EXPERIMENT:10 Write the python program to implement A* algorithm

PROGRAM:

```
import heapq


def a_star(graph, start, goal, h):
    open_set = []
    heapq.heappush(open_set, (h[start], start))
    came_from = {}
    g_score = {node: float('inf') for node in graph}
    g_score[start] = 0

    while open_set:
        _, current = heapq.heappop(open_set)
        if current == goal:
            path = []
            while current in came_from:
                path.append(current)
                current = came_from[current]
            path.append(start)
            return path[::-1]

        for neighbor, cost in graph[current]:
            tentative_g = g_score[current] + cost
            if tentative_g < g_score[neighbor]:
                came_from[neighbor] = current
                g_score[neighbor] = tentative_g
                f_score = tentative_g + h[neighbor]
                heapq.heappush(open_set, (f_score, neighbor))
    return None
```

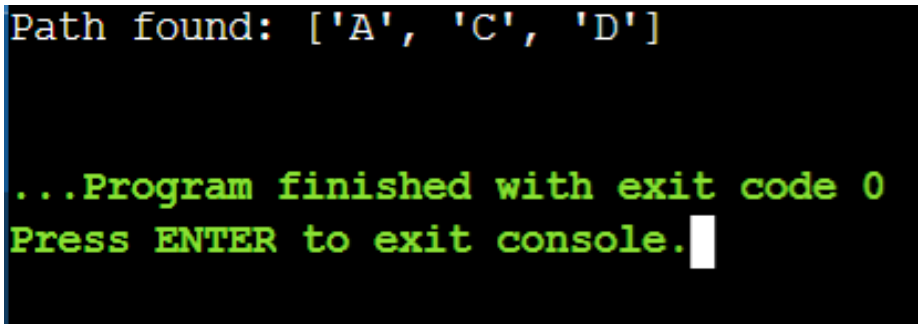
```
# Example graph as adjacency list: node -> [(neighbor, cost), ...]
graph = {
    'A': [('B', 1), ('C', 4)],
    'B': [('A', 1), ('C', 2), ('D', 5)],
    'C': [('A', 4), ('B', 2), ('D', 1)],
    'D': [('B', 5), ('C', 1)]
}

# Heuristic values for each node (straight-line distance to goal 'D')
h = {
    'A': 7,
    'B': 6,
    'C': 2,
    'D': 0
}

start = 'A'
goal = 'D'

path = a_star(graph, start, goal, h)
if path:
    print("Path found:", path)
else:
    print("No path found.")
```

OUTPUT:


Path found: ['A', 'C', 'D']
...Program finished with exit code 0
Press ENTER to exit console.