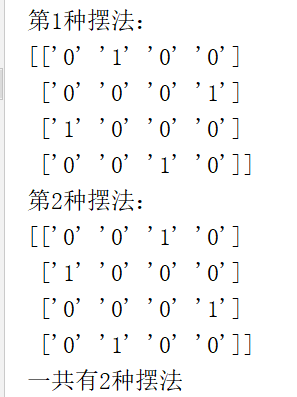
1. **四皇后问题：**

**代码：**

**import** numpy *#引用numpy包*arr2 =[[0 **for** x **in** range(4)]**for** y **in** range(4)]  
num = 0  
**def** four\_queen(arr,j=0):  
 **if** j == 4: *#如果放置4行 则完成了一种摆法* **for** x **in** range(len(arr2)):  
 **for** y **in** range(len(arr2)):  
 arr2[x][y]=0  
 **global** num *#将num定义为全局变量 才能进行自加操作* num+=1  
 print(**"第%s种摆法："** % num)  
 **for** i **in** range(4):  
 **for** x **in** range(len(arr2)):  
 **for** y **in** range(len(arr2)):  
 **if** i == x **and** arr[i]==y:  
 arr2[x][y] = **"1"** arr3 = numpy.array(arr2) *#将二维数组转化成array格式* print(arr3)  
 **return** 0  
 **for** x **in** range(len(arr)): *#对整个列进行扫描，将列标的标号赋值给数组中对应的元素* arr[j] = x  
 flag = **True  
 for** y **in** range(j):  
 **if** arr[y] == x **or** abs(arr[y]-x) == j-y: *#判断是否能放皇后*

flag = **False  
 if** flag==**True**:  
 four\_queen(arr,j+1)  
**if** \_\_name\_\_ == **'\_\_main\_\_'**:  
 four\_queen([0]\*4)  
 **if** num != 0:  
 print(**"一共有%s种摆法"** % num)  
 **else**:  
 print(**"无解"**)

**运行结果：**



1. **深度优先搜索的四个特性：**

**完整性： 深度优先如果误入了一条没有深度限制又不包含目标节点的分支，那么深度优先搜索就永远不会终止；因此是不完备的**

**最优性：深度优先如果先搜索的分支含目标节点，那么就会将它作为解返回：因此深度优先不是最优的。**

**时间复杂度：O（bm），其中m是任何节点的最大深度**

**空间复杂度：O(bm)**