《操作系统实验》综合设计报告



选题名称： 操作系统功能设计与实现

姓 名： 肖 鹏

学 号： 201808010718

专业班级： 智能1802

零、快速引导

* [*整体设计流程图*](#整体设计)
* [*功能介绍*](#功能介绍)
* [*截图展示*](#截图)
* [*问题一：图片转化成字符数组*](#图片转数组)
* [*问题二：游戏角色重合显示问题*](#游戏角色重合)
* [*问题三：绘制地图、边界检测*](#绘制地图、实现边界判断)
* [*问题四：NPC的随机游走、道具随机生成*](#随机)
* [*问题五：NPC追踪道具、追踪/躲避玩家*](#追踪)
* [*课程心得体会*](#心得)

1. 实验目的

通过对操作系统启动过程的模拟，以及软件功能设计与实现，进一步理解和掌握《操作系统实验》课程中学习到的各个功能模块的原理和用法，提高自己的系统设计能力。

1. 实验环境

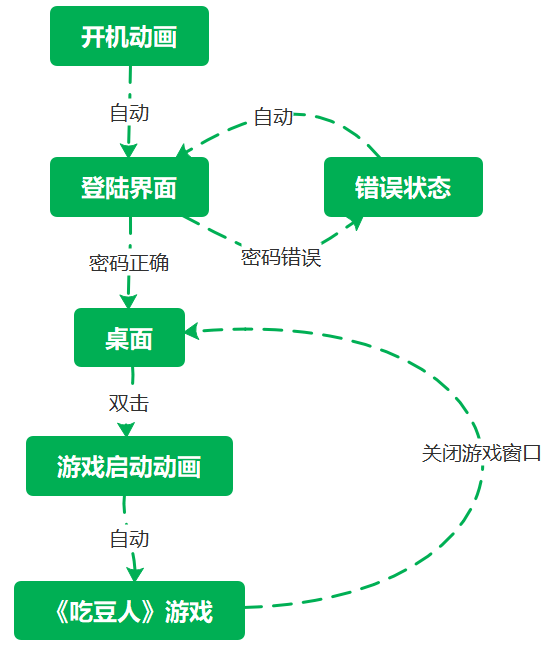
硬件环境：计算机，无特殊配置要求

软件环境：QEMU虚拟机、批处理文件、相关编译工具等

语言环境：C语言、汇编语言

1. 功能介绍及运行效果截图

整体设计介绍



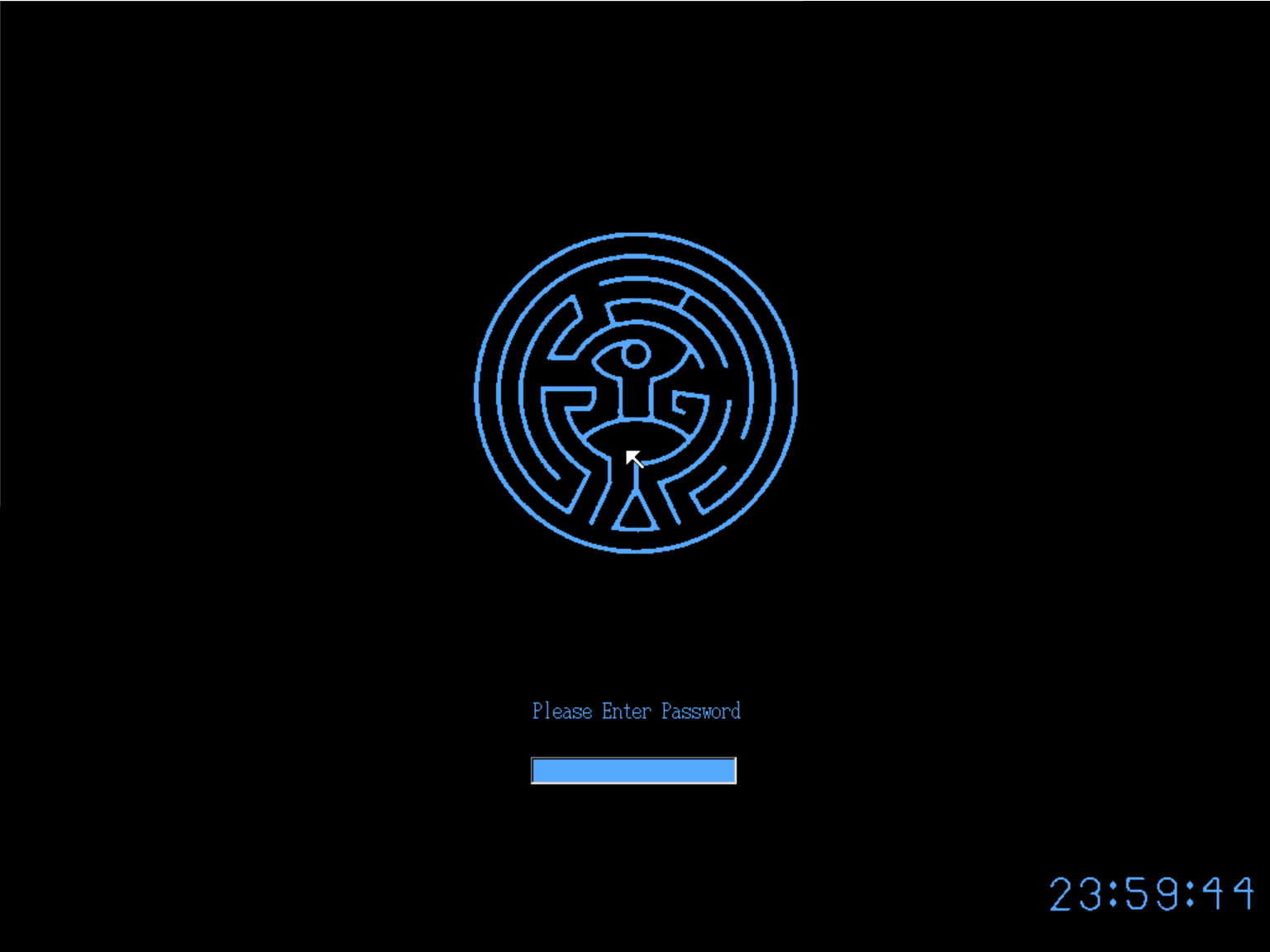
*（系统状态图）*

功能介绍

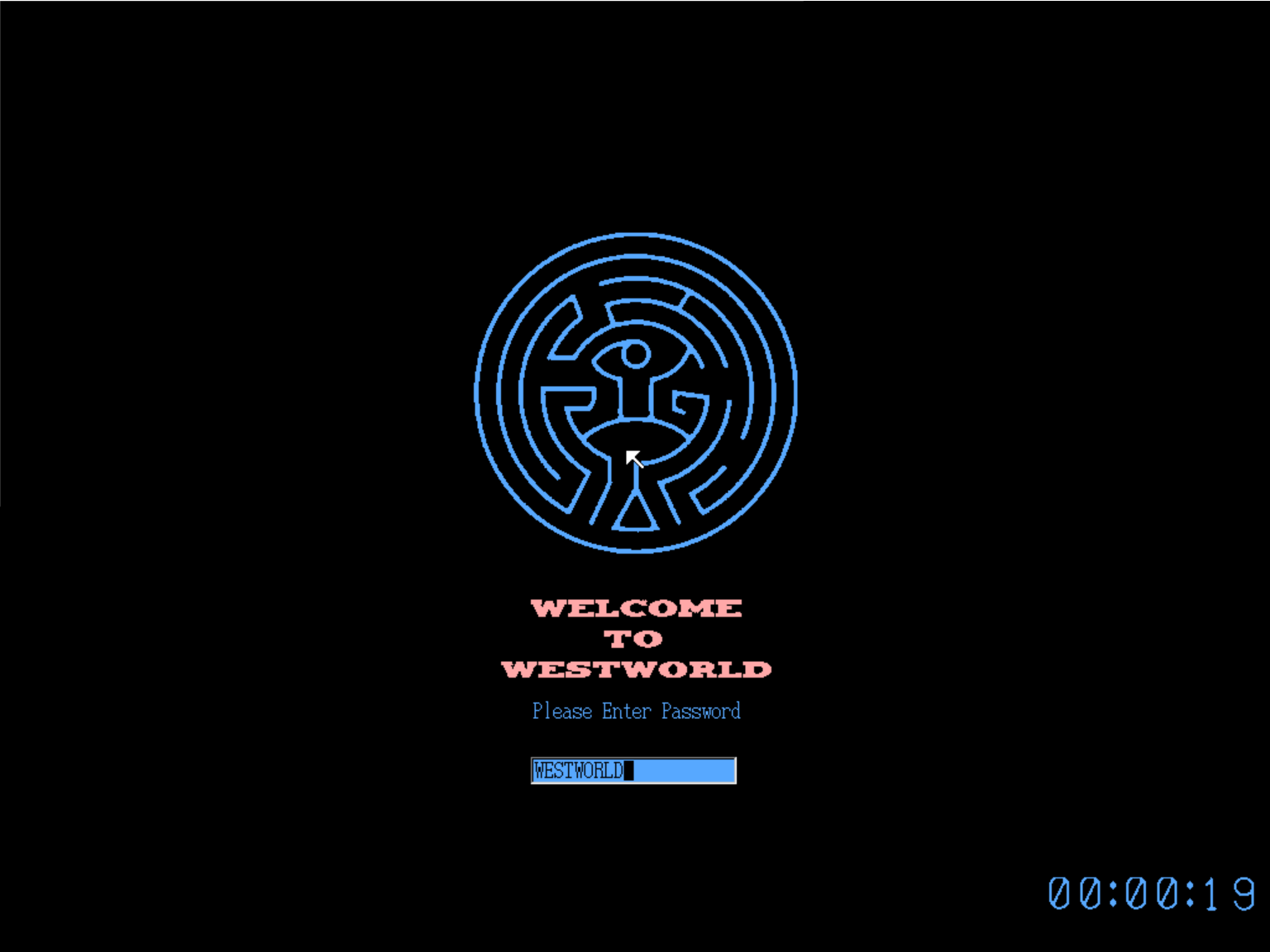
1. 开机界面动画，比较简洁，是对一个图案的逐渐刷新以达到动态显示的效果。而且开机后时钟启动。
2. 桌面显示LOGO、时钟。
3. 游戏有进入动画。
4. “吃豆人”游戏

* 玩家控制的“pacman”角色初始有5s无敌时间，在无敌时间内可以“吃掉”四个颜色的NPC——“ghost”，无敌时间在右边游戏状态栏显示。
* “pacman”每拾取一个道具，游戏积分加50，并且获得7s额外的无敌时间。
* “pacman”每吃掉一个“ghost”，游戏积分加200
* “ghost”在其NPC“盟友”被吃掉的情况下每拾取一个道具，复活一名死亡的NPC——“ghost”
* “pacman”在正常状态下，四个“ghost”中有两个会全力“追杀”玩家所操控的“pacman”、另外两个“ghost”会全力“搜集”道具。所以，NPC会尽所能来限制玩家，尽力取胜吧！
* Tips：“pacman”在正常状态下跑的比“ghost”慢哦，所以躲避不是长久之计，努力寻找道具吧！
* Tips：不小心“阵亡”了的话，按“enter”原地复活喔！

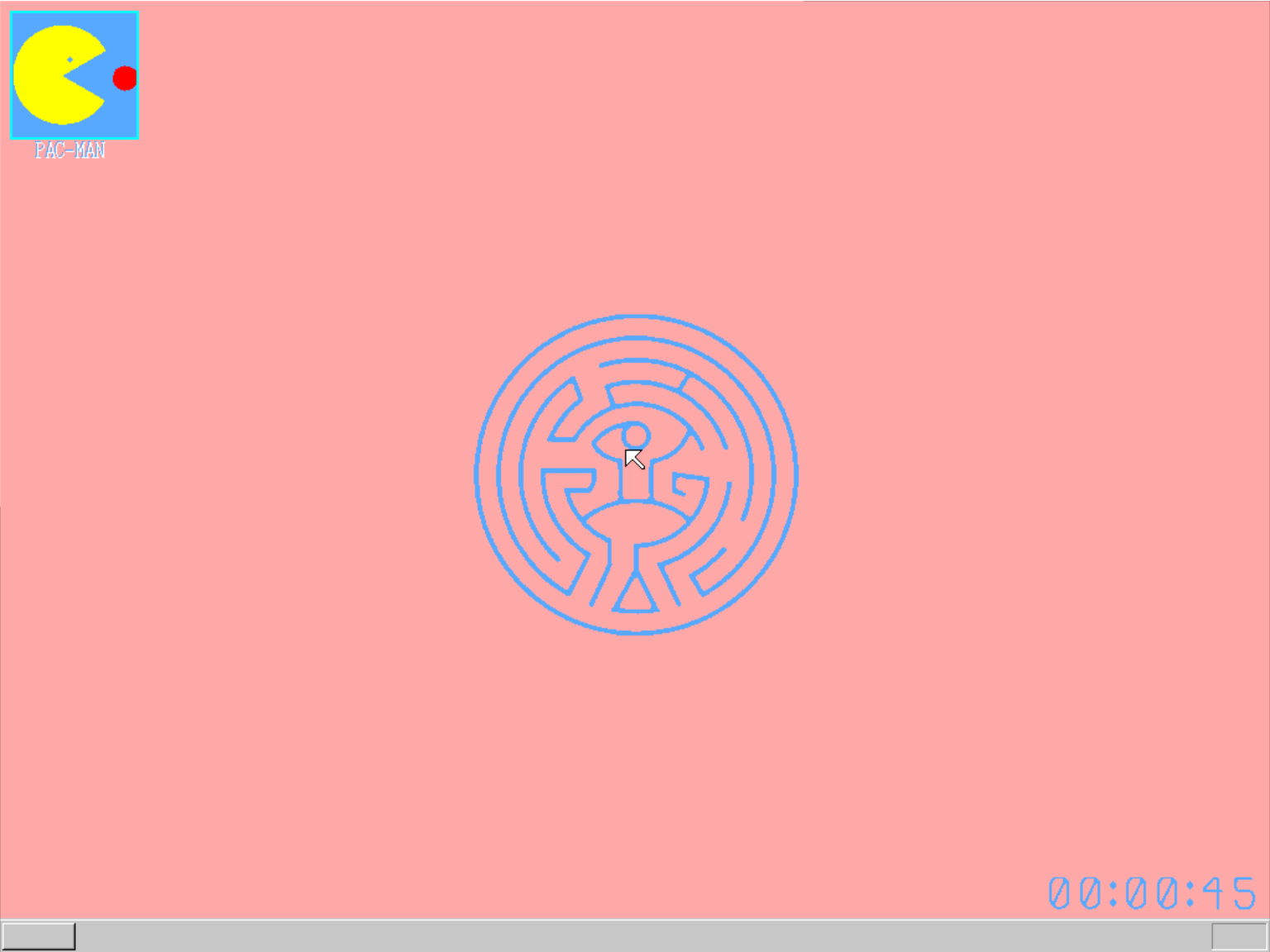
界面展示



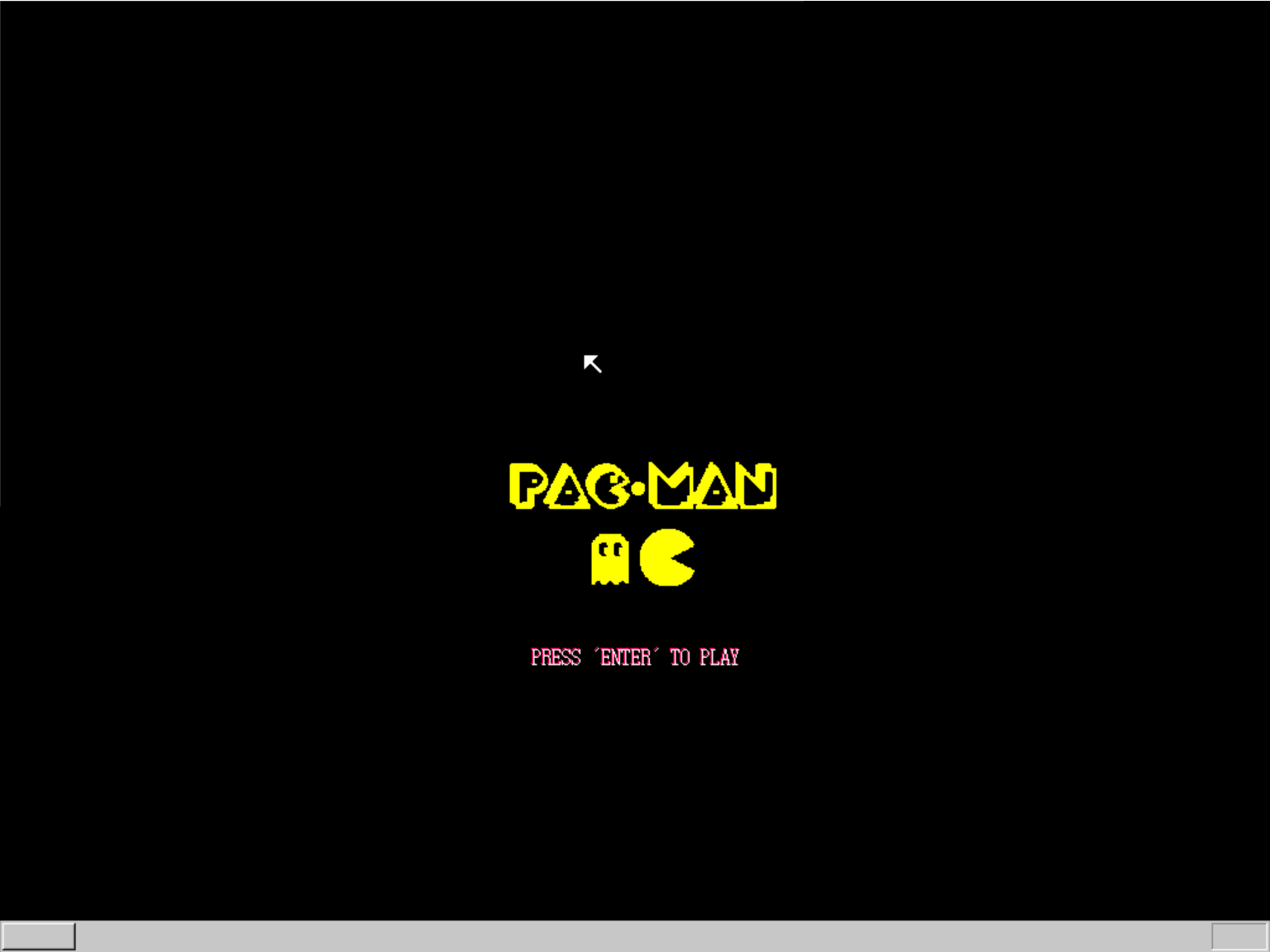
*（开机界面）*



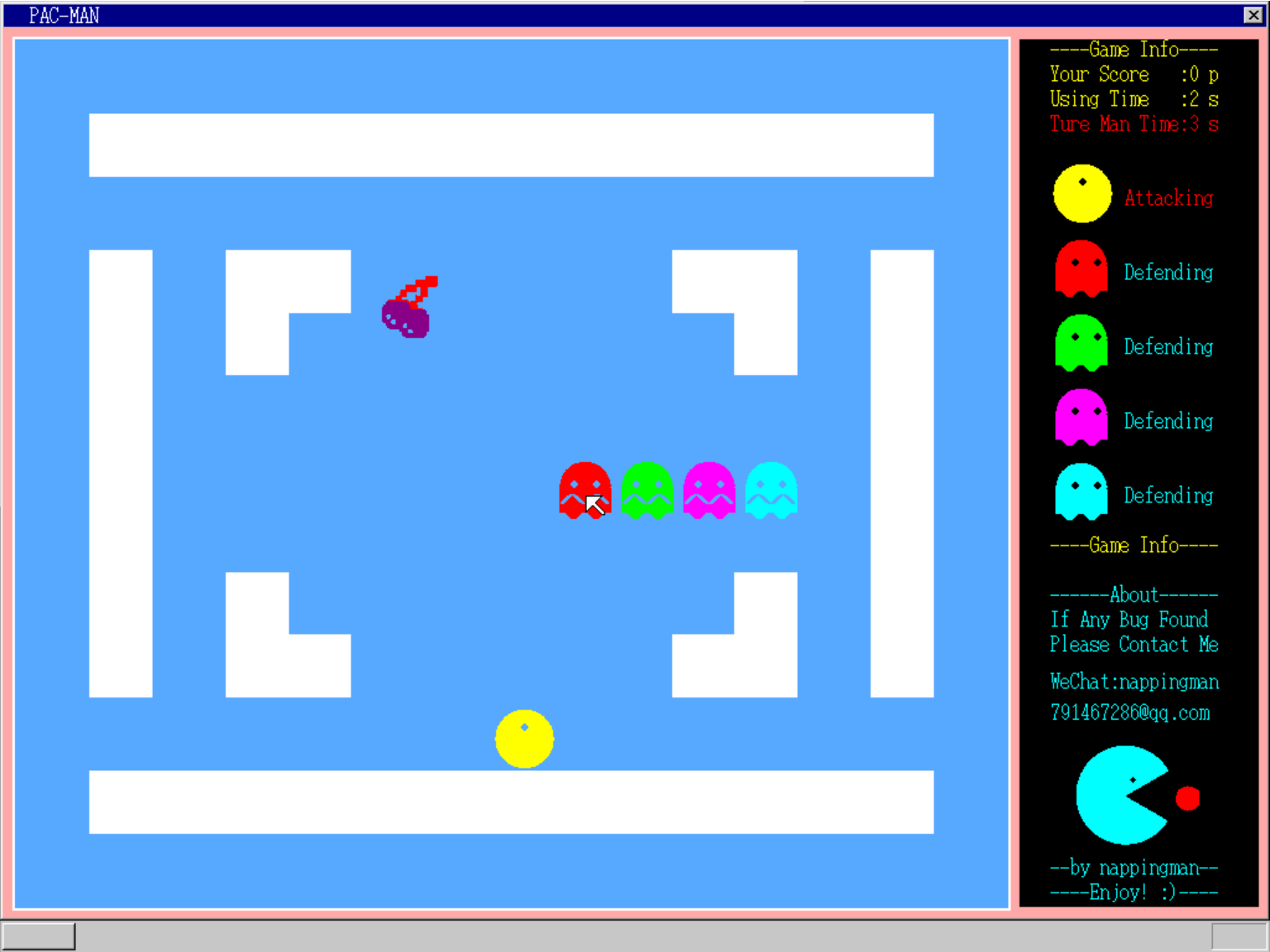
*(密码正确欢迎动画)*

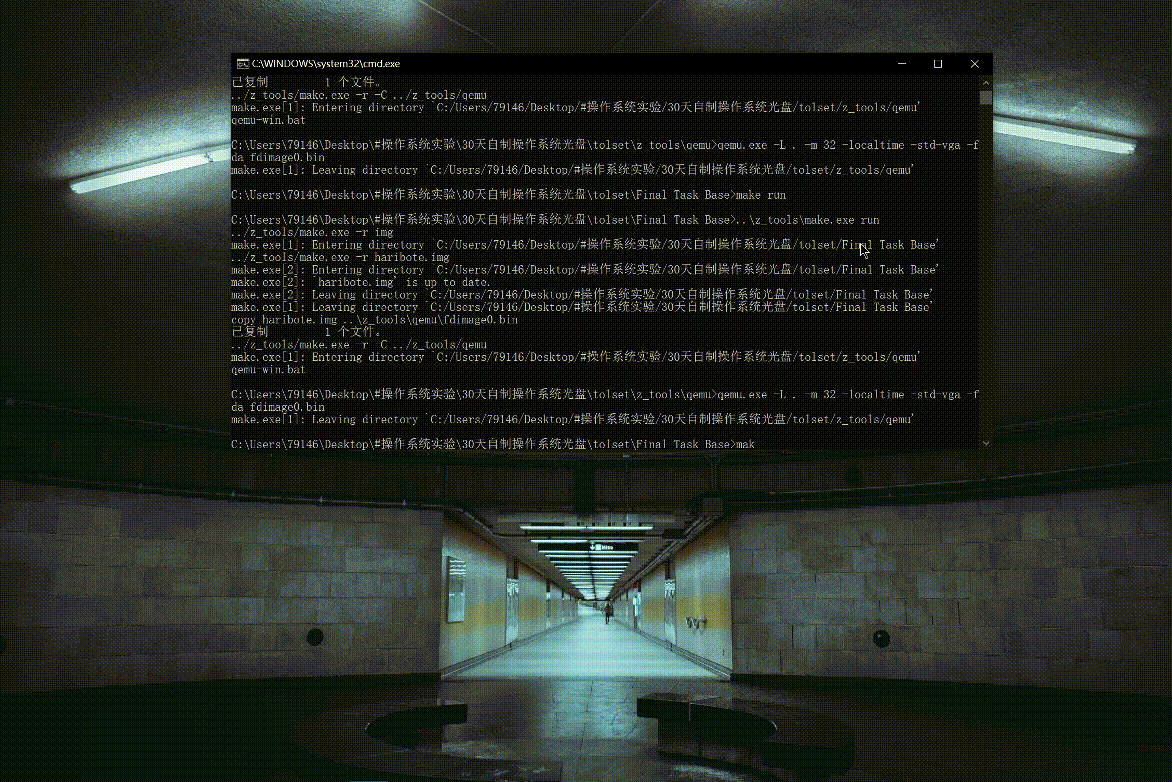


*（桌面）*



*（进入游戏动画）*

*（游戏主界面）*



*（运行GIF）*

1. 遇到的问题及解决方法

问题1：图片转化成字符数组

