Giải thích chương trình

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

-Thư viện sử dụng:

+networkx

+matplotlib

+sys

-Hàm Visit và IDDFS\_Visit là 2 hàm xử lý trường hợp duyệt đồ thị theo thuật toán Iterative Deepening Depth First Search (IDDFS)

+IDDFS\_Visit: gọi hàm Visit, trả về cách duyệt đồ thị

+Visit:

.Sử dụng phương pháp đệ quy

.Mỗi lần đệ quy sẽ giảm max\_depth đi 1, thêm đỉnh hiện tại vào path

.Dừng khi thỏa điều kiện của max\_depth

-Hàm Search và IDDFS\_Search là 2 hàm xử lý trường hợp tìm kiếm trên đồ thị theo thuật toán Iterative Deepening Depth First Search (IDDFS)

+IDDFS\_Search: gọi hàm Search, trả về đường đi từ root đến goal nếu có, hoặc in ra thông báo

nếu không có

+Search:

.Sử dụng phương pháp đệ quy

.Mỗi lần đệ quy sẽ giảm max\_depth đi 1, các new\_path được tạo để lưu lại đường đi

trước đó ([hcm->qn]->hn/[hcm->qn]->nb)

.Dừng khi tìm thấy đỉnh cần tìm (goal) hoặc khi max\_depth thỏa điều kiện

-Điểm chung:

+ Mảng closed có chức năng lưu lại những đỉnh đã duyệt qua để tránh việc trùng lặp

+ Sử dụng đệ quy để hỗ trợ cho việc duyệt qua các phần tử theo yêu cầu của thuật toán

-Điểm khác:

+Với visit thì đệ quy theo đỉnh, vì chỉ cần lưu lại đỉnh để thể hiện cách duyệt của thuật toán

+Với search thì đệ quy theo danh sách, danh sách lưu lại đường đi từ đỉnh bắt đầu (root) trước

khi đệ quy, cần lưu đường đi vì yêu cầu bài toán

-Hàm Grapth\_Visit dùng để xử lý trường hợp duyệt đồ thị theo 2 thuật toán Breath-First Search (BFS) và Depth-First Search

+Với DFS: sử dụng duyệt mảng theo stack (First In, Last Out)

+Với BFS: sử dụng duyệt mảng theo queue (First In, First Out)

-Điểm chung:

+ Mảng closed có chức năng lưu lại những đỉnh đã duyệt qua để tránh việc trùng lặp

+Mảng open để xử lý thứ tự duyệt đỉnh theo yêu cầu của từng thuật toán

+Trả về cách duyệt theo từng thuật toán (path)

-Điểm khác:

+DFS: vì duyệt theo 1 nhánh cho tới hết nên những đỉnh vào sau được duyệt trước ->Stack

+BFS: duyệt từng nhánh đều nhau theo cấp đệ nên đỉnh vào trước được duyệt trước ->Queue

-Hàm Graph\_Search dùng để xử lý trường hợp tìm kiếm trên đồ thị theo 2 thuật toán Breath-First Search (BFS) và Depth-First Search

(Tương tự như trên)

-Bổ sung:

+Thêm danh sách queue để lưu lại các đường đi từ đỉnh xuất phát (path)

+Với mỗi lần duyệt thì lấy đỉnh cuối cùng của mỗi đường đi

+new\_path dùng để tạo mới đường đi theo các hướng khác nhau

+Trả về đường đi từ đỉnh xuất phát (root) đến đỉnh cần tìm (goal) nếu có, hoặc in ra thông báo

nếu không có

-Phần chính (main): phần code chính, (phần trên chỉ là các hàm)

+Lưu đồ thị theo Grapth (thuộc thư viện networkx)

+str.strip(): loại bỏ các dấu khoảng trắng ở đầu và cuối chuỗi

+try {} expect: xử lý lỗi của khố lệnh trong try{} mà không ngừng chương trình

+sys.exit(): thoát chương trình () (thuộc thư viên sys)

+while True: nhập dữ liệu thỏa thích, chỉ thoát có lệnh break (hoặc 1 số lệnh khác)

+Vẽ hình (sử dụng thư viện matplotlib)

.pos là cách bố trí các đỉnh: ở đây chọn kiểu spring

.nx.draw (..): vẽ hình với các chỉ số tự định

.plt.show(): hiện thị đồ thị đã vẽ