```python
from queue import PriorityQueue
def a_star(graph, heuristic, start, goal):
    open_list = PriorityQueue()
    open_list.put((0, start))
    came_from = {}
    g_score = {node: float('inf') for node in graph}
    g_score[start] = 0
    while not open_list.empty():
        _, current = open_list.get()
        if current == goal:
            path = []
            while current in came_from:
                path.append(current)
                current = came_from[current]
            path.append(start)
            path.reverse()
            print("Shortest Path:", ' -> '.join(path))
            print("Total Cost:", g_score[goal])
            return
        for neighbor, cost in graph[current].items():
            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 + heuristic[neighbor]
                open_list.put((f_score, neighbor))
    print("No path found!")
graph = {
    'A': {'B': 1, 'C': 3},
    'B': {'A': 1, 'D': 3, 'E': 5},
    'C': {'A': 3, 'F': 7},
    'D': {'B': 3, 'E': 1, 'G': 3},
    'E': {'B': 5, 'D': 1, 'G': 2},
    'F': {'C': 7, 'G': 2},
    'G': {'D': 3, 'E': 2, 'F': 2}
}
heuristic = {
    'A': 10,
    'B': 8,
    'C': 5,
    'D': 7,
    'E': 3,
    'F': 6,
    'G': 0  # Goal node
}
a_star(graph, heuristic, 'A', 'G')
```

```
Shortest Path: A -> B -> E -> G
Total Cost: 8
```