TRƯỜNG ĐẠI HỌC THỦ DẦU MỘT
VIỆN KỸ THUẬT CÔNG NGHỆ

BÁO CÁO TIỂU LUẬN MÔN HỌC
TRÍ TUỆ NHÂN TẠO

ĐỀ TÀI: XÂY DỰNG GAME A MAZE SỬ DỤNG THUẬT TOÁN DIJKSTRA

GVHD: ThS.Hoàng Mạnh Hà

SVTH: Nguyễn Kim Hoàng

MSSV: 2124802010093

LỚP: D21CNTT02

BÌNH DƯƠNG - 03/2024

TRƯỜNG ĐẠI HỌC THỦ DẦU MỘT
VIỆN KỸ THUẬT CÔNG NGHỆ

BÁO CÁO TIỂU LUẬN MÔN HỌC
TRÍ TUỆ NHÂN TẠO

ĐỀ TÀI: XÂY DỰNG GAME A MAZE SỬ DỤNG THUẬT TOÁN DIJKSTRA

GVHD: ThS.Hoàng Mạnh Hà

SVTH: Nguyễn Kim Hoàng

MSSV: 2124802010093

LỚP: D21CNTT02

BÌNH DƯƠNG - 03/2024

LỜI CẢM ƠN

- Em xin gửi lời cảm ơn chân thành đến Thầy Hoàng Mạnh Hà, người đã đóng góp và hỗ trợ cho dự án này trở thành hiện thực.
- Đặc biệt, Em muốn gửi lời tri ân đến các nhà phát triển của thuật toán Dijkstra và các nguồn mở phổ biến khác trong lĩnh vực trí tuệ nhân tạo và khoa học máy. Sự đóng góp của họ đã tạo ra cơ sở vững chắc cho sự phát triển của dự án của em.
- Em cũng muốn bày tỏ lòng biết ơn đặc biệt đến các Thầy, người đã dày công nghiên cứu, phát triển và thử nghiệm ứng dụng này. Sự cống hiến và nỗ lực của thầy đã giúp em hoàn thành dự án một cách thành công.

Một lần nữa, em xin chân thành cảm ơn Thầy Hoàng Mạnh Hà, người đã đóng góp vào thành công của dự án này.

MỤC LỤC

DANH MỤC HÌNH

MỞ ĐẦU

- Trong thế giới hiện đại, các vấn đề liên quan đến tối ưu hóa đường đi là rất phổ biến và quan trọng. Từ việc điều hướng xe cộ trong giao thông đô thị đến tìm đường đi ngắn nhất cho robot di động, việc tìm ra giải pháp hiệu quả có thể tiết kiệm thời gian và tài nguyên là một thách thức không nhỏ. Trong lĩnh vực trí tuệ nhân tạo và khoa học máy, thuật toán Dijkstra đã trở thành một công cụ quan trọng để giải quyết bài toán tìm đường đi ngắn nhất trong đồ thị.
- Trong phạm vi đề tài này, em muốn giới thiệu về một ứng dụng thực tế sử dụng thuật toán Dijkstra để giải quyết bài toán tìm đường trong mê cung. Đây không chỉ là một ứng dụng giải trí mà còn là một cách tuyệt vời để trực quan hóa cách hoạt động của thuật toán Dijkstra trong thực tế.
- Trong phần tiếp theo của bài báo cáo này, em sẽ đi sâu vào mô tả về thuật toán Dijkstra, cách thức hoạt động của nó và cách áp dụng vào việc tìm đường trong mê cung trong ứng dụng của chúng tôi.

CHƯƠNG 1. KHẢO SÁT VÀ PHÂN TÍCH BÀI TOÁN

1.1. Thuật toán Dijkstra

1.1.1. Giới thiệu thuật toán

- Thuật toán Dijkstra, được phát triển bởi nhà toán học Edsger W. Dijkstra vào năm 1956, là một trong những công cụ quan trọng và phổ biến nhất trong lĩnh vực tối ưu hóa và đồ thị. Thuật toán này được sử dụng để tìm đường đi ngắn nhất giữa hai điểm trong một đồ thị có trọng số dương. Bài viết này sẽ trình bày về cách hoạt động, ưu điểm và ứng dụng của thuật toán Dijkstra.
- Thuật toán Dijkstra hoạt động bằng cách xây dựng một đồ thị từ điểm xuất phát đến tất cả các điểm còn lại trong đồ thị. Ban đầu, khoảng cách từ điểm xuất phát đến chính nó được gán là 0, và các khoảng cách đến các điểm khác được gán là vô cùng. Sau đó, thuật toán duyệt qua tất cả các đỉnh của đồ thị và cập nhật khoảng cách ngắn nhất từ điểm xuất phát đến các đỉnh kề cạnh. Quá trình này được lặp lại cho đến khi tất cả các đỉnh đã được duyệt qua.
- Thuật toán Dijkstra có nhiều ưu điểm, bao gồm: Tính toán đường đi ngắn nhất giữa hai điểm nhanh chóng và hiệu quả, Đảm bảo tìm ra đường đi ngắn nhất trong đồ thị có trọng số dương., Dễ hiểu và triển khai, phù hợp cho nhiều ứng dụng thực tế.
- Thuật toán Dijkstra được sử dụng rộng rãi trong nhiều lĩnh vực, bao gồm: Hệ thống điều hướng GPS: Tìm đường đi ngắn nhất từ một địa điểm đến một địa chỉ khác trên bản đồ, Quản lý tuyến đường giao thông: Tối ưu hóa tuyến đường cho xe cứu hỏa, xe cấp cứu và các phương tiện khẩn cấp khác, Mạng lưới và hệ thống viễn thông: Tìm đường đi ngắn nhất giữa các điểm truy cập trong một mạng lưới.

1.1.2. Mô tả thuật toán Dijkstra

- Thuật toán Dijkstra hoạt động bằng cách tìm đường đi ngắn nhất từ một điểm xuất phát đến tất cả các điểm khác trong đồ thị có trọng số không âm. Dưới đây là mô tả cách thức hoạt động của thuật toán:

  - Khởi Tạo: Bắt đầu với một đỉnh xuất phát và gán giá trị khoảng cách là 0 cho nó, còn đối với tất cả các đỉnh khác, gán giá trị khoảng cách là vô cùng (infinitive).

- Chọn Đỉnh Gần Nhất: Duyệt qua tất cả các đỉnh của đồ thị và chọn đỉnh gần nhất chưa được duyệt (có khoảng cách ngắn nhất) làm đỉnh hiện tại.
- Cập Nhật Khoảng Cách: Cập nhật khoảng cách từ đỉnh hiện tại đến các đỉnh kề cạnh chưa được duyệt thông qua đỉnh hiện tại. Nếu khoảng cách mới nhỏ hơn khoảng cách hiện tại, cập nhật khoảng cách.
- Đánh Dấu Đỉnh: Đánh dấu đỉnh hiện tại là đã duyệt.
- Lặp Lại: Lặp lại quá trình này cho đến khi tất cả các đỉnh đã được duyệt qua.
- Kết Quả: Khi tất cả các đỉnh đã được duyệt qua, thuật toán trả về một tập hợp các đỉnh và khoảng cách tương ứng từ điểm xuất phát đến mỗi đỉnh trong tập hợp đó.

- Thuật toán Dijkstra đảm bảo tìm ra đường đi ngắn nhất từ điểm xuất phát đến tất cả các điểm khác trong đồ thị nếu trọng số trên các cạnh là không âm. Tuy nhiên, nếu có cạnh âm, thuật toán sẽ không hoạt động đúng. Độ phức tạp thời gian của thuật toán Dijkstra là O(V^2), trong đó V là số lượng đỉnh trong đồ thị.

1.1.3. Mô tả Ma trận kề

- Ma trận kề là một cách để biểu diễn đồ thị trong lập trình máy tính. Đồ thị được biểu diễn dưới dạng ma trận hai chiều, trong đó hàng và cột của ma trận tương ứng với các đỉnh của đồ thị. Mỗi phần tử trong ma trận chỉ ra mối quan hệ giữa hai đỉnh.
- Cụ thể, mỗi phần tử A[i][j] ủa ma trận kề là một số nguyên, thường là 0 hoặc 1, hoặc một giá trị biểu diễn trọng số của cạnh nếu đồ thị có trọng số. Giá trị của A[i][j] biểu thị có tồn tại một cạnh nối từ đỉnh i đến đỉnh j hay không. Nếu A[i][j] bằng 1, có cạnh nối từ đỉnh i đến đỉnh j; ngược lại, nếu A[i][j] bằng 0, không có cạnh nối.
- Ma trận kề có thể được sử dụng để thực hiện nhiều thao tác trên đồ thị như tìm kiếm đường đi ngắn nhất, kiểm tra sự kết nối giữa các đỉnh, hoặc tính toán các thuật toán tối ưu khác. Đối với đồ thị vô hướng, ma trận kề là một ma trận đối xứng (symmetric matrix), nghĩa là A[i][j]=A[j][i] . Đối với đồ thị có hướng, ma trận kề không nhất thiết phải đối xứng.

CHƯƠNG 2. GIỚI THIỆU CÔNG NGHỆ SỬ DỤNG

2.1. Công nghệ sử dụng

2.1.1. Giới thiệu về Python

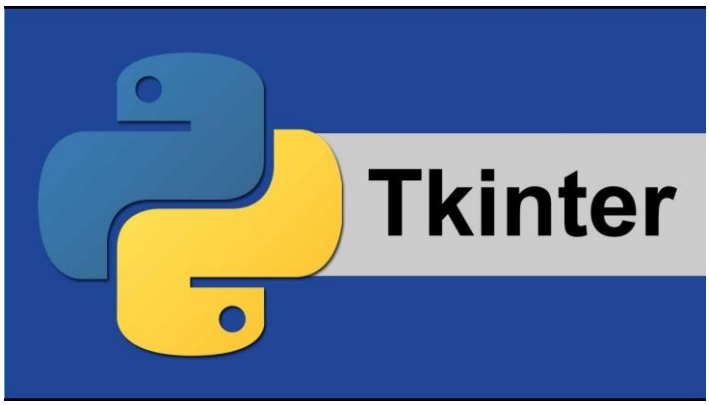
Hình 1 : Logo Python

- Python là một ngôn ngữ lập trình bậc cao cho các mục đích lập trình đa năng, do Guido van Rossum tạo ra và lần đầu ra mắt vào năm 1991. Python được thiết kế với ưu điểm mạnh là dễ đọc, dễ học và dễ nhớ. Python là ngôn ngữ có hình thức rất sáng sủa, cấu trúc rõ ràng, thuận tiện cho người mới học lập trình và là ngôn ngữ lập trình dễ học; được dùng rộng rãi trong phát triển trí tuệ nhân tạo. Cấu trúc của Python còn cho phép người sử dụng viết mã lệnh với số lần gõ phím tối thiểu.Vào tháng 7 năm 2018, van Rossum đã từ chức lãnh đạo trong cộng đồng ngôn ngữ Python sau 30 năm làm việc.
- Python hoàn toàn tạo kiểu động và dùng cơ chế cấp phát bộ nhớ tự động; do vậy nó tương tự như Perl, Ruby, Scheme, Smalltalk, và Tcl. Python được phát triển trong một dự án mã mở, do tổ chức phi lợi nhuận Python Software Foundation quản lý.
- Ban đầu, Python được phát triển để chạy trên nền Unix. Nhưng rồi theo thời gian, Python dần mở rộng sang mọi hệ điều hành từ MS-DOS đến Mac OS, OS/2, Windows, Linux và các hệ điều hành khác thuộc họ Unix. Mặc dù sự phát triển của Python có sự đóng góp của rất nhiều cá nhân, nhưng Guido van Rossum hiện nay vẫn là tác giả chủ yếu của Python. Ông giữ vai trò chủ chốt trong việc quyết định hướng phát triển của Python.

