操作系统课程设计项目文档

——汉诺塔

# 一、开发环境

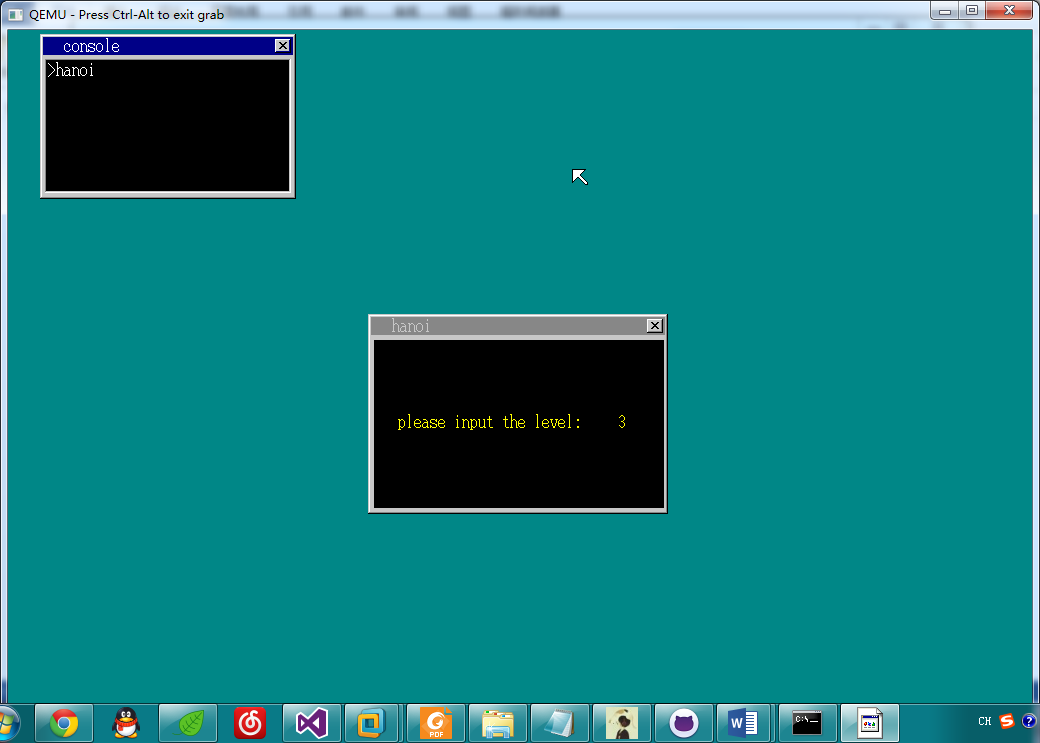
参考书目《三十天自制操作系统》，开发系统windows7，模拟器QEMU，汇编编译器以及C编译器均为书中自带编译器。

# 二、项目要求

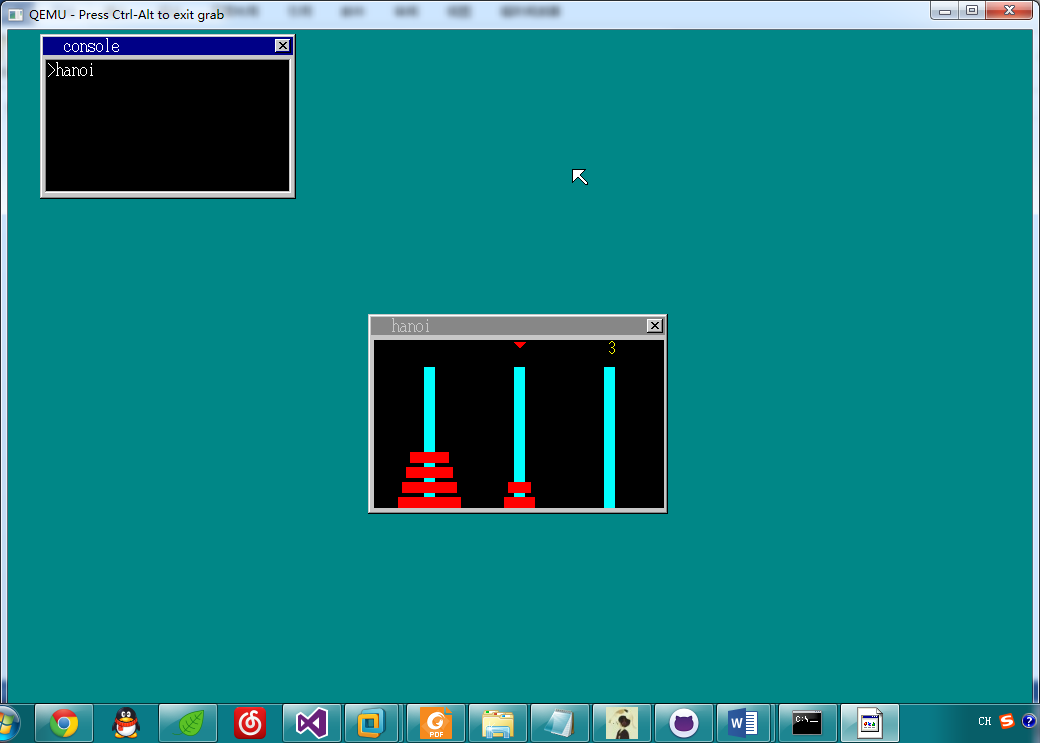
在书中自带系统源码基础上，开发一个用户级应用。此处开发的用户级应用为汉诺塔游戏，支持3级到9级选择（其实可以更多级，但是内存限制问题难以将窗口更大，选择将窗口置为300\*200），支持游戏胜利条件达成后自动结束游戏，并且记录游戏的步数。

