**路由技术与原理程序设计**

**--4096游戏的设计与实现**

**实**

**验**

**报**

**告**

**班 级 2016211302**

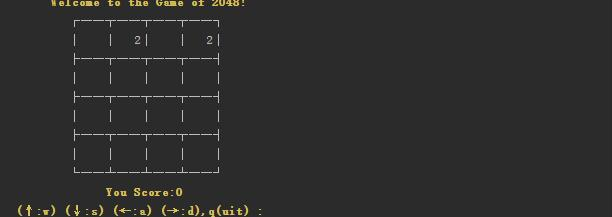
**姓 名 毛维嘉**

**学 号 2016211067**

1. 实验内容

Python 实现无界面的4096游戏

**项目描述：**



**项目要求：**

* + - 以4\*4矩阵形式呈现
    - 初始化生成随机数
    - 将游戏结束标志改为4096（提高）

1. 实验原理方法
2. 界面设计

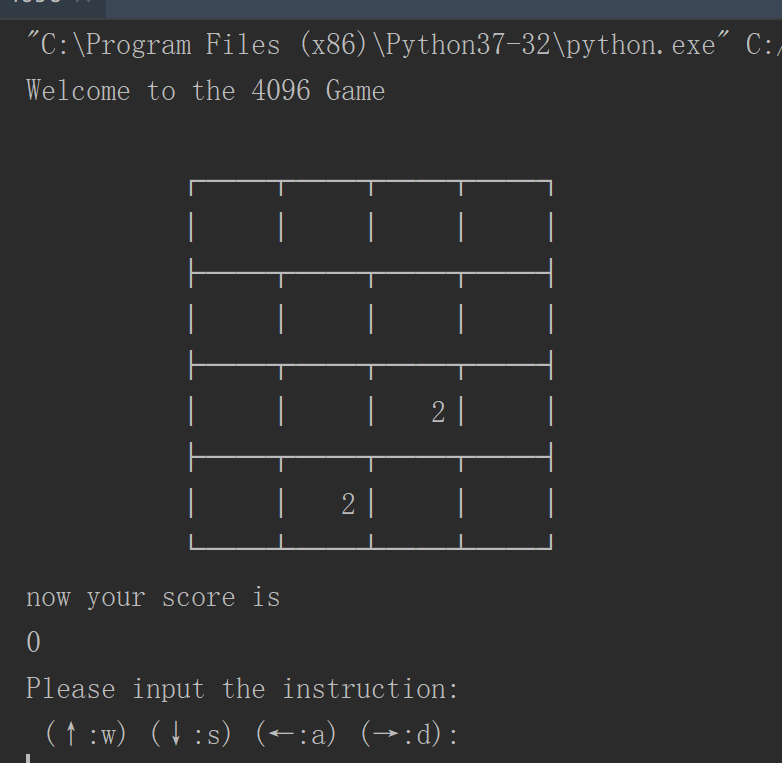
由于没有采用图形库，所以我们用命令行进行结果的显示和输出，这里面我们只需要打印出相应线条来在命令行上组合成相应的表格。然后在每个格的中央打印出对应的数值即可

然后在整个图形的下方首先显示分数，就是截止到目前，玩家所获得的总分数是多少。

下面再次显示游戏指令，显示按键操作。

然后下方就是输入行，等待用户输入指令。

以下是基本界面



1. 游戏初始化

由于没有采用图形库，所以我们用命令行进行结果的显示和输出，这里面我们只需要打印出相应线条来在命令行上组合成相应的表格。然后在每个格的中央打印出对应的数值即可

然后在整个图形的下方首先显示分数，就是截止到目前，玩家所获得的总分数是多少。

下面再次显示游戏指令，显示按键操作。

然后下方就是输入行，等待用户输入指令。

首先游戏刚开始的时候，我们打印图形，并且利用随机数生成的函数在界面任意空的位置生成2 或者 4。

1. 接受用户操作

等到用户输入相应指令之后，代码解析指令，判断此时用户输入的指令是向上下左右四个方向移动，还是无效操作，解析出来之后对棋盘进行移动和相应计算

1. 重新更新表格

我们在得到用户指令之后，遍历整个数组，判断数字位置是否可以往相应方向移动。这分为以下几种情况，首先是在数字移动的方向没有遇到阻碍，这样的情况只需要我们将数字移动到相应的最上下左右的位置就可以了。第二种情况就是在移动数字的过程中遇到了阻碍，然后发现两个数字相同可以合并，那么数值相加变为一个数，再次移动到最上下左右的位置，这个时候同时需要更新分数，发现分数增加了两分或者四分。最后一种情况是在移动的过程中遇到了数值不同的数字，那么这个时候不能合并，同时移动这两个数字到终点。