在设计开机动画的时候，就想要把一些自己喜欢的图案给打印出来；之后在设计游戏角色的时候，必不可免的人物动画也是要画出一些精致的图案的。那么，对于这些动辄100\*100像素大小的图片，怎么能够用一种高效的方式转化成我们能够直接使用的形式——字符数组呢？

解决方法：灰度图片转numpy数组

一番摸索之后，最终总结出以下python代码！



*（图片转数组函数）*

代码的原理是，把输入图片转化成灰度图像，然后根据灰度值，在同规模的二维数组中赋予相应的标记值。一般我们只需要提取2~3种颜色的图片，当然如果想要做的更多姿多彩也是可以的。

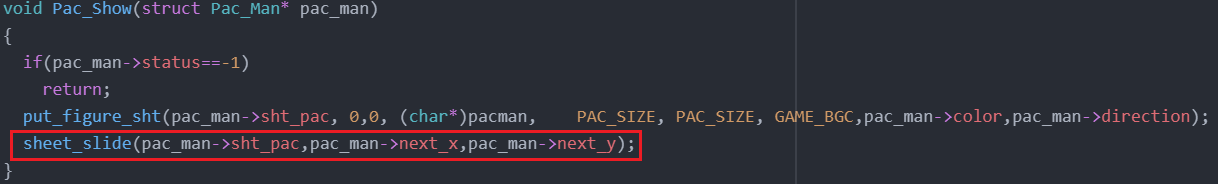
当图片过大时，则需要想hankaku.txt那样，使用bin2obj编译了，详情可以参考Day5的代码。

问题2：解决游戏角色之间的重合问题

在实现游戏人物可以正常移动之后，遇到的第一个问题就是，当人物重叠时，如何显示地更为自然。由于我一开始是在一个sht\_game图层上显示所有的人物，所以每次人物的移动就像实现鼠标移动一样（先把旧坐标处的人物消除、再在新坐标处显示出人物），这样的话每次都是在操作一个矩形，会使得人物重叠时的效果很差，显示不完全。

解决方法：使用多图层！

每个人物对应一个图层，这样的方式十分高效，因为对于图层的移动我们之前的实验中以及实现而且封装到一个sheet\_slide函数中了，这样我们对于人物的移动，基本上只需要调用一行图层slide就可以了，比之前自己写的代码即简洁、又高性能！



