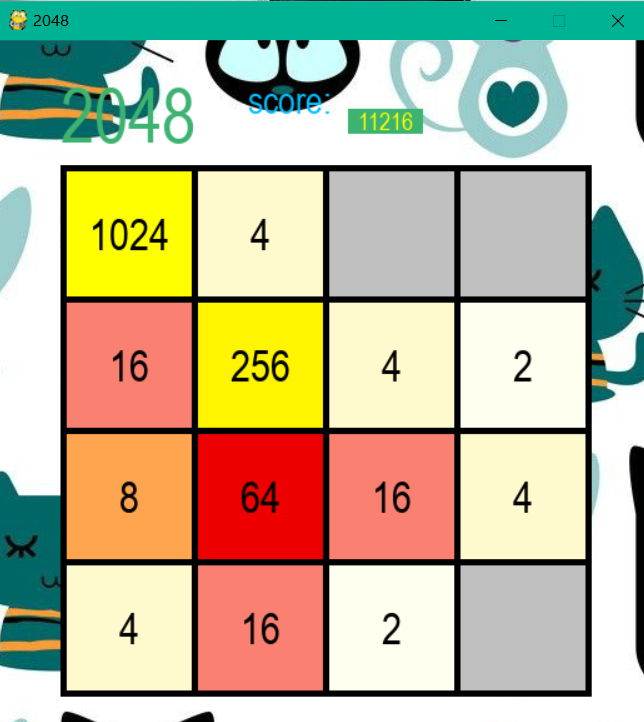
Python实验报告

1. 简介：

2048的游戏规则很简单，每次可以选择上下左右其中一个方向去滑动，每滑动一次，所有的数字方块都会往滑动的方向靠拢外，系统也会在空白的地方乱数出现一个数字方块，相同数字的方块在靠拢、相撞时会相加。系统给予的数字方块不是2就是4，玩家要想办法在这小小的16格范围中凑出“2048”这个数字方块。游戏的画面很简单，一开始整体16个方格大部分都是灰色的，当玩家拼图出现数字之后就会改变颜色，整体格调很是简单。在玩法规则也非常的简单，一开始方格内会出现2或者4等这两个小数字，玩家只需要上下左右其中一个方向来移动出现的数字，所有的数字就会向滑动的方向靠拢，而滑出的空白方块就会随机出现一个数字，相同的数字相撞时会叠加靠拢，然后一直这样，不断的叠加最终拼凑出2048这个数字就算成功。



1. 主要设计：

2048游戏的主要算法就是实现数字的移动和相加，我一开始是想的一个数字有四个相邻的位置，上下左右用列表存储，然后遍历判断，但是真正写的时候发现有很多重复的情况存在，而且位置的存储没有办法确定。所以最后写了一个只实现相加的函数，然后再分别从四个方向去调用它。具体的就是用一个列表，判断它的长度也就是存储连续的相邻的数字个数，然后判断是否有相邻的元素相等，如果有的话相加，单一的单一存储，累计分数也是在这个函数里实现的。具体算法如下：

**def** combinate(L): *#实现相加操作* **global** score  
 ans = [0, 0, 0, 0]  
 num = []  
 **for** i **in** L:  
 **if** i != 0:  
 num.append(i)  
 length = len(num)  
 **if** length == 4: *#当有四个数的多种情况* **if** num[0] == num[1]:  
 ans[0] = num[0] + num[1]  
 score += ans[0]  
 **if** num[2] == num[3]:  
 ans[1] = num[2] + num[3]  
 score += ans[1]  
 **else**:  
 ans[1] = num[2]  
 ans[2] = num[3]  
 **elif** num[1] == num[2]:  
 ans[0] = num[0]  
 ans[1] = num[1] + num[2]  
 ans[2] = num[3]  
 score += ans[1]  
 **elif** num[2] == num[3]:  
 ans[0] = num[0]  
 ans[1] = num[1]  
 ans[2] = num[2] + num[3]  
 score += ans[2]  
 **else**:  
 **for** i **in** range(length):  
 ans[i] = num[i]  
 **elif** length == 3: *#有三个数* **if** num[0] == num[1]:  
 ans[0] = num[0] + num[1]  
 ans[1] = num[2]  
 score += ans[0]  
 **elif** num[1] == num[2]:  
 ans[0] = num[0]  
 ans[1] = num[1] + num[2]  
 score += ans[1]  
 **else**:  
 **for** i **in** range(length):  
 ans[i] = num[i]  
 **elif** length == 2: *#有两个数* **if** num[0] == num[1]:  
 ans[0] = num[0] + num[1]  
 score += ans[0]  
 **else**:  
 **for** i **in** range(length):  
 ans[i] = num[i]  
 **elif** length == 1: *#有一个数* ans[0] = num[0]  
 **else**:  
 **pass  
 return** ans

1. 详细过程：

1、生成数字为0、2、4、8……如果是0则不显示数字，如果是其他数字则根据数字判断底色

2、实现上下左右滑动事件，基本原理是这样，先去判断每行（列），以为移动后的序列，将这4个值存入一个列表，根据游戏规则计算，再返回到这行（列），说的比较抽象，我们可以这样看：2 2 0 4 假如这是某行，我们现在左移，首先将2 2 0 4 存入一个列表（去掉0），即列表为 2 2 4 然后根据规则2和2合并，则列表为 4 4，最后返回到这一行（列）。

3、判断游戏结束，若空格有0，游戏一定没有结束，若为0，则判断每行一一相邻是否存在相同数字，每列一一相邻是否存在相同数字

4、画方格是用了一个类来存储方格的各种属性，也是整个游戏里唯一的一个类

四、总结：

Python的小游戏也算是弥补了C语言没有做的了的遗憾了。整个代码敲下来也不容易。一开始各种不熟悉python的语法，和C语言各种混淆，不由自主的敲分号，但是真正熟悉起来以后python还是比C语言简单多了。在写这个游戏以前我也在网上找了不少的代码来看，给了我不少启示，然后才着手写它，很多不理解的函数都是一个一个百度来的用法，看了人家的写法做参考一点一点实现。而且我发现了自己的一个大问题就是我的逻辑真的有问题，虽然不容易，但是最终也是完成了。对于python我学到的可能连皮毛都算不上，还是要找机会好好学习一下，弥补自己的缺漏。