多。

Queue：就是一个队列。因为蛇吃到food之后不是直接变长，而是尾巴到了那里之后再长长，所以需要一个Queue来存储吃到的food的位置，进而判断什么时候应该长长。

SnakeWindow：窗口界面，包含了从时钟实例继承的timeEvent和render等等函数。成员变量和函数大都比较明显易懂。其中spefood是特殊食物，吃到了会随机加速或者减速（有上下限）而且分数也会随机地相应变化。

Score之后的需要解释。本来这些都是我在cpp里面定义的变量，但是为了restart就必须把他们放到这里。Score就是得分——会在游戏结束时结算。i和N有注释，在后面也会介绍。那几个bool都是判断有没有交集——这个会在重新生成砖块以及食物的时候有用。Blink是为了最后展现一个死亡特效而引入的“计时器”。n和pre都是为了记录死亡时候的状态。

然后进入test.cpp  
 keyPressEvent就是对按键的反应，也就是上下左右控制方向。此外，按空格（Space）还可以暂停不动，之后按方向可以继续动。

SetInterval是为了后来加速减速，调节刷新时间以实现。

TimeEvent里面是蛇根据我们的direction而动。Predirection是给direction留一个副本，因为蛇不能够和自己的方向反着走。

主要是在render里面。大都有注释，比较容易读。这里的food是三种不一样的普通food按着顺序出现，各自的分数不一样，而spefood都是一样的。

需要特别说明的就是死亡之后的操作。首先把direction设置成一个无关的字符，让蛇不动。此外N记录数加1启动下面的操作——这里++是为了只进行一次下面的操作，不然会一直弹弹弹出来。之后，Qlabel就是最后的计分板。MessageBox就是一个简单弹窗，告知游戏结束。之后的blink就是为了营造一闪一闪的效果让blink=（blink+1）%2，在1、2之间循环，就可以一次画，一次不画，就闪起来了。

其实后来的MultiMode没什么好说的，只不过是在单人模式上整两个蛇（多条蛇也是一样的道理），多一些判断（比如两个蛇相撞）罢了。有部分功能只是在单人模式里面实现 因为没必要复制粘贴那么多到另一个文件里面

ai蛇真的有点笨。

这里用begin参数设置了一个按空格开始的操作，避免相互干扰。（其实后来debug之后就没有干扰了？）然后和单人模式一样关闭之后会restart一下。

它会根据离蛇头的距离来判断目标以及运动方向。当然是最短路径 但是总想着让蛇聪明一点。就加了众多的死亡判定而对于自己吃到自己的情况不能够很好地防止。对于有些同学说的广搜，应该可以操作，但是我这里已经是通过direction来判断运动了，所以对于即成路径的适应性可能不是那么好。  
 没有在ai蛇里面设置砖块，那样会增大难度，也就还将就吧。  
 不管怎样改都还是会出现提前死亡的情况  
 还是讲一下具体代码吧（虽然很笨）

一开始的初始化加上了初始方向direction=‘r’。一开始会向右。然后为什么它一开始会向下而不是向上呢？因为我的food的初始化是设置了一个9999的绝对大数，避免一开始就会相交，所以会先按着9999判断。这个其实无伤大雅。

在timerEvent里面。先确定最近的食物，存入XY。这里我把空间根据蛇头的位置和运动方向分成了四份，根据XY和蛇头的位置来进行“推荐方向”的判断——也就是不管到底会不会死亡的最短路径，在此基础上还加上了死亡判定要是会死就换个方向或者保持原方向。需要提醒的一点就是参数C，这个用于判断是不是已经在一次刷新的时候改过了方向，不然地话，之前改好的方向会在之后又被改一边，甚至会出现反向进入自己的情况。

最后稍微提一下特殊功能吧。就是那个Setting，点进去可以选择bgm，有三个，点一次开始放，再点一次可以停止。——需要提醒的是，Qt中的多媒体播放，底层是使用DirectShowPlayerService，所以需要安装一个DirectShow解码器，例如LAV，进而可以正常播放。

其实这个是我在设计过程中突发奇想整的一个，只是为了让游戏更像游戏一点。当时还没有意识到自己应该设计一个特殊功能，因为觉得蛇已经比较完善了，再强行进行创新会显得有点老套，就是画“蛇”添足的味道，所以在此外进行添加。其实基本要求真的蛮少的，所以上面我自己加的什么计分板啊以及暂停之类的功能也可以算在拓展功能范畴之内。