贪吃蛇设计报告

高程大作业

设计思路与功能描述

这次写的贪吃蛇主要分为 4 个类：Wall、Food、Snake、Menu，下面依次介绍

Wall.h：

1.地图方面：将整个地图分为 80 列，60 行，然后用数组 Map[150][150]去记录下每个位置上储存的字符信息，*对应蛇，#对应墙，$对应食物，!对应额外增加的炸弹，&对应额外增加的回血奖励，实现主要是依靠 void putCharInMap (int x, int y, char c)函数，它会在 Map[x][y]写入 c，以实现记录。然后还有一个很重要的功能，就是到显示屏上的特定位置进行输出字符串，依靠的是对光标的操作，主要依靠函数 void goMapXY(int x,int y)，但是要注意这一函数操作的仅仅是对菜单界面的也就是一开始的界面，到了游戏界面后，因为是用了 easyX 进行了优化画面，所以在那里不再起效，依靠的是 outtextxy（）easyX 自带的输出字符串的函数。然后在地图方面注意的是，在菜单界面的窗口，x,y 就是 x,y，但是在游戏界面，我是将其分为了以 10*10 为单位的地图，也就是说，在 Map 中 x,y 的位置，对应的游戏界面是 10*x,10*y 到 10+10*x,10+10*y 的画面。

2.墙壁方面：将墙壁对应的位置录入进 Map 数组中去，然后在记录完了之后，还有就是绘制，这里因为依靠的是 easyX 的绘制功能，所以我就直接就在四周画上了 4 个矩形，为了契合软墙和硬墙的要求，最上面和最下面那 2 条，用了不同的颜色。还有一个是增加墙壁的功能，就是 void addWall(int x,int y)函数，将额外变为墙壁的地方记录进去，主要是为了对应蛇变成墙壁这一功能要求，这样记录主要是因为在绘制的时候不是选择遍历的方式的输出图片，而是定点，感觉这样会高效一些，这样就需要定点知道有没有增加的墙壁，增加的墙壁在哪了。

3.提示方面：这个类还承包了一个输出提示功能，前面的都是惯例输出，但是在游戏规则那一块需要依据不同的游戏选择来输出与之对应的信息。然后是用 outtextxy（）进行输出文字，注意到要把数字转换为字符串再写上去。然后里面的记录时间的功能也是要求了它需要记录游戏开始的时间，然后其他的 主要靠参数传入。

Food.h

Food 是在 Wall 的基础上产生的，主要用来管理食物，里面有数组用来记录食物的位置、总量、该点的食物是否被吃掉了，现在剩余的食物量，关键的随机产生食物要先考虑到是否还有位置可以产生食物，我采取的办法就是提前遍历 Map 数组，如果有 1 个位置是空的，那就说明是有空间可以产生食物的，如果检测到有 5 个位置是空的，那就说明是可以像平常一样，随机产生 1~5 个食物的，然后继续，如果遍历下来一个空的位置都没有，说明空间已经满了，就返回失败，然后游戏结束，如果有大于 0 但是小于 5，说明也至少还是有位置的，所以只用在这个食物数量范围内随机产生食物就好了，产生食物嘛，就是先刷机随机种子，然后随机出一个 x 一个 y，再检测 Map 数组中[x][y]的位置是不是空的，如果是空的，就放入$的标记，标记这里是食物了，并且把它放入储存食物位置的数组中，如果不是，就继续随机出下一个。还有一些函数，drawFood()，用来展示食物的图片在界面上，不过要注意没有被吃掉的才需要展示出来，void addFood(int x, int y)是在 x,y 位置增加食物，用额外的函数主要是因为要让食物的总数也要随着发生变化。clearFood（）函数主要是用来清空原本的食物，对应的是一些游戏模式里面，蛇死后需要刷新食物，所以需要先去掉之前的食物标记，把记录食物的数量值归零，把原本的食物在 Map 数组中改为空，并且标记为这些都是被吃掉过了的。还有一个 eatFood(int x,int y)，主要是标记 x,y 的食物被吃掉了，这主要是为了清

楚哪些食物已经被吃掉了，这样在清空食物的时候不至于误伤，因为可能那个已经被吃掉的食物的位置现在有蛇身在那里，会把它改为空，对应的 checkIsEat(COORD p)也可以对后面的 AI 帮其认识到之前锁定的食物目标现在是不是已经被吃掉了。getAfood()也是为 AI 服务的，主要是随机获得一个食物的位置。

