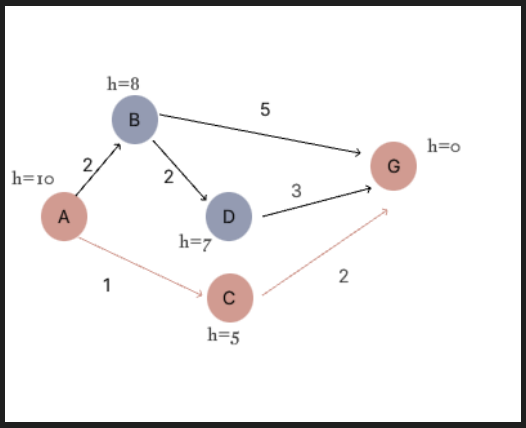
Artificial intelligence (CSC403)

Student name: Razan Abdulrahman Alrashed ID: 443500087

ASSIGNMENT 5

```python
from queue import PriorityQueue
class Graph:
    def __init__(self):
        self.graph = {}
        self.heuristics = {}
    def add_node(self, node):
        if node not in self.graph:
            self.graph[node] = []
    def add_edge(self, node1, node2, value):
        if node1 in self.graph and node2 in self.graph:
            self.graph[node1].append((node2, value))
            self.graph[node2].append((node1, value))
    def add_heuristic(self, node, heuristic):
        self.heuristics[node] = heuristic
def A_Star(graph, start, target):
    visited = set()
    queue = PriorityQueue()
    queue.put((0, start, [start]))  # (cost, node, path)
    while not queue.empty():
        cost, node, path = queue.get()  # get will remove and return from queue
        if node not in visited:
            visited.add(node)
            if node == target:
                return cost, path
            for neighbor, weight in graph.graph[node]:
                if neighbor not in visited:
                    total_cost = cost + weight + graph.heuristics.get(neighbor, 0)
                    queue.put((total_cost, neighbor, path + [neighbor]))
g = Graph()
g.add_node('A')
g.add_node('B')
g.add_node('C')
g.add_node('D')
g.add_node('G')
g.add_edge('A', 'B', 2)
g.add_edge('A', 'C', 1)
g.add_edge('B', 'D', 2)
g.add_edge('B', 'G', 5)
g.add_edge('C', 'G', 2)
g.add_edge('D', 'G', 3)
g.add_heuristic('A', 10)
g.add_heuristic('B', 8)
g.add_heuristic('C', 5)
g.add_heuristic('D', 7)
g.add_heuristic('G', 0)
print(f"PATH: {' -> '.join(path)}")
print(f"Cost from A to G : {cost}")
```




```
PATH: A -> C -> G
Cost from A to G : 8
```

output