

```
1  """
2  Lab-1
3  Implement A* Search Algorithm
4  """
5
6  inf=99999
7  g=[
8      [0,3,inf,inf,4,inf,inf,inf],
9      [inf,0,4,inf,5,inf,inf,inf],
10     [inf,inf,0,4,inf,5,inf,inf],
11     [inf,inf,inf,0,inf,inf,inf,inf],
12     [inf,inf,inf,inf,0,2,inf,inf],
13     [inf,inf,inf,inf,inf,0,4,inf],
14     [inf,inf,inf,inf,inf,inf,0,3.5],
15     [inf,inf,inf,inf,inf,inf,inf,0],
16 ]
17 h=[11.5,10.1,5.8,3.4,9.2,7.1,3.5,0]
18
19 src=0
20 goal=6
21
22 class obj:
23     def __init__(self,cost,path):
24         self.cost=cost
25         self.path=path
26
27 arr=[]
28 new_item=obj(h[src],[src])
29 arr.append(new_item)
30
31 while arr:
32     cur_item=arr[0]
33     cur_node=cur_item.path[-1]
```

```
34     cur_cost=cur_item.cost
35     cur_path=cur_item.path
36     for i in range(0,len(h)):
37         if g[cur_node][i]!=inf and g[cur_node][i]!=0:
38             new_cost=cur_cost-h[cur_node]+h[i]+g[cur_node][i]
39             new_path=cur_path.copy()
40             new_path.append(i)
41             if i==goal:
42                 print(new_cost)
43                 print(new_path)
44                 new_item=obj(new_cost,new_path)
45                 arr.append(new_item)
46     arr.pop(0)
47     arr=sorted(arr,key=lambda item: item.cost)
48
49 """
50 Output:
51
52 13.5
53 [0, 4, 5, 6]
54 17.5
55 [0, 1, 4, 5, 6]
56 19.5
57 [0, 1, 2, 5, 6]
58 """
```