Snake.h

私有数据中主要是蛇的长度，用于链表的蛇的结构体，蛇头，蛇的生命值，是否死亡，得分值，上一次移动的方向，停顿时间修改，以及一个子函数 move（），后面介绍。蛇的数据主要是依靠链表来记录的，记录的是每一条蛇身的 x,y 值，这样方便修改。

随机产生蛇和食物一样，也要先遍历一下 Map 数组，看看有没有足够的空间可以产生蛇，因为我产生的蛇都是竖直的 4 格，所以就遍历每一列，看看它这里有没有连续的 4 个空格可以用来产生蛇，如果有，就将它记录入数组之中，遍历完所有列，如果一列都没有，就说明不能产生蛇了，就返回失败，如果可以，就继续，因为刚刚记录了那一列是可以产生蛇的，所以就在这中间随机出蛇头的 x，然后随机出 y，但是 y 注意要和最底下有 3 格的空格，不然蛇身就进墙里面去了，随机出蛇头坐标之后就创建列表，蛇头往下 3 格是蛇身，把这些数据都记录进去链表里。然后是 drawSnake()，就是遍历蛇的链表，在对应的位置进行图片展示。

蛇的移动 moveSnake(char dir)，注意到按不同的方向键对应的只是蛇头的位置改变不同，所以就提出了一个 void move(int dx, int dy)函数，然后不同的方向对应不同的 dx，dy 传入，然后计算得到新的蛇头数据，同时通过循环得到蛇尾的位置，如果这时候蛇头的位置上对应 Map 数组是空的，或者特殊的移动到了蛇尾的位置，就说明移动很正常，就把最后的蛇尾在 Map 数组中设为空，并且删除掉最后一个节点的数据，释放空间。如果吃到了食物，就先标记这里的食物被吃掉了，然后最后的蛇尾就不用变空也不用删除了。如果吃到了奖励（灭火器），就增加长度，生命，分数，并记录不再显示奖励了，如果吃到了炸弹，就死亡，分数-3，分数归零，记录不再显示奖励了，如果不是第四版如果撞到墙了就生命-1 死亡，并且不进行后面的刷新新的蛇身的 Map 信息了，直接返回，如果是第四版，因为设计了软墙和硬墙，所以这时候需要来判断它撞的是哪一个墙，如果是下面的硬墙，就总计生命-2，如果是上面的软墙，就减少 3 个蛇身，如果这时候每有蛇身了，就生命-1，返回，如果还有蛇身，就遍历链表到蛇的第四节，并将前面的蛇蛇在 Map 中记录为空白，然后在 Map 中记录剩下的蛇身位置，然后返回。最后需要将蛇身的位置在 Map 中刷新记录。

停顿时间，根据不同的游戏等级来停顿 不同的时间，因为有游戏四中的加速减速区，所以，有一个 chang，但是要注意到不能让停顿时间小于 0.

将蛇变为墙、食物、空，主要就是遍历蛇的链表，然后将其在 Map 上的字符转化为对应的字符填充，注意到一般都是从头部的下一个开始。

尝试保存游戏信息，在玩家输入完了姓名之后，对于历史最高分记录需要先进文件中的数据输入到数组中去，然后在对应游戏模式进行比较，如果分数比原本的记录高，就改变对应数组的数据，否则不改变，然后再回到文件开头，将所有数据输入到文件中去。对于历史记录，不用输入，因为不用比较，直接在文件的末尾加入相应的数据——游戏模式、玩家姓名、分数。但是在这些操作中都需要注意的是就是在输入的时候要注意，在每次输入数据之间要输入空格，以此方便后面的数据输入。

Menu.h

绘制界面 drawMenu(), 主要是依靠 wall.goMapXY()来绘制, 将光标移动到指定的位置，然后 cout 字符串，然后 Menu::drawChose(int n)，就是跑到指定位置行的左右两侧，来输出 > <这个标识，表示选择，但是需要注意的是就是要在上下对应的 2 侧输出空格来掩盖掉原本的> <标识，然后运用的是上下键来控制选择，依据网上的资料，用 2 个输入键