- Python luôn được xếp hạng vào những ngôn ngữ lập trình phổ biến nhất.
- Python là một ngôn ngữ lập trình đa mẫu hình, lập trình hướng đối tượng và lập trình cấu trúc được hỗ trợ hoàn toàn, và nhiều tính năng của nó cũng hỗ trợ lập trình hàm và lập trình hướng khía cạnh (bao gồm siêu lập trình và siêu đối tượng (phương thức thần kỳ)).Các mẫu hình khác cũng được hỗ trợ thông qua các phần mở rộng, bao gồm thiết kế theo hợp đồng và lập trình logic.
- Python đã trở thành một yếu tố chính trong khoa học dữ liệu, cho phép các nhà phân tích dữ liệu và các chuyên gia khác sử dụng ngôn ngữ này để thực hiện các phép tính thống kê phức tạp, tạo trực quan hóa dữ liệu, xây dựng thuật toán học máy, thao tác và phân tích dữ liệu cũng như hoàn thành các nhiệm vụ khác liên quan đến dữ liệu.
- Python có thể xây dựng nhiều dạng trực quan hóa dữ liệu khác nhau, chẳng hạn như biểu đồ đường và thanh, biểu đồ hình tròn, biểu đồ 3D. Python cũng có một số thư viện cho phép các lập trình viên viết chương trình để phân tích dữ liệu và học máy nhanh hơn và hiệu quả hơn, như TensorFlow và Keras.
- Python đang trở nên phổ biến trong cộng đồng lập trình nhờ có các đặc tính sau:
    - Ngôn ngữ thông dịch: Python được xử lý trong thời gian chạy bởi Trình thông dịch Python.
    - Ngôn ngữ hướng đối tượng: Nó hỗ trợ các tính năng và kỹ thuật lập trình hướng đối tượng.
    - Ngôn ngữ lập trình tương tác: Người dùng có thể tương tác trực tiếp với trình thông dịch python để viết chương trình.
    - Ngôn ngữ dễ học: Python rất dễ học, đặc biệt là cho người mới bắt đầu.
    - Cú pháp đơn giản: Việc hình thành cú pháp Python rất đơn giản và dễ hiểu, điều này cũng làm cho nó trở nên phổ biến.
    - Dễ đọc: Mã nguồn Python được xác định rõ ràng và có thể nhìn thấy bằng mắt.
    - Di động: Mã Python có thể chạy trên nhiều nền tảng phần cứng có cùng giao diện.

- Có thể mở rộng: Người dùng có thể thêm các mô-đun cấp thấp vào trình thông dịch Python.
- Có thể cải tiến: Python cung cấp một cấu trúc cải tiến để hỗ trợ các chương trình lớn sau đó là shell-script.

