8. Write the python program to implement A\* algorithm.

from heapq import heappop, heappush

def astar(start, goal, graph, h):

open\_set = [(h[start], 0, start, [start])]

closed = set()

while open\_set:

f, g, node, path = heappop(open\_set)

if node == goal:

return path, g

if node in closed:

continue

closed.add(node)

for neigh, cost in graph[node]:

if neigh not in closed:

heappush(open\_set, (g+cost+h[neigh], g+cost, neigh, path+[neigh]))

# Example Graph

graph = {

'A': [('B',1), ('C',3)],

'B': [('D',3), ('E',1)],

'C': [('F',5)],

'D': [('G',2)],

'E': [('G',6)],

'F': [('G',2)],

'G': []

}

h = {'A':7,'B':6,'C':5,'D':4,'E':3,'F':2,'G':0} # heuristic

path, cost = astar('A','G',graph,h)

print("Best Path:", path)

print("Cost:", cost)