来接受上下键的信息，按回车键表示进行了选择，第一个是开始游戏，进入游戏界面，后面讲。

第二个是选择难度，跳入另一个界面，依靠的也主要是通过清空界面，然后重新绘制相关的游戏模式信息文字，然后在界面中选择，如果按在某一游戏界面按回车，就表示选择了这一游戏模式，同时在一边显示出选择了什么游戏模式，然后下面有一个返回菜单选择，选择后就结束这一函数，然后继续回到菜单界面。

第三个是显示最高记录，就是回车选择后，读入历史最高记录的文件，然后将他们输出到屏幕上去，然后按任意键返回。

第四个是显示历史记录，也是读入历史文件，但是为了实现改查的功能，所以将文件内容读入到一个链表中去，然后在展示过程中，因为将最近一次的游戏记录在文件的末尾，并且注意到界面最多只能展示 18 条记录，所以在展示的时候，要倒着展示，将第到数 18 个记录展示在界面最下面，然后依次向上推进，然后就能使得最上面的是最近一次的即文件最末尾的记录，然后用>　符号来标识选到了哪一条，然后删除记录的操作实现依靠的是检测到在选择某一条后如果按下了删除键，这时候就出来提示是否要删除，如果是，就将这一条在链表中删除掉，否则不操作，然后上面还有一个查询操作，如果选择后就刷新界面，然后等待输入要查询的玩家姓名，输入完毕后，就在链表中遍历查询，如果可以查到就输出这一信息，然后继续询问是否需要该名字，是之后，就等待输入新的玩家姓名，然后结束，回到历史记录界面，如果上面有一步拒绝了，就直接回到历史记录界面，然后上面还有一个选择键，选择之后就退出函数，回到了菜单界面。

第五个是直接退出游戏，即直接在主函数中返回，退出程序。

Test.cpp：

接下来介绍在主 cpp 中主要的设计逻辑，以及各种游戏的实现。

Main 函数一开始有一个打开文件的操作，其实应该说是建立文件，因为如果是第一次完游戏的话，是没有后面所需要的文件的，所以就干脆先提前建立了一个。然后主要是调用 menu 的功能，其他功能不再赘述，这里主要是游戏功能，在选择了开始游戏之后，进入到一个 play 函数，然后依据选择的游戏模式，进入到不同的游戏中，下面介绍不同游戏的实现。

easyPlay:

最基础的模式，一开始就是靠 loadPhoto（）加载背景图进 img 变量中，然后将绘制对象改为 img，然后将蛇，食物，墙，菜单初始化，然后进入循环，这里循环的条件是蛇不能死掉，进入循环，第一个是进行展示页面的初始化，因为采用的是图片版，而且带有背景，所以这里介绍一下 void showAll(Snake& s,　Wall& w, Food& f, Menu& m)的功能实现，首先是调用 void rePhoto(int n)，依据不同的游戏模式来选择不同的背景图放到 img 的图片内容中去，注意到游戏 4 中因为减速加速区的设置，需要在界面的一定部分加上点，然后将它贴到原本的图片上层，这样的一个最大用处是将原本界面上过时的画面全部遮盖掉，然后就是画蛇、墙、提示、食物、奖励、炸弹到图片上去，然后是将图片再次展现到界面上去，这样就实现了界面的刷新（不过好像还有有点慢），然后结束展示之后是等待案件 awsd 的输入，当然需要判断是这 4 个字符，才将其录入到方向键中，不然还是原本的方向，然后是循环，循环的条件是!_kbhit() && !snake.getIsDead()，意思是只有按下了按键才去获得最新的输入，或者蛇在移动过程中死了，才去退出循环，然后循环的内容主要是依靠现有的方向进行移动，然后移动后适当停顿，然后如果食物数量变为 0 了，就刷新食物，在移动了一格之后，就进行一次页面刷新。最后如果蛇死了，就死亡提示，然后按任意键返回，注意清理掉蛇的申请的内存，然后回到原本的菜单那的界面进行输入玩家姓名，回到那主要是在 easyX 中找不到方便的输入。。。然后输入完后就退出了 play 函数，又回到了菜单。

hardPlay:

