312605018_郭珈源_hw1

1. Provide the route planning from Arad to the destination station Bucharest.
   - Arad 周圍的車站，與 Bucharest 的直線距離分別是

     `[['Zerind', 374], ['Timisoara', 329], ['Sibiu', 253]]`

     所以下一站會選擇與 Bucharest 的直線距離最近的 Sibiu

   - Sibiu 周圍的車站，與 Bucharest 的直線距離分別是

     `[['Arad', 366], ['Oradea', 380], ['Fagaras', 176], ['Rimnicu Vilcea', 193]]`

     所以下一站會選擇與 Bucharest 的直線距離最近的 Fagaras

   - Fagaras 周圍的車站，與 Bucharest 的直線距離分別是

     `[['Sibiu', 253], ['Bucharest', 0]]`

     所以下一站會選擇與 Bucharest 的直線距離最近的 Bucharest，也就是終點站

2. Provide the route planning from Oradea and Mehadia to the destination station Bucharest, respectively.

   From Oradea to the destination station Bucharest:
   - Oradea 周圍的車站，與 Bucharest 的直線距離分別是

     `[['Zerind', 374], ['Sibiu', 253]]`

     所以下一站會選擇與 Bucharest 的直線距離最近的 Sibiu

   - Sibiu 周圍的車站，與 Bucharest 的直線距離分別是

     `[['Arad', 366], ['Oradea', 380], ['Fagaras', 176], ['Rimnicu Vilcea', 193]]`

     所以下一站會選擇與 Bucharest 的直線距離最近的 Fagaras

   - Fagaras 周圍的車站，與 Bucharest 的直線距離分別是

     `[['Sibiu', 253], ['Bucharest', 0]]`

     所以下一站會選擇與 Bucharest 的直線距離最近的 Bucharest，也就是終點站

From Mehadia to the destination station Bucharest:

- Mehadia 周圍的車站，與 Bucharest 的直線距離分別是

  `[['Lugoj', 244], ['Dobreta', 242]]`

  所以下一站會選擇與 Bucharest 的直線距離最近的 Dobreta

- Dobreta 周圍的車站，與 Bucharest 的直線距離分別是

  `[['Mehadia', 241], ['Craiova', 160]]`

  所以下一站會選擇與 Bucharest 的直線距離最近的 Craiova

- Craiova 周圍的車站，與 Bucharest 的直線距離分別是

  `[['Dobreta', 242], ['Pitesti', 100], ['Rimnicu Vilcea', 193]]`

  所以下一站會選擇與 Bucharest 的直線距離最近的 Pitesti

- Pitesti 周圍的車站，與 Bucharest 的直線距離分別是

  `[['Rimnicu Vilcea', 193], ['Craiova', 160], ['Bucharest', 0]]`

  所以下一站會選擇與 Bucharest 的直線距離最近的 Bucharest，也就是終點站

3. Discuss the Pros and Cons of Greedy Best-First Search.

   Pros:

   - 它是用啟發式函數來引導搜索，傾向於選擇距離目標最近的節點，讓它在某些情況下能非常快速的找到目標
   - 它較容易理解，實作上也相對簡單，只需選擇啟發函數最小的節點進行擴展，無須複雜的計算
   - 只需較少的儲存空間，像 A* 便會需要維護更多資訊，需要較多的儲存空間

   Cons:

   - 無法保證得到最短路徑，因為它忽略了全局的資訊，有時候甚至會得到非常糟糕的路徑
   - 若啟發式函數設計的不合理或不精準，很有可能影響最後做出的決定，效果會較低很多，甚至可能退化成 BFS 或 DFS 這種普通的搜尋法

4. Provide suggestions for improving the disadvantages of Greedy Best-First Search.
    - 使用更精確的啟發式函數，比如用歐基里得或曼哈頓距離，來代替簡單的直距離，或結合地形等訊息，讓啟發式函數更有意義
    - 使用多種啟發式函數，不只是距離，可以是車程時間或搭乘數量等訊息，這樣可以求出距離較近、時間較快的路線
    - 將之前走過的累積路徑納入考慮，轉化成 A* 的做法，使得 f(n) = g(n) + h(n)，並在發現目前路線比其他長時，回溯至其他較短的路線，如此便能保證求出最短路徑