



HILL CLIMBING-manhattan and misplaced tiles

HILL CLIMBING-manhattan and misplaced tiles

```

import copy,random

goal=[[1,2,3],[8,0,4],[7,6,5]]

def manhattan(s):
    pos={goal[r][c]:(r,c) for r in range(3) for c in range(3)}
    return sum(abs(i-pos[s[i][j]][0])+abs(j-pos[s[i][j]][1])
               for i in range(3) for j in range(3) if s[i][j]!=0)

def misplaced(s):
    return sum(s[i][j]!=0 and s[i][j]!=goal[i][j]
               for i in range(3) for j in range(3))

def find_blank(s):
    for i in range(3):
        for j in range(3):
            if s[i][j]==0: return i,j

def neighbors(s):
    x,y=find_blank(s)
    moves=[(1,0),(-1,0),(0,1),(0,-1)]
    result=[]
    for dx,dy in moves:
        nx,ny=x+dx,y+dy
        if 0<=nx<3 and 0<=ny<3:
            ns=copy.deepcopy(s)
            ns[x][y],ns[nx][ny]=ns[nx][ny],ns[x][y]
            result.append(ns)
    return result

def hill_climbing(start,hfunc):
    current=start
    steps=0
    while True:
        cur_h=hfunc(current)
        print(f"Step {steps}, h={cur_h}")
        for row in current: print(row)
        print("----")
        if cur_h==0: return current
        neigh=neighbors(current)
        if not neigh: return current
        nxt=min(neigh,key=hfunc)
        if hfunc(nxt)>=cur_h: return current
        current=nxt; steps+=1
    start=[[2,8,3],[1,6,4],[7,0,5]]
    hill_climbing(start,manhattan)

```

```

Step 0, h=5
[2, 8, 3]
[1, 6, 4]
[7, 0, 5]
----
Step 1, h=4
[2, 8, 3]
[1, 0, 4]
[7, 6, 5]
----
Step 2, h=3
[2, 0, 3]

```

```
[1, 8, 4]
[7, 6, 5]
----
Step 3, h=2
[0, 2, 3]
[1, 8, 4]
[7, 6, 5]
----
Step 4, h=1
[1, 2, 3]
[0, 8, 4]
[7, 6, 5]
----
Step 5, h=0
[1, 2, 3]
[8, 0, 4]
[7, 6, 5]
----
[[1, 2, 3], [8, 0, 4], [7, 6, 5]]
```