第二个模式对应的是将蛇身变成墙的游戏模式，大体上相同，就是退出循环的要求改为了当不能产生新的食物或者新的蛇的时候才退出游戏循环，然后进行死亡提示，然后后续。游戏过程中如果蛇死亡了就调用 snake.snakeToWall()，将蛇身变为墙，并且通过 food.clearFood(snake.gameLeve)，food.randomFood()刷新食物，然后如果不能产生食物和蛇了就标记游戏结束了。

hhhardPlay()：

大体相同，就退出循环改为了死亡了五次或者不能产生新的食物或者新的蛇，蛇死亡后调用 snake.snakeToFood()蛇身转化为食物，然后刷新食物，其他差不多。

funPlay:

这个主要是为了处理加分项的一些 idea，

加速、减速：因为在原本的 move 中有过一些准备，所以在移动中如果蛇头到了对应的区域，就会改变 sleepChange 的数值，进而实现加速减速。

软墙、硬墙：检测蛇头撞到了哪一个墙，上面那一条是软，撞到之后蛇长-3，下面那一条为硬墙，撞到后生命-2.

奖励、炸弹：血量为 4 的时候出现炸弹，不是 4 就不显示，然后出现的范围是 2*2,吃到就生命-2，分数归 0，血量为 2 的时候出现奖励，范围 2*2，不是 2 不显示，吃掉后血量+4。其他也没啥区别了。

AI：

设立了 2 个人工智障一起贪吃蛇，所有加上自己控制的，一开始要初始化 3 条蛇，然后因为有 3 条蛇还重新写了个展示函数，主要是在原来展示的基础上要多展示 2 条蛇，游戏结束循环的条件是玩家操作的蛇的生命值没了，然后在循环中，玩家的蛇就像之前一样，依靠获得键盘的输入，但是 AI 蛇不一样，它先获得一个随机的食物的位置，然后每动一步，就检测那个地方的食物有没有被吃掉，如果没有，就判断下一步该往哪走，如果被吃掉了，就再去获取一下一个食物的位置，然后判断往哪走，注意的是每一个蛇动一步都有可能将食物吃完，所以每走一条蛇就要判断是否要刷新食物，然后走后如果死了，就刷新出一条。

下面介绍一下 AI 蛇的移动规则：在获取了一个食物的坐标之后，先比较食物的 x 和蛇头的 x 来初步决定方向，如果在食物在蛇的左边就往左走，在右边就往右，x 相同就比较 y，来决定是向上还是向下，但是还是要进一步判断的，如果应该是向左，但是之前的方向是向右，那么就要判断如果往上安全就往上转一下，否则往下安全就往下转，如果之前不是向右，那就需要判断一下向左走了之后，会不会碰到墙或者蛇身，如果是的话，就要判断如果往上安全就往上转一下，否则往下安全就往下转。其他的也是类似，比较简单的逻辑吧。

以上就差不多是整体的设计思路了，感谢观看!!!

在实验过程中遇到的问题及解决方法

遇到的第一个问题当然是如何下手的问题。。。
这里必须得说站在前人的肩膀上是很有效率的一种事，当时五一的时候就去网上找到了一些能跑的贪吃蛇，然后来研究他们怎么实现贪吃蛇的，然后也获得了很多灵感吧，但是网上我找到的贪吃蛇都太简陋了，只能够教我入门，但是至少教会了我用 Map 数组来储存图形信息，然后进行判断，还有依靠链表来进行蛇身的设计，还解决了我如何在特定位置输出字符的疑惑，教会了我哪些应该当作类来处理，然后经过差不多 2 天的研究吧，终于能写出一个极其简陋，就只有撞墙和吃食物功能的贪吃蛇。

然后不得不说菜单界面设计是一个很麻烦的事
因为要实现各种功能，然后之中因为读写的文件必需得是存在的文件，所以我后面采取了一种比较凑合的方式，就是在点开程序的时候，如果文件不存在就创立 2 个文件，然后关闭。还有就是上下左右键的 ASCII 码的获取，也是找了好久才知道是有 2 个返回值的，还有比较棘手的历史记录的删改查，也是最后依靠链表的功能解决了。

记录数据入文件的格式问题
其实对于最高分那个问题不大，因为是固定的就那么些数据，只用输出再输入进去就好了，但是对于历史记录而言，它是越来越多的，之前为了将最近一次的历史记录放到最前面，用过在前面换行，但是发现这样总是会把原本在前面的数据覆盖掉，所以最后还是选择加入到文件最后，而依靠在输出的时候的一些变化，将最近一次的游戏数据展示到最前面。

