```
1 def astaralgo(start node, stop node):
     open set=set(start node)
 3
     closed_set=set()
    g={}
 5
     parents={}
 6
    g[start node]=0
 7
     parents[start_node]=start_node
 8
 9
     while len(open_set)>0:
10
       n=None
11
       for v in open set:
12
         if n==None or g[v] + heuristic(v)<g[n]+heuristic(n):</pre>
13
14
       if n==stop_node or Graph_nodes[n]==None:
15
         pass
16
       else:
17
         for(m,weigth) in get neighbors(n):
           if m not in open set and m not in closed set:
18
19
             open set.add(m)
20
             parents[m]=n
21
             g[m]=g[n]+weigth
22
23
           else:
             if g[m]>g[n]+weigth:
24
25
               g[m]=g[n]+weigth
26
               parents[m]=n
27
               if m in closed set:
                 closed set.remove(m)
28
29
                 open_set.add(m)
30
       if n==None:
31
         print('Path does not exist! ')
32
         return None
33
       if n==stop node:
34
         path=[]
35
36
         while parents[n]!=n:
37
           path.append(n)
           n=parents[n]
38
39
         path.append(start_node)
40
         path.reverse()
41
         print('Path found: {}'.format(path))
42
         return path
43
44
       open set.remove(n)
45
       closed set.add(n)
46
     print('Path does not exist! ')
47
     return None
48
49 def get neighbors(v):
50
     if v in Graph nodes:
```

```
51
       return Graph nodes[v]
52
     else:
53
       return None
54
55 def heuristic(n):
    h dist={
56
57
         'A':11,
         'B':6,
58
59
         'C':99,
60
         'D':1,
61
         'E':7,
         'G':0
62
63
64
     return h_dist[n]
65
66 Graph_nodes={
       'A':[('B',2),('E',3)],
67
       'B':[('C',1),('G',9)],
68
69
       'C':None,
70
       'E':[('D',6)],
71
       'D':[('G',1)]
72 }
73 astaralgo('A','G')
    Path found: ['A', 'E', 'D', 'G']
     ['A', 'E', 'D', 'G']
```

Colab paid products - Cancel contracts here

X