重新计算并且更新了二维数组并且重新得到了新的分数之后，就可以再次调用界面显示函数显示当前界面，以及等待用户进行下一次操作。

这里面我们罗列一种情况的函数

def goDown():  
 global score  
 for i in range(4):  
 for j in range(3, 0, -1):  
 for k in range(j - 1, -1, -1):  
 if matrix[k][i] > 0:  
 if matrix[j][i] == 0:  
 matrix[j][i] = matrix[k][i]  
 matrix[k][i] = 0  
 elif matrix[j][i] == matrix[k][i]:  
 matrix[j][i] \*= 2  
 score += matrix[j][i]  
 matrix[k][i] = 0  
 break  
 AddRandom()

1. 判断游戏是否结束

这里面的情况有两种，一种是玩家在取得4096这个分数前，已经没有办法在移动数字，这就代表着玩家失败游戏已经结束，这里我们用的方法是判断是否还有空格，或者还有两个数字一样并且在相邻的位置。如果有上述情况发生，那就代表着游戏没有结束，如果没有则代表游戏结束，玩家失败。

第二种是玩家已经胜利，这种就直接判断分数就可以，如果分数已经到达了4096，则玩家胜利。

def check(): # 检查游戏是否GG  
 for i in range(4):  
 for j in range(3):  
 if matrix[i][j] == 0 or matrix[i][j] == matrix[i][j + 1] or matrix[j][i] == matrix[j + 1][i]:  
 return True  
 else:  
 return False

def ifwin():  
 for i in range(4):  
 for j in range(4):  
 if matrix[i][j]==4096:  
 return True  
 return False