在刷新食物的过程中需要把原本的食物都在 Map 中变为空嘛，但是有时会让在那里的蛇身被误伤为空格
解决方法就是加入了 isEat[][]数组，然后就可以知道在 x,y 处的食物有没有被吃掉了，然后在清理的时候，就只清理那些没有被吃掉的食物。

如何将非图形转化为 easyX 格式
因为我一开始就没有用 easyX 来进行绘制，因为当时想着先把逻辑弄好，再弄这些花里胡哨的，不过也确实这样会让我逻辑好过一点，当时移动的时候本来是想在原本的工程文件中改的，结果太麻烦就直接放弃了，最后还是重新开了个工程文件，然后这时候就显现出了 class 的好处，高度的规模化，对应 Menu 这里面的内容差不多直接复制过来就可以用，对应其他的可以复制过来后优化，然后一个功能一个功能的优化，真的感觉还是可以的，然后里面涉及到的比较麻烦的就是原本消灭掉一个地方的符号只需要将那里变为空格就好了，这样就能掩盖掉，但是在图像里，如果调用单纯的 clear 只会在图像上留下一个黑洞，不能达到效果，当时采取的方式就是对于 clearFood 这样要求的函数，就单独进行一次重新绘制墙，蛇，提示，然后蛇的移动也是先单独绘制一遍墙，食物，但是到最后我自己的逻辑都被搞混乱了，所以有一天早上突然想到为什么不能将这些显示整合到一起呢，于是就有了 showAll 函数，这样就是相当于每走一步就刷新一帧，在逻辑上也更舒服了。

奖励炸弹占了 2*2 的地图，如何做到吃掉一个就把所有的都消失掉
这里主要后面是加入了 isPutV 和 isPutBoom 来判断，后面如果吃到了奖励或者炸弹就把这个置零，然后就是每次移动都会去绘制一遍，如果要，就在相应的位置绘制并在 Map 中加

入字符，否则就不加入。然后因为每次吃到奖励或炸弹，生命值都会改变，所以当生命值改变了之后就判断如果原本那地方还有奖励或炸弹的标记，就变为空白。

然后 AI 那方面的实现问题
总结起来也只能说是在试错中摸索吧，我一开始就是放入一条 AI 蛇然后观察它的活动，然后但是发现了一个问题就是原本向右，现在要向左了，就没办法了，所以就增加了一段 if 语句判断，然后放入 2 条的时候发现会有蛇和蛇相撞的愚蠢行为，所以就又加入了一个 if，用来预测会不会撞到东西，会的话就转向，这样就能避免很多，然后再把人操作的蛇加进去，然后就是关于蛇死后重生的问题，后面的尝试中才发现，需要在每个蛇移动之后都要判断一下它是否死亡了，好及时刷新蛇。

关于在游戏中输入文字的问题
因为之前是依靠 goMapXY 来 cout 语句的，现在需要改成 outtextxy（）需要对距离的掌控比较严了，但是有好处是可以控制背景颜色，字体颜色，字体大小，但是要把数字改为字符还是挺麻烦的。。。

将蛇，食物转化为图片
其实也不算麻烦吧，主要就是原本是在 xy 出填上*或者#的，但是现在因为把地图分成了很多小块，所以一开始我采用的是画个直径为 10 的圆而言，后面为了变为方块图片，就改为在 10*x,10*y 到 10*x+10,10*y+10 的位置范围放入方块，当时其实比我想象的要顺利许多的。

程序编译时经常出现错误：

LNK1104	无法打开文件“D:\高程大作业\大作业-贪吃蛇\我的贪吃蛇图像版\Release\我的贪吃蛇图像版.exe”

就是会发现程序经常挂在后台，然后就运行不了新的 exe，每次都需要我打开任务管理器把它关闭后再次运行才行。
后面我有猜测原因，主要是因为我在运行的时候其实打开了 2 个运行窗口一个是菜单的，一个是游戏的 easyX 的，然后当我在关闭的时候只关闭了一个游戏窗口的时候，原本的菜单窗口仍然在后台运行，没有关闭，这样就出现了上述的问题。