Sarthak Kapaliya
20BCP072

A * Algorthm

Find the path from A-G for the given graph by using A* algorithm.

Code:

```
nodes = {
    "A": [["B", 10], ["C", 4]],
    "B": [["A", 10], ["C", 4], ["D", 6]],
    "C": [["A", 4], ["B", 4], ["E", 6]],
    "D": [["B", 6], ["E", 6], ["F", 4]],
    "E": [["C", 6], ["D", 6], ["G", 14]],
    "F": [["D", 4], ["G", 4]],
    "G": [],
}

hValue = {"A": 16, "B": 4, "C": 16, "D": 2, "E": 1, "F": 4, "G": 0}

start = "A"
end = "G"
visited = []
cost = {}
openL = []
closed = {}

for i in nodes.keys():
    li = []
    for j in nodes[i]:
        li.append(j[0])
    closed[i] = li

openL.append(["A", 0])
cost[openL[0][0]] = 0
lastNode = openL[-1]

while True:
    lastNode = openL[0]
    l = nodes[openL[0][0]]
    visited.append(openL[0])
    openL.pop(0)
    flag = False
    for i in l:
        if i not in visited:
            if i[0] not in cost:
                flag = True
```

Sarthak Kapaliya
20BCP072

```
                cost[i[0]] = i[1] + lastNode[1]
                openL.append([i[0], cost[i[0]]])
            else:
                if cost[i[0]] + hValue[i[0]] > i[1] + lastNode[1]:
                    flag = True
                    for j in openL:
                        if j[0] == i[0]:
                            cost[i[0]] = i[1] + lastNode[1]
                            openL[openL.index(j)] = [i[0], cost[i[0]]]

    if not flag and len(l) != 0:
        visited.pop(-1)
    openL.sort(key=lambda i: i[1])
    if openL == []:
        break


print("Path", end=': ')
for i in visited[:-1]:
    print(i[0], end="--> ")
print(visited[-1][0])
```

Output: