**实验题目：A\*算法解决八数码问题**

**实验代码：**

**import** time **as** tm  
g\_dict\_layouts = {}  
g\_dict\_layouts\_deep = {}  
g\_dict\_layouts\_fn = {}  
*#每个位置可交换的位置集合*g\_dict\_shifts = {0:[1, 3], 1:[0, 2, 4], 2:[1, 5],  
 3:[0,4,6], 4:[1,3,5,7], 5:[2,4,8],  
 6:[3,7], 7:[4,6,8], 8:[5,7]}  
**def** swap\_chr(a, i, j, deep, destLayout):  
 **if** i > j:  
 i, j = j, i  
 *#得到ij交换后的数组* b = a[:i] + a[j] + a[i+1:j] + a[i] + a[j+1:]  
 *#存储fn,A\*算法* fn = cal\_dislocation\_sum(b, destLayout)+deep  
 **return** b, fn  
*#返回错码和正确码距离之和***def** cal\_dislocation\_sum(srcLayout,destLayout):  
 sum=0  
 a= srcLayout.index(**"0"**)  
 **for** i **in** range(0,9):  
 **if** i!=a:  
 sum=sum+abs(i-destLayout.index(srcLayout[i]))  
 **return** sum  
**def** solvePuzzle\_A(srcLayout, destLayout):  
 *#先进行判断srcLayout和destLayout逆序值是否同是奇数或偶数* src=0;dest=0  
 **for** i **in** range(1,9):  
 fist=0  
 **for** j **in** range(0,i):  
 **if** srcLayout[j]>srcLayout[i] **and** srcLayout[i]!=**'0'**:*#0是false,'0'才是数字* fist=fist+1  
 src=src+fist  
 **for** i **in** range(1,9):  
 fist=0  
 **for** j **in** range(0,i):  
 **if** destLayout[j]>destLayout[i] **and** destLayout[i]!=**'0'**:  
 fist=fist+1  
 dest=dest+fist  
 **if** (src%2)!=(dest%2):*#一个奇数一个偶数，不可达* **return** -1, **None** g\_dict\_layouts[srcLayout] = -1  
 g\_dict\_layouts\_deep[srcLayout]= 1  
 g\_dict\_layouts\_fn[srcLayout] = 1 + cal\_dislocation\_sum(srcLayout, destLayout)  
 stack\_layouts = []  
 gn=0*#深度值* stack\_layouts.append(srcLayout)*#当前状态存入列表* **while** len(stack\_layouts) > 0:  
 curLayout = min(g\_dict\_layouts\_fn, key=g\_dict\_layouts\_fn.get)  
 **del** g\_dict\_layouts\_fn[curLayout]  
 stack\_layouts.remove(curLayout)*#找到最小fn，并移除  
 # curLayout = stack\_layouts.pop()* **if** curLayout == destLayout:*#判断当前状态是否为目标状态* **break** *# 寻找0 的位置。* ind\_slide = curLayout.index(**"0"**)  
 lst\_shifts = g\_dict\_shifts[ind\_slide]*#当前可进行交换的位置集合* **for** nShift **in** lst\_shifts:  
 newLayout, fn = swap\_chr(curLayout, nShift, ind\_slide, g\_dict\_layouts\_deep[curLayout] + 1, destLayout)  
 **if** g\_dict\_layouts.get(newLayout) == **None**:*#判断交换后的状态是否已经查询过* g\_dict\_layouts\_deep[newLayout] = g\_dict\_layouts\_deep[curLayout] + 1*#存入深度* g\_dict\_layouts\_fn[newLayout] = fn*#存入fn* g\_dict\_layouts[newLayout] = curLayout*#定义前驱结点* stack\_layouts.append(newLayout)*#存入集合* lst\_steps = []  
 lst\_steps.append(curLayout)  
 **while** g\_dict\_layouts[curLayout] != -1:*#存入路径* curLayout = g\_dict\_layouts[curLayout]  
 lst\_steps.append(curLayout)  
 lst\_steps.reverse()  
 **return** 0, lst\_steps  
**if** \_\_name\_\_ == **"\_\_main\_\_"**:  
 *#测试数据* srcLayout = **"013425786"** destLayout = **"647850321"** retCode, lst\_steps = solvePuzzle\_A(srcLayout, destLayout)  
 **if** retCode != 0:  
 print(**"目标布局不可达"**)  
 **else**:  
 **for** nIndex **in** range(len(lst\_steps)):  
 print(**"step #"** + str(nIndex + 1))  
 print(lst\_steps[nIndex][:3])  
 print(lst\_steps[nIndex][3:6])  
 print(lst\_steps[nIndex][6:])

实验结果：