1. 实验环境

Python 3.7

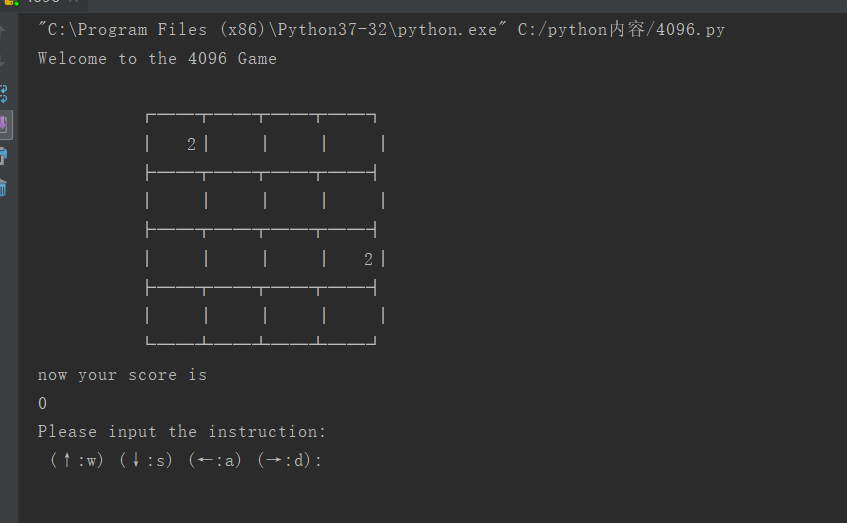
IDE：pycharm

1. 函数列表

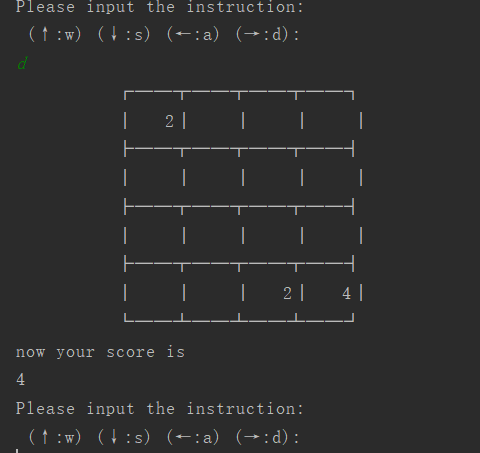
|  |  |  |
| --- | --- | --- |
| 函数名 | 参数 | 作用 |
| Def showPic | 无 | 进行命令行界面显示 |
| Def initGame | 无 | 进行游戏的初始化 |
| Def check | 无 | 检查游戏是否已经结束 |
| Def AddRandom | 无 | 添加随机数 |
| Def goLeft | 无 | 响应用户向左移动的指令 |
| Def goRight | 无 | 响应用户向右移动的指令 |
| Def goDown | 无 | 响应用户向下移动的指令 |
| Def goUp | 无 | 响应用户向上移动的指令 |
| Def ifwin | 无 | 判断用户是否已经胜利 |
| Def checkEmpty | 无 | 判断表格是否已满 |
| Def notzeor | 无 | 判断是否为空 |

1. 程序运行结果

初始化界面

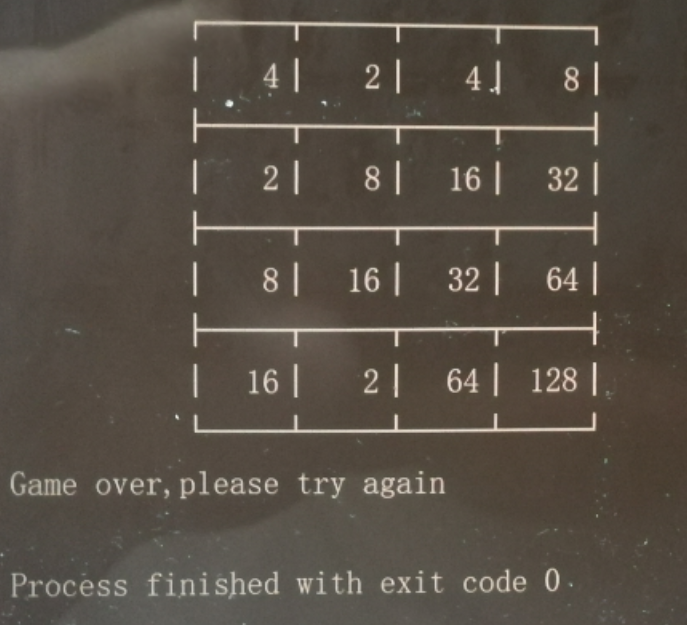


运行时刻：



我们此时可以看到分数的变化以及表格数字的变化

接下来我们看到游戏结束的情景



1. 实验收获心得

这次实验我通过4096小游戏的制作，首先是对python有了更加深刻的理解，由于是对游戏的制作，我也对写代码有了更加深厚的兴趣。与此同时，我也对用python如何进行游戏的制作有了更加深刻的体会。经过整个实验我的收获非常大。

1. 源代码
2. import random  
     
   score=int(0)  
   matrix = [[0 for i in range(4)] for i in range(4)]  
     
   #判断是否为空  
   def notzero(s):  
    return s if s != 0 else ''  
     
   #显示界面  
   def showPic():  
    print("\r\  
    ┌──┬──┬──┬──┐\n\  
    │%4s│%4s│%4s│%4s│\n\  
    ├──┬──┬──┬──┤\n\  
    │%4s│%4s│%4s│%4s│\n\  
    ├──┬──┬──┬──┤\n\  
    │%4s│%4s│%4s│%4s│\n\  
    ├──┬──┬──┬──┤\n\  
    │%4s│%4s│%4s│%4s│\n\  
    └──┴──┴──┴──┘" \  
    % (notzero(matrix[0][0]), notzero(matrix[0][1]), notzero(matrix[0][2]), notzero(matrix[0][3]), \  
    notzero(matrix[1][0]), notzero(matrix[1][1]), notzero(matrix[1][2]), notzero(matrix[1][3]), \  
    notzero(matrix[2][0]), notzero(matrix[2][1]), notzero(matrix[2][2]), notzero(matrix[2][3]), \  
    notzero(matrix[3][0]), notzero(matrix[3][1]), notzero(matrix[3][2]), notzero(matrix[3][3]),)  
    )  
     
     
   #游戏初始化  
   def initGame():  
    num=0  
    while 1:  
    if random.randrange(0, 10)>1:  
    s=2  
    else:  
    s=4  
    m = divmod(random.randrange(0, 16), 4)  
    if matrix[m[0]][m[1]]==0:  
    num+=1  
    matrix[m[0]][m[1]]=s  
    if num==2:  
    break  
    showPic()  
     
     
   #判断棋盘是否为空  
   def checkEmpty():  
    for i in range(4):  
    for j in range(4):  
    if matrix[i][j]==0:  
    return True  
    return False  
     
   #判断游戏是否结束  
   def check():  
    for i in range(4):  
    for j in range(3):  
    if matrix[i][j] == 0 or matrix[i][j] == matrix[i][j + 1] or matrix[j][i] == matrix[j + 1][i]:  
    return True  
    else:  
    return False  
     
   #增加随机数  
   def AddRandom(): # 处理完移动后添加一个新的随机数  
    while 1:  
    k = 2 if random.randrange(0, 10) > 1 else 4  
    s = divmod(random.randrange(0, 16), 4)  
    if matrix[s[0]][s[1]] == 0:  
    matrix[s[0]][s[1]] = k  
    break  
    showPic()  
     
     
   #向左移动  
   def goLeft():  
    global score  
    for i in range(4):  
    for j in range(3):  
    for k in range(1 + j, 4):  
    if matrix[i][k] > 0:  
    if matrix[i][j] == 0:  
    matrix[i][j] = matrix[i][k]  
    matrix[i][k] = 0  
    elif matrix[i][j] == matrix[i][k]:  
    matrix[i][j] \*= 2  
    score += matrix[i][j]  
    matrix[i][k] = 0  
    break  
    if checkEmpty()==True:  
    AddRandom()  
    else:  
    showPic()  
      
   #向上移动  
   def goUp():  
    global score  
    for i in range(4):  
    for j in range(3):  
    for k in range(j + 1, 4):  
    if matrix[k][i] > 0:  
    if matrix[j][i] == 0:  
    matrix[j][i] = matrix[k][i]  
    matrix[k][i] = 0  
    elif matrix[k][i] == matrix[j][i]:  
    matrix[j][i] \*= 2  
    score += matrix[j][i]  
    matrix[k][i] = 0  
    break  
    if checkEmpty()==True:  
    AddRandom()  
    else:  
    showPic()  
     
   #向下移动  
   def goDown():  
    global score  
    for i in range(4):  
    for j in range(3, 0, -1):  
    for k in range(j - 1, -1, -1):  
    if matrix[k][i] > 0:  
    if matrix[j][i] == 0:  
    matrix[j][i] = matrix[k][i]  
    matrix[k][i] = 0  
    elif matrix[j][i] == matrix[k][i]:  
    matrix[j][i] \*= 2  
    score += matrix[j][i]  
    matrix[k][i] = 0  
    break  
    if checkEmpty()==True:  
    AddRandom()  
    else:  
    showPic()  
     
   #向右移动  
   def goRight():  
    global score  
    for i in range(4):  
    for j in range(3, 0, -1):  
    for k in range(j - 1, -1, -1):  
    if matrix[i][k] > 0:  
    if matrix[i][j] == 0:  
    matrix[i][j] = matrix[i][k]  
    matrix[i][k] = 0  
    elif matrix[i][j] == matrix[i][k]:  
    matrix[i][j] \*= 2  
    score += matrix[i][j] # 将当前数作为score加上  
    matrix[i][k] = 0  
    break  
    if checkEmpty()==True:  
    AddRandom()  
    else:  
    showPic()  
      
   #判断是否获胜  
   def ifwin():  
    for i in range(4):  
    for j in range(4):  
    if matrix[i][j]==4096:  
    return True  
    return False  
      
   def main():  
    print("Welcome to the 4096 Game\n");  
    initGame();  
    while 1:  
    print("now your score is ")  
    print(score)  
    print("Please input the instruction:\n (↑:w) (↓:s) (←:a) (→:d):")  
    instr=input()  
    if instr=='w':  
    goUp()  
    elif instr=='s':  
    goDown()  
    elif instr=='a':  
    goLeft()  
    elif instr=='d':  
    goRight()  
    else:  
    pass  
    print("please follow the correct instructions:\n")  
    if not check():  
    print("Game over,please try again")  
    break  
    if ifwin():  
    print("Congratulations. You have won!!")  
    break  
     
      
      
     
     
   if \_\_name\_\_ == '\_\_main\_\_':  
    main()