Giải thích chương trình

-Chương trình được xây dựng trên đồ thị không có hướng

-Thư viện sử dụng:

+networkx

+sys

-Về các object và function liên quan đến G có thể tìm hiểu thêm về thư viện networkx:

<https://networkx.github.io/documentation/stable/>

-Bài toán xử lý thuật toán A\* dựa trên thư viện theo ý tưởng sau:

+Mỗi thành phố sẽ được xem như là 1 đỉnh (node) trên đồ thị

+Đường đi từ thành phố này đến thành phố khác được xem như là cạnh (edge) giữa 2 đỉnh

+Khoảng cách giữa 2 đỉnh được gán vào nhãn (label) “distance” của cạnh

+Khoảng cách từ đỉnh xuất phát (root) đến đỉnh hiện tại được gán vào nhãn “cost” của đỉnh

+Khoảng cách đường chim bay từ đỉnh đang xét đến đỉnh cần tìm (goal) được gán vào nhãn “skyway” của đỉnh

+Tổng của “cost” và “skyway” được gán vào nhãn “total” của đỉnh

+Đỉnh trước của đỉnh đang xét được gán vào nhãn “pre” của đỉnh

+Còn lại là làm theo ý tưởng của thuật toán A\*

-nx.Graph() là đồ thị dựa trên thư viện networkx

-Xử lý cho việc nhập dữ liệu:

+while True: lệnh vô thời hạn

+sys.exit(): lệnh dừng và thoát chương trình dựa trên thư viện sys

+str.strip(): loại bỏ khoảng trắng thừa ở đầu và cuối chuỗi ký tự

+G.add\_edge: thêm 2 đỉnh và cạnh giữa 2 đỉnh, tạo và gắn thêm label cho cạnh nếu có (distance)

+G.nodes[Node][label]: nhãn của đỉnh trên đồ thị

+G[Node1][Node2][label]: nhãn của cạnh trên đồ thị

-Hàm MinF dùng để tìm đỉnh có nhãn total nhỏ nhất từ list frontier

-Hàm FindPath dùng để truy lại đường đi từ đỉnh xuất phát đến đỉnh cần tìm sau khi thực hiện thuật toán A\* theo phương pháp đệ quy

-Hàm A\_Star thực hiện theo thuật toán A\* và trả về đường đi nếu có, trả về thông báo nếu không