2.1.2. Giới thiệu về thư viện Tkinter

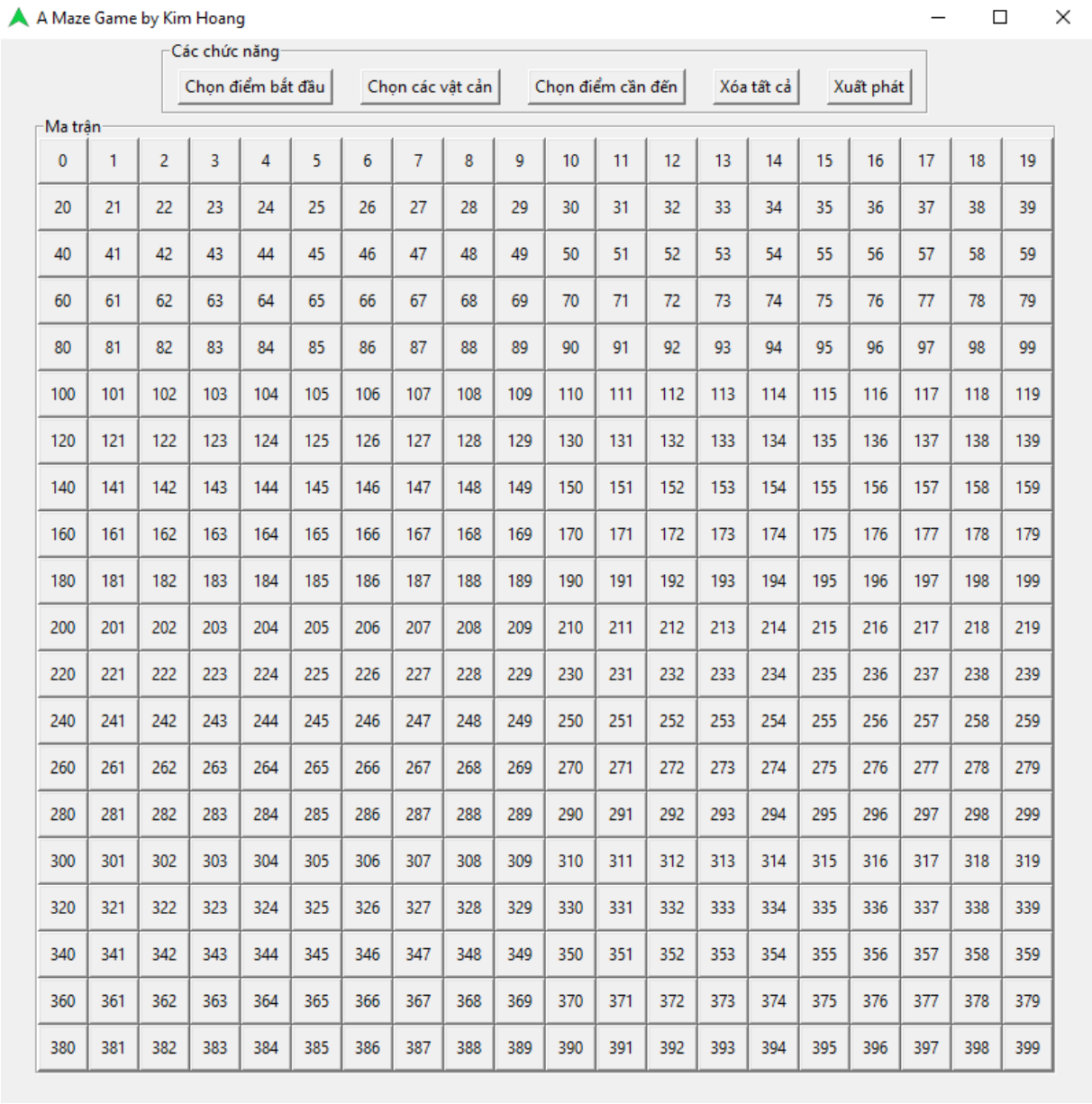


Hình 2 : Thư viện Tkinter

- Tkinter là một thư viện trong ngôn ngữ lập trình Python được sử dụng để tạo giao diện đồ họa người dùng (GUI). "Tkinter" là viết tắt của "Tk interface,", một toolkit đồ họa cung cấp các công cụ để phát triển giao diện người dùng.
- Tkinter là một phần của thư viện tiêu chuẩn của Python và đã được tích hợp sẵn trong hầu hết các cài đặt Python. Điều này giúp cho Tkinter trở thành một lựa chọn phổ biến cho việc phát triển ứng dụng với giao diện đồ họa đơn giản trong Python.
- Một số đặc điểm của Tkinter bao gồm khả năng tạo các thành phần giao diện như cửa sổ, nút, ô văn bản, và các widget khác để tương tác với người dùng. Tkinter cung cấp cả các sự kiện và phương thức để xử lý tương tác người dùng và thay đổi trạng thái của ứng dụng.

CHƯƠNG 3. THIẾT KẾ CHƯƠNG TRÌNH

3.1. Giao diện bắt đầu trò chơi



Hình 3 : Giao diện bắt đầu trò chơi

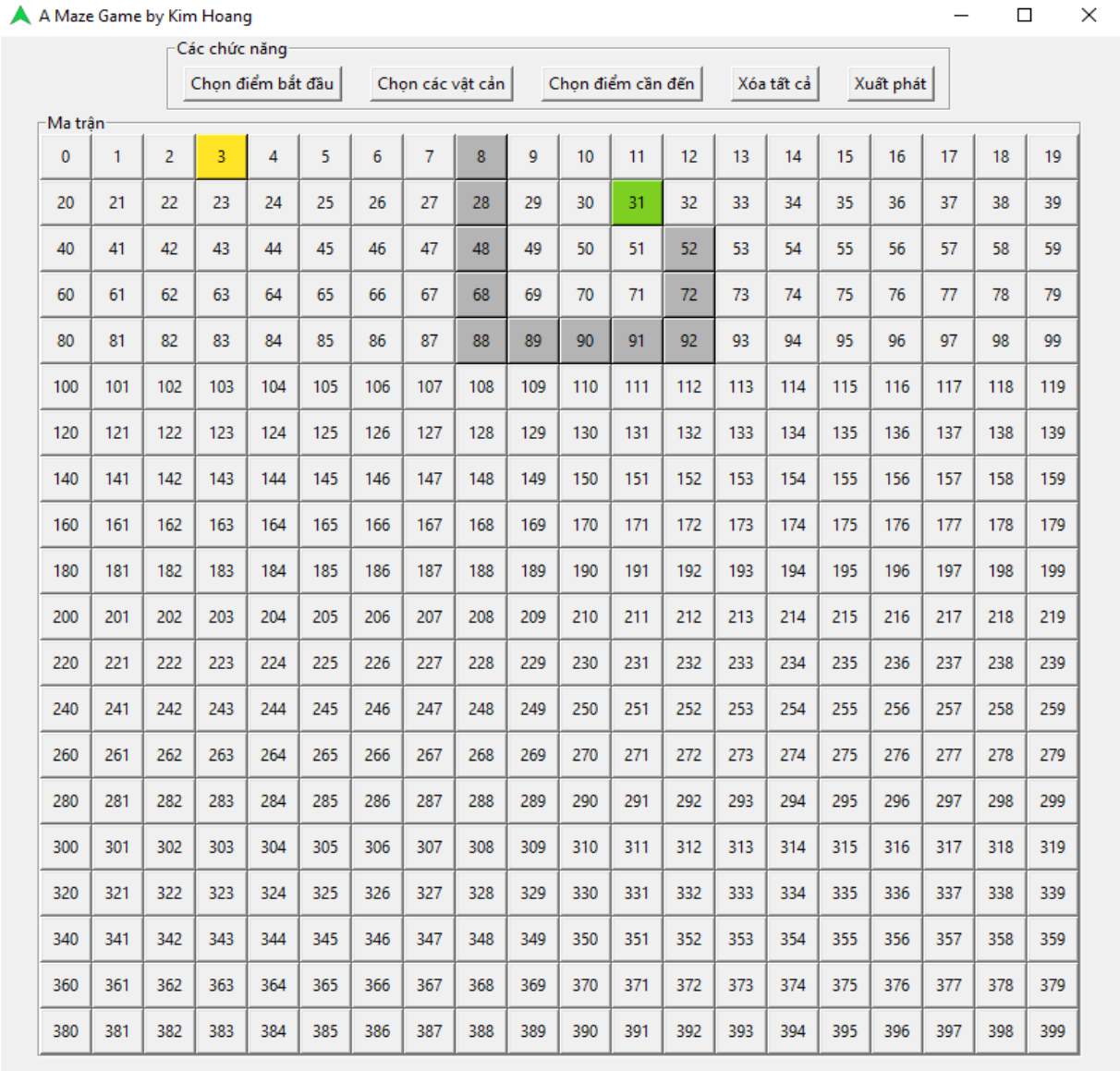
- Giao cho phép người dùng lựa chọn điểm xuất phát, đích đến và các vật cản để tạo ra một mê cung độc đáo.

- Người chơi có thể tương tác với giao diện bằng cách chọn các chức năng từ menu, bao gồm chọn điểm bắt đầu, chọn các vật cản, chọn điểm đích và xóa tất cả. Sau khi thiết lập mê cung, người dùng có thể nhấn nút "Xuất phát" để ứng dụng sử dụng thuật toán Dijkstra để tìm đường từ điểm xuất phát đến điểm đích.

- Thuật toán Dijkstra được sử dụng để tìm đường đi ngắn nhất từ một điểm đến tất cả các điểm còn lại trong mê cung. Nó hoạt động bằng cách duyệt các đỉnh theo từng bước và cập nhật khoảng cách ngắn nhất từ điểm xuất phát đến các đỉnh khác trong đồ thị.

- Kết quả sau khi tìm đường sẽ được hiển thị trên giao diện người dùng, giúp người chơi dễ dàng nhận biết đường đi và vật cản trên mê cung. Điều này giúp tạo ra một trải nghiệm trò chơi thú vị và hấp dẫn, đồng thời giúp người chơi hiểu và áp dụng thuật toán Dijkstra trong thực tế.

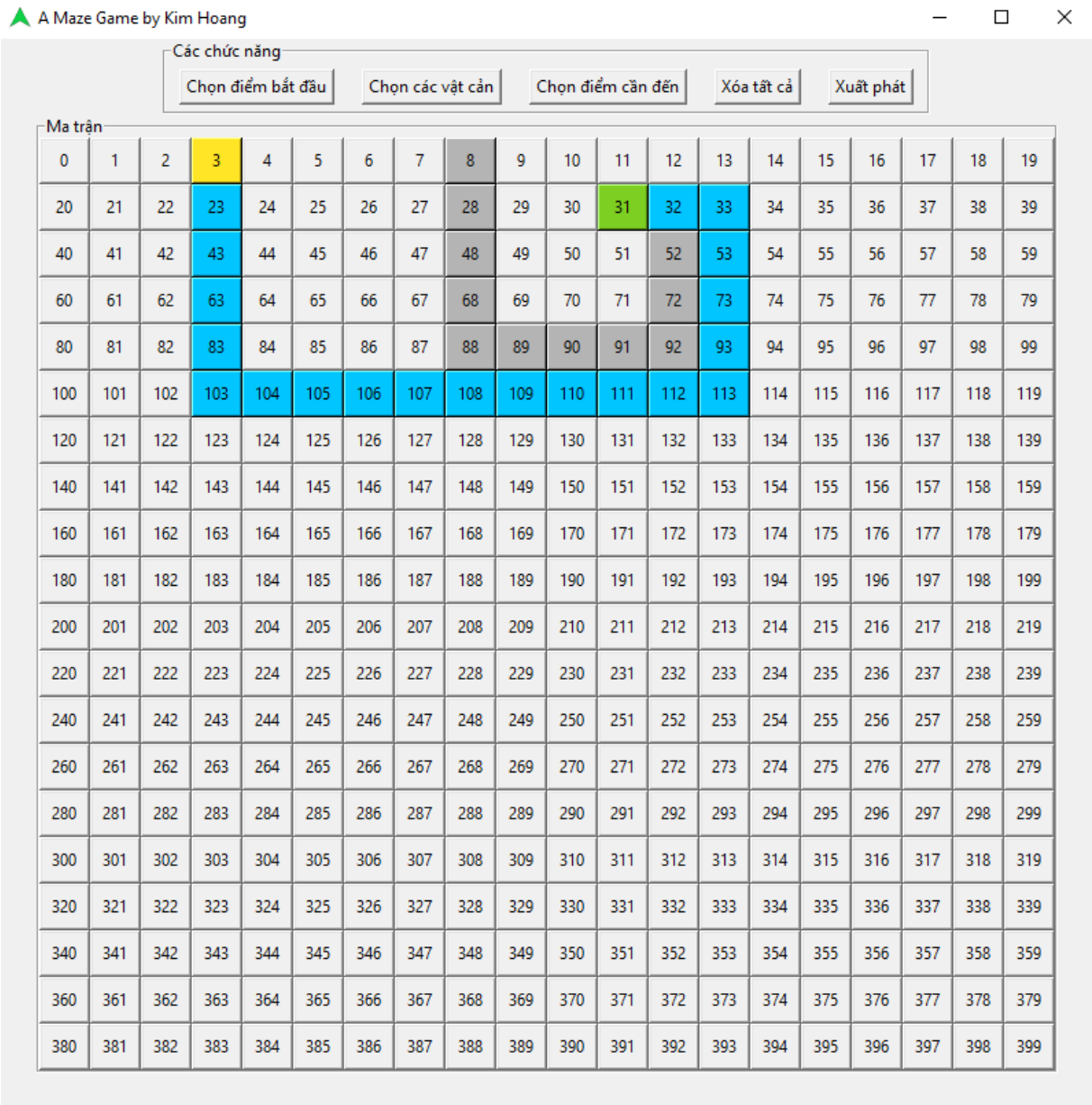
3.2. Giao diện Khi chọn điểm bắt đầu, điểm kết thúc và vật cản



Hình 4 : Giao diện Khi chọn điểm bắt đầu, điểm kết thúc và vật cản

- Ở giao diện này, khi người dùng chọn điểm bắt đầu(màu vàng), điểm kết thúc (màu xanh lá) và vật cản (màu xám)trên giao diện của trò chơi, họ đang xác định các yếu tố quyết định trong việc tạo ra một mê cung độc đáo và xác định đường đi trong đó.
- Khi chọn điểm bắt đầu, người chơi đặt nền móng cho hành trình sắp tới. Điểm này sẽ là điểm xuất phát của đường đi và một phần quan trọng trong việc xác định đường đi ngắn nhất trong mê cung.
- Sau đó, khi chọn điểm kết thúc, người chơi định rõ mục tiêu của họ trong mê cung. Điểm này là điểm cuối cùng mà họ muốn đến và thuật toán sẽ tìm cách tạo ra đường đi ngắn nhất từ điểm bắt đầu đến điểm kết thúc này.
- Ngoài ra, khi chọn các vật cản, người chơi đặt ra các thách thức và khó khăn trong việc điều hướng qua mê cung. Các vật cản này sẽ tạo ra những rào cản và hạn chế đường đi có thể đi qua, yêu cầu thuật toán phải tìm ra đường đi thông qua những khu vực không bị chặn.

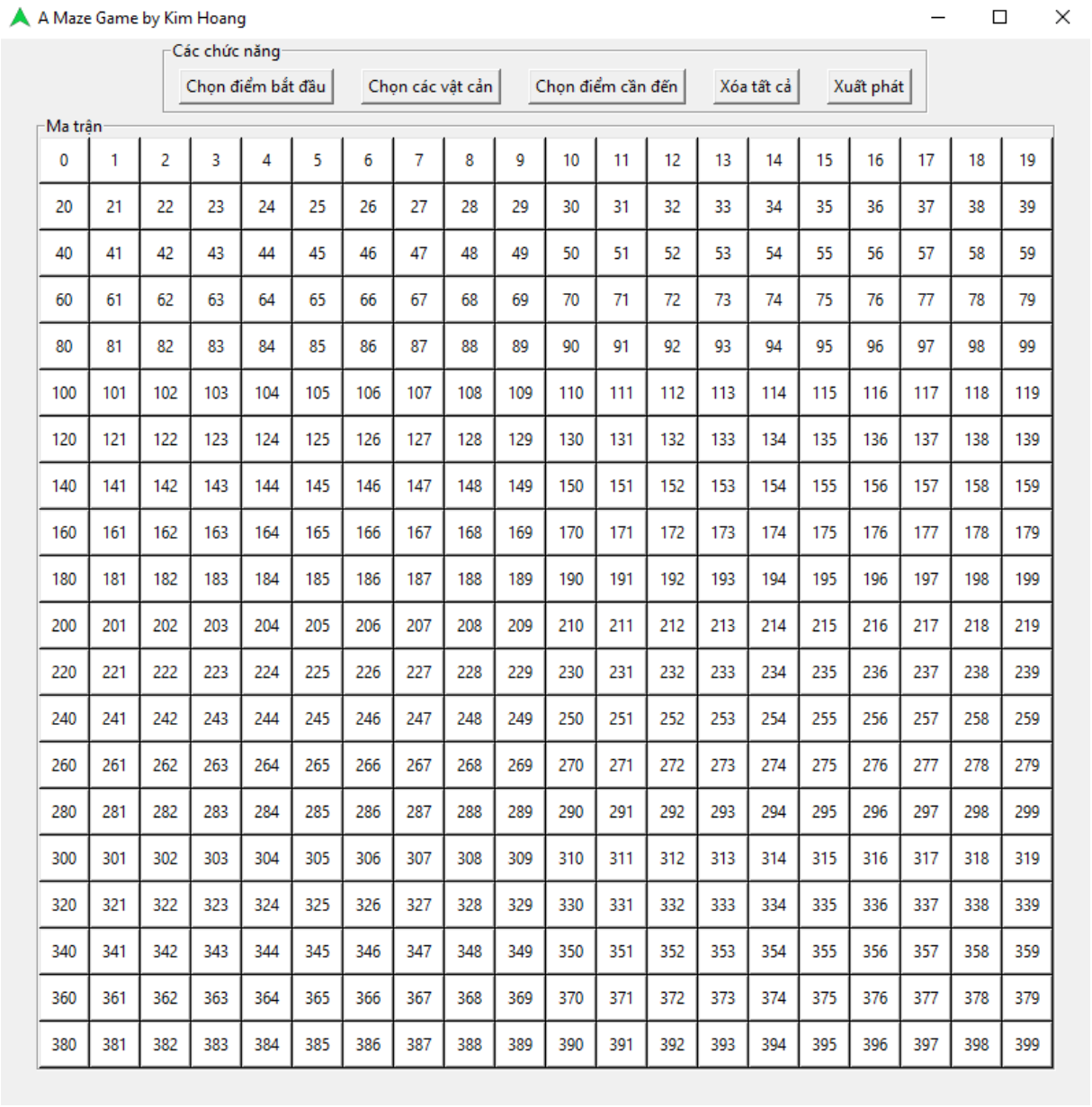
3.3. Giao diện Người dùng xuất phát



Hình 5 : Giao diện Người dùng xuất phát

- Ở giao diện này, khi người dùng bấm nút xuất phát, người dùng chờ đợi với mong đợi và hi vọng rằng họ đã đưa ra những quyết định đúng đắn và sẽ tiến lên được trong cuộc hành trình. Họ sẽ chứng kiến sự hoạt động của thuật toán Dijkstra và quá trình tìm đường đi tối ưu trong mê cung.Đường đi sẽ né các vật cản được đặt ra và tìm đường đi ngắn nhất từ điểm đầu (màu vàng) đến điểm điểm đích (màu xanh lá).