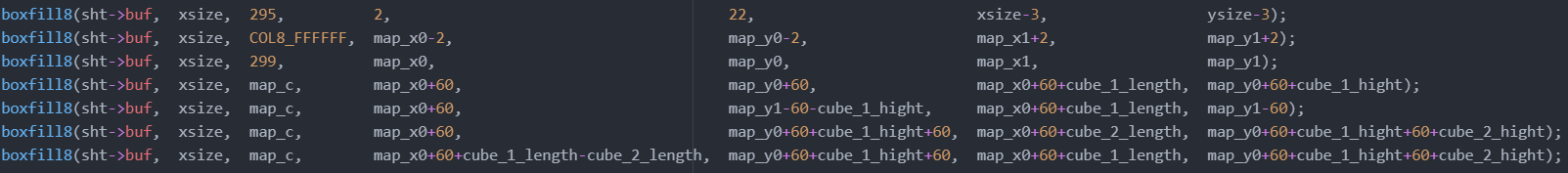
*（优化后的人物移动）*

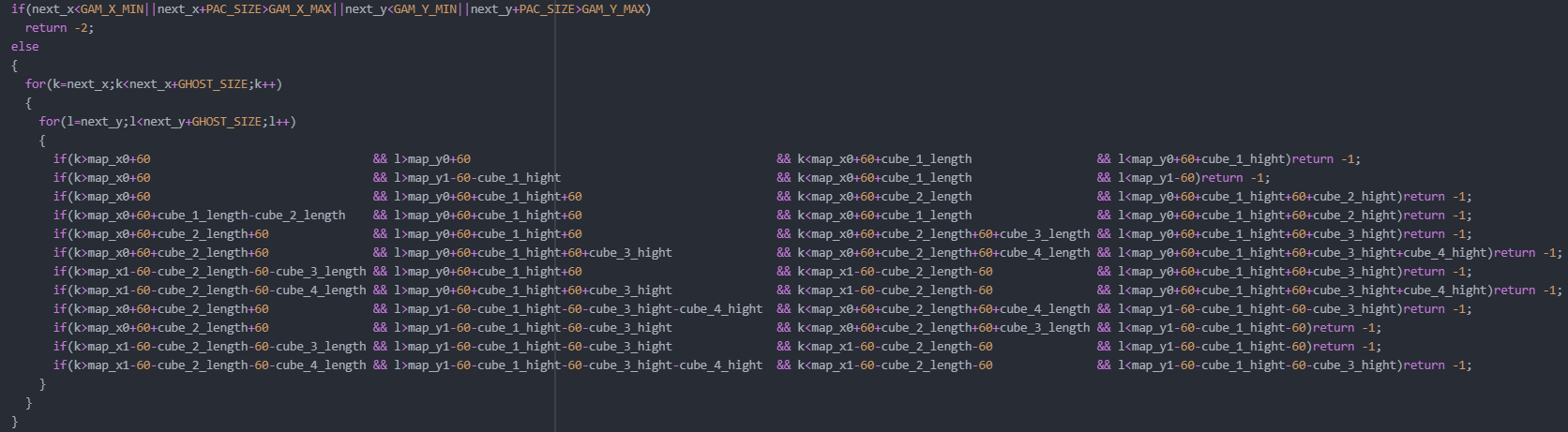
问题3：绘制地图、实现边界判断

一开始也想要使用图片导入的方法设计地图，但是尝试过很多次后，效果并不是很好，原因有两个：①找不到很适合的图片，大多数图片的分辨率不高，导致转化后的数组有很多噪音点，考虑到这些点可能会对边界检测有所影响便没有继续。②图片的比例不适合已经设计好的人物大小，当时已经根据分辨率设计好了50\*50大小的人物大小，但是找到的地图中间通道的宽度都没有很贴切这个尺寸。

解决方法：使用boxfill8函数，手动绘制地图

后来一狠心（其实也是时间不够了），想着算了，阉割一下地图，做的稍微简单一点，就直接用12个矩形实现了一个地图。这样做边界检测也很方便了，直接判断点是不是在这12个矩形之中就可以了。

 这样大大减少了在地图上的工作，但是为后面的人物游走埋下了一点点麻烦。

*（部分绘制地图代码）*

*（判断边界代码）*

问题4：NPC的随机游走、道具随机生成

最开始调试的时候，给的设想是人物随机走动。但是生成随机数的时候有点问题，头文件不能包括进来，最后折中了一个实现，找了100个数作为随即数表，然后用一个全局下标，实现了一个伪随机。

这样每个NPC的方向是可以随机的，然后现在出现新的问题——NPC撞墙了怎么办？

解决方法：方向设定、坐标微调、循环生成坐标直到合格

最开始弄的时候各种bug——人物穿墙走动，卡墙，还有的时候走着走着人没了……

反正就是各种令人懵的bug。

后来想到了一个比较合理的解决方法：

对于NPC，在每次撞墙后，根据撞墙前的行动方向给其设定一个新的方向，对于卡墙——明明空间够，但是无法走进通道的问题，也是根据移动的方向，每次对坐标微调（1个像素单位），直到下一个位置通过边界检测：



*（NPC撞墙后自动更改路线）*

对于玩家部分，就不需要方向上的检测了，我们只需要在卡墙问题上加上相同的处理方式，这样就能够99%解决我们现在的问题了，为啥是99%，因为即使是最终的代码里面，在人物移动步长变大后，个别地方还是会卡不进去，但是我们就当问题解决了吧！

对于道具的随机生成，所做的也是边界检测，因为随机数表的存在，我这里使用的是一种比较低效率的方式，就是当随机生成了一个不合格的坐标时，陷入while循环，直到生成的坐标合格，再使用。虽然在运行流畅度上没有发现很大的问题，但是在道具位置的多样性上有点差，给人的感觉变来变去就那几个位置，但不是我们的主要问题，就当作解决了吧！

问题5：实现NPC对玩家控制角色的追踪、躲避，以及对追踪道具

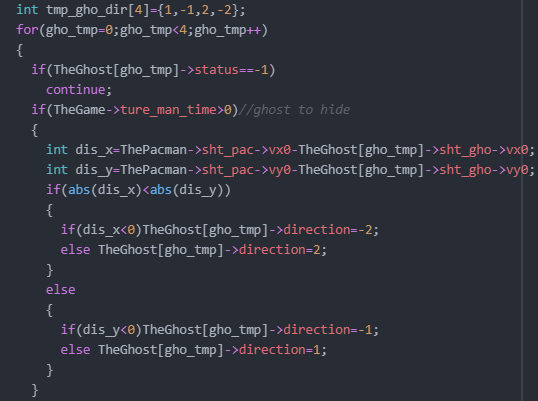
在验收的前一两天，突然想到好像做一个追着人跑的NPC貌似也不是很难，由于NPC的方向更改部分已经完善了，那么根据玩家和NPC的坐标差值来相应给每个NPC赋予新的方向，岂不是比随机方向更好一些？

