**Báo cáo: Giải bài toán N-Puzzle với thuật toán DFS**

**1. Giới thiệu bài toán**

N-Puzzle là một trò chơi trượt ô số gồm N×N ô vuông, trong đó có N²-1 ô được đánh số từ 1 đến N²-1 và 1 ô trống. Mục tiêu là di chuyển các ô từ một cấu hình ban đầu về trạng thái đích.

**2. Định nghĩa bài toán**

**2.1. Trạng thái bài toán**

* Được biểu diễn bởi lớp [NPuzzleState](vscode-file://vscode-app/c:/Users/GMT/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html) chứa:
  + [board](vscode-file://vscode-app/c:/Users/GMT/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html): List các số biểu diễn trạng thái bảng
  + [blank\_pos](vscode-file://vscode-app/c:/Users/GMT/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html): Vị trí ô trống hiện tại

**2.2. Trạng thái khởi đầu và kết thúc**

* **Trạng thái khởi đầu**: Cấu hình ngẫu nhiên của các số từ 0 đến N²-1
* **Trạng thái đích**: Các số được sắp xếp theo thứ tự từ 1 đến N²-1, số 0 (ô trống) ở cuối

**3. Luật di chuyển**

Được định nghĩa trong hàm [get\_valid\_moves](vscode-file://vscode-app/c:/Users/GMT/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html):

* Ô trống có thể di chuyển theo 4 hướng:
  + Lên trên (nếu không ở hàng đầu)
  + Xuống dưới (nếu không ở hàng cuối)
  + Sang trái (nếu không ở cột đầu)
  + Sang phải (nếu không ở cột cuối)

**4. Thuật toán giải**

**4.1. Giới thiệu**

Sử dụng thuật toán DFS (Depth-First Search) với giới hạn độ sâu để tìm đường đi từ trạng thái đầu đến trạng thái đích.

**4.2. Cách triển khai**

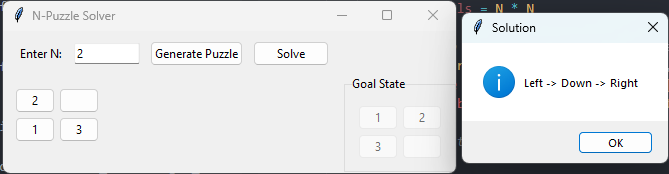
* Sử dụng stack để lưu trữ các trạng thái cần khám phá
* Sử dụng tập hợp visited để tránh lặp lại trạng thái
* Giới hạn độ sâu tìm kiếm để tránh tràn stack

**5. Tóm tắt mã nguồn**

* [n\_puzzle.py](vscode-file://vscode-app/c:/Users/GMT/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html): Chứa logic chính của trò chơi
  + [NPuzzleState](vscode-file://vscode-app/c:/Users/GMT/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html): Lớp biểu diễn trạng thái
  + [get\_valid\_moves](vscode-file://vscode-app/c:/Users/GMT/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html): Hàm lấy các nước đi hợp lệ
  + [apply\_move](vscode-file://vscode-app/c:/Users/GMT/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html): Hàm thực hiện di chuyển
  + [dfs](vscode-file://vscode-app/c:/Users/GMT/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html): Thuật toán tìm lời giải
* [gui.py](vscode-file://vscode-app/c:/Users/GMT/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html): Giao diện người dùng
  + [NPuzzleGUI](vscode-file://vscode-app/c:/Users/GMT/AppData/Local/Programs/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-sandbox/workbench/workbench.html): Lớp xây dựng giao diện
  + Các chức năng: tạo puzzle, giải và hiển thị animation

**6. Kết quả và kiểm thử**

* Chương trình có thể:
  + Tạo bảng N×N với N tùy chọn
  + Tìm lời giải tự động
  + Hiển thị quá trình giải từng bước
  + Xử lý các trường hợp lỗi và hiển thị thông báo phù hợp



A screenshot of a chat

Description automatically generated