# 三、项目界面

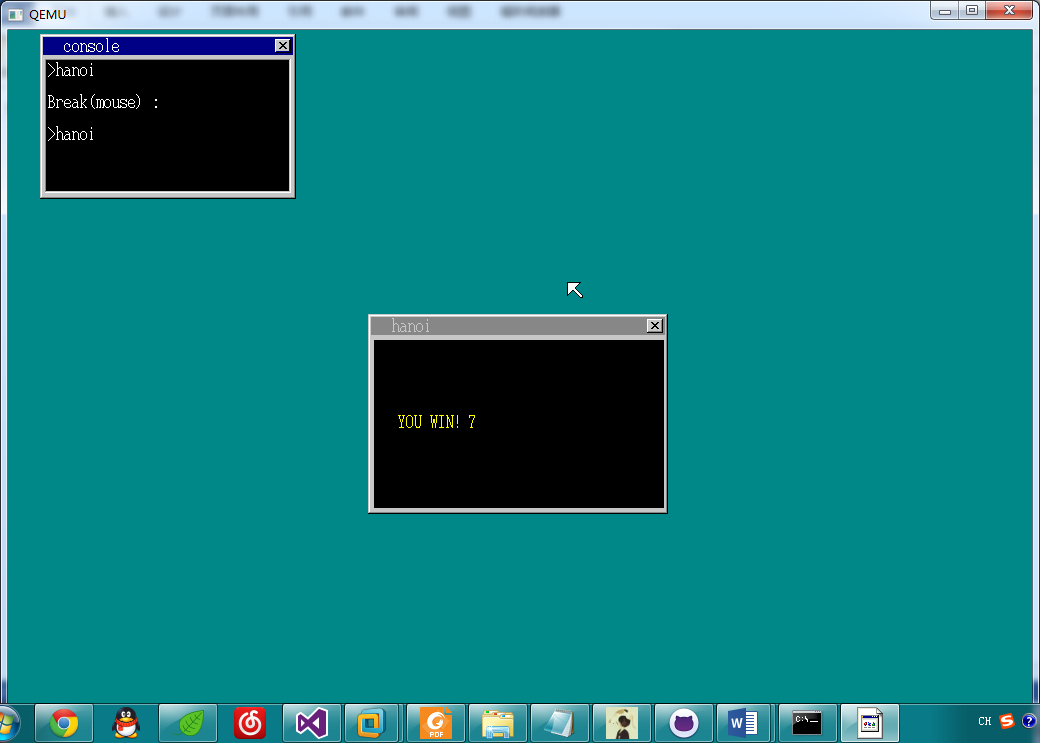
开始界面：



游戏界面：

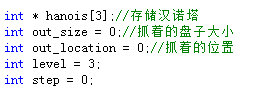


游戏结束界面：



# 四、项目实现

## 1）数据结构实现

因为c语言标准库中并没有栈这种数据结构，所以我选择了使用一个二维数组对栈进行模拟。首先我设置了5个全局变量，第一个即为对于汉诺塔主体结构的模拟。通过设置一个1维的整型指针数组来存储每个柱子上的盘子数量以及大小；带有out的整型变量则是存储抓着的盘子大小以及抓手当前的位置；而剩下的两个变量，则是用于记录这关游戏的等级以及这关的步数。

与此同时，我写了1个关于数据结构上的子函数hanoi\_gettop，正如我前面所说，c的标准库中并没有栈，于是我采用这个函数来把数组模拟成栈。这个函数的作用是输入一个i值，即hanois这个以为整型指针数组的下标，来获得这个它“最上面”的值。如果这个指针对应的数组为空则返回-1，否则返回最上面一个有实际数字的值的下标。通过这个函数返回值的判断，可以对数组进行出栈以及压栈的操作。

其次要说明的是对于系统api的调用，在这个系统中，char为一个字节，但是int是4字节，在api\_malloc中的分配规则是分配字节数，所以要把level\*4给这个int\*分配内存。

然后要说的是四个操作函数up,down,left,right：这四个函数是对于位置的操作，左右是控制out\_location，上下是操作out\_size。除此之外也已经对边界情况进行了处理，避免报错。

最后一点要说明的就是关于游戏胜利的检测：只有在游戏按下down时才可能胜利，所以仅仅在这一刻进行了检测。检测若第三个柱子最上面一个盘子的位置为level-1的时候就是胜利（因为数组从0开始计算）。胜利时则跳出当前循环，重新绘制图案，打印you win字样并显示所用的步数。

## 2）界面实现

这个操作系统中自带某些绘制函数的api，在这里我用到了矩形的绘制、字符串的打印、键盘输入的检测以及对线段的绘制进行了封装。除了主界面绘制之外，在汉诺塔的柱子以及盘子的绘制上也是选择了矩形的绘制函数。在这里要特别注明的是，这里对于线段绘制进行了一次封装，使得可以绘制三角形。具体方法就是选定两个点进行逐步绘制，并不停使纵坐标+1、横坐标靠拢，直到横坐标相遇。三角形的绘制用来显示当前抓手的位置。并且每有一次键盘有效操作都会触发相应的操作指令，并对界面进行重新绘制，这就是界面刷新的方式。

# 五、小结

在这次项目中，虽然很遗憾的没有能写到更加底层的东西，但是也对系统的底层实现有了更加深刻的了解，尤其是对于内存的分配方式上，了解了每个系统的实现都是不同的，每种对于硬件的管理方式都不是完美的。另外也很大程度上明白了关于windows中界面的实现。虽然本来想对这个系统的内存管理方式进行修改，但是因为不懂汇编所以没有办法修改某些内核中的问题。而且因为时间问题，整本书读的囫囵吞枣很不仔细。也许在暑假中我会重新把这本书仔细读一遍或者研究一下windows中更加底层的api，来实现一些原来想要实现的程序。