解决方法：判断x、y轴坐标差距

想实现的就是：NPC找道具、NPC追玩家、NPC躲玩家这三个功能，其实就相当于一个功能。

实现起来也很简单，我们每次在要改变方向的时候，计算一下不同角色之间x轴和y轴的差距，根据大小和正负，我们可以得到两个角色在水平方向还是竖直方向更近、想要躲避/追逐是要往哪个方向走。

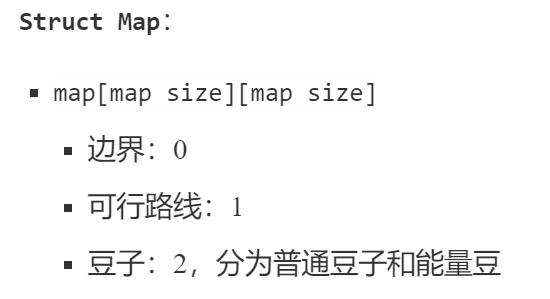
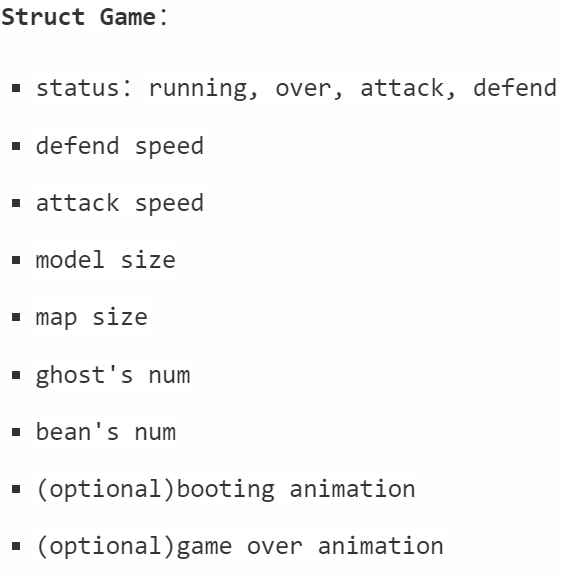
实现代码如下，很清晰，容易懂：

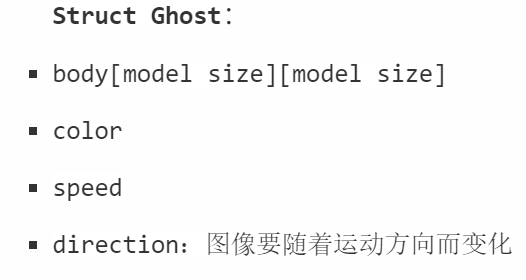


*（NPC躲避玩家）*

心得体会

做这个综合设计从题目公布的时候就已经有了大致的想法了，当天实验晚上还写了一下几个结构体的大致设计：







不过后面慢慢动手做起来的时候，有很多修改，有的功能没有实现，有的实现的效果也还不错，和我最初想的一样。

主要花的时间是元旦假期吧，上午睡到快中午了然后起来写（哈哈哈哈哈懒🐕属性暴露），感觉主要花的时间是在素材的搜集和总体框架的设计上，因为我比较容易纠结，“这一张图片好看吗？”，“感觉还是上一张更好看”这类问题。。。

框架的改动属于工作量比较大的，往往是写到后面，由于代码太长了，又得跑到定义的地方修改参数。。。

怎么说呢，当初想做这个题目的时候没有想到会有这么多工作量，光摸索一条图片转化数组的方式我觉得就占了总时间的10%，但好在最后还是弄出来了一个像样的东西了。

可能是我导入的数组太多了，在后面还想要做一个屏保功能的时候，想导入新的图案，发现数组加进去后代码就崩了，跑不出来，后来也就没做屏保了。

说一说整体感受吧，其实我在做的时候就有很强的感觉，觉得自己是在做软件开发之类的，而不是操作系统的实验。。。觉得有点矛盾吧，虽然这整本书也说得是设计一个图形界面操作系统，但涉及到系统层面的代码其实都是作者封装好的，我做到快3/4的时候就觉得，我们都是在验证作者写的这些代码有多好，重用性有多高，我们这么多人写的都不一样，但都能适用，说明作者的代码写的多好！

emmmmm，整个实验课程我觉得很不错，是这几门核心课里面上的最舒服的一门了，每次周四晚上去实验室，然后边听音乐边看书写代码，偶尔抬头瞅一眼窗外，真的很舒服，老师也很好！

哈哈哈哈哈哈哈，最后一次报告写完啦！

附一张某次做实验时候拍的照片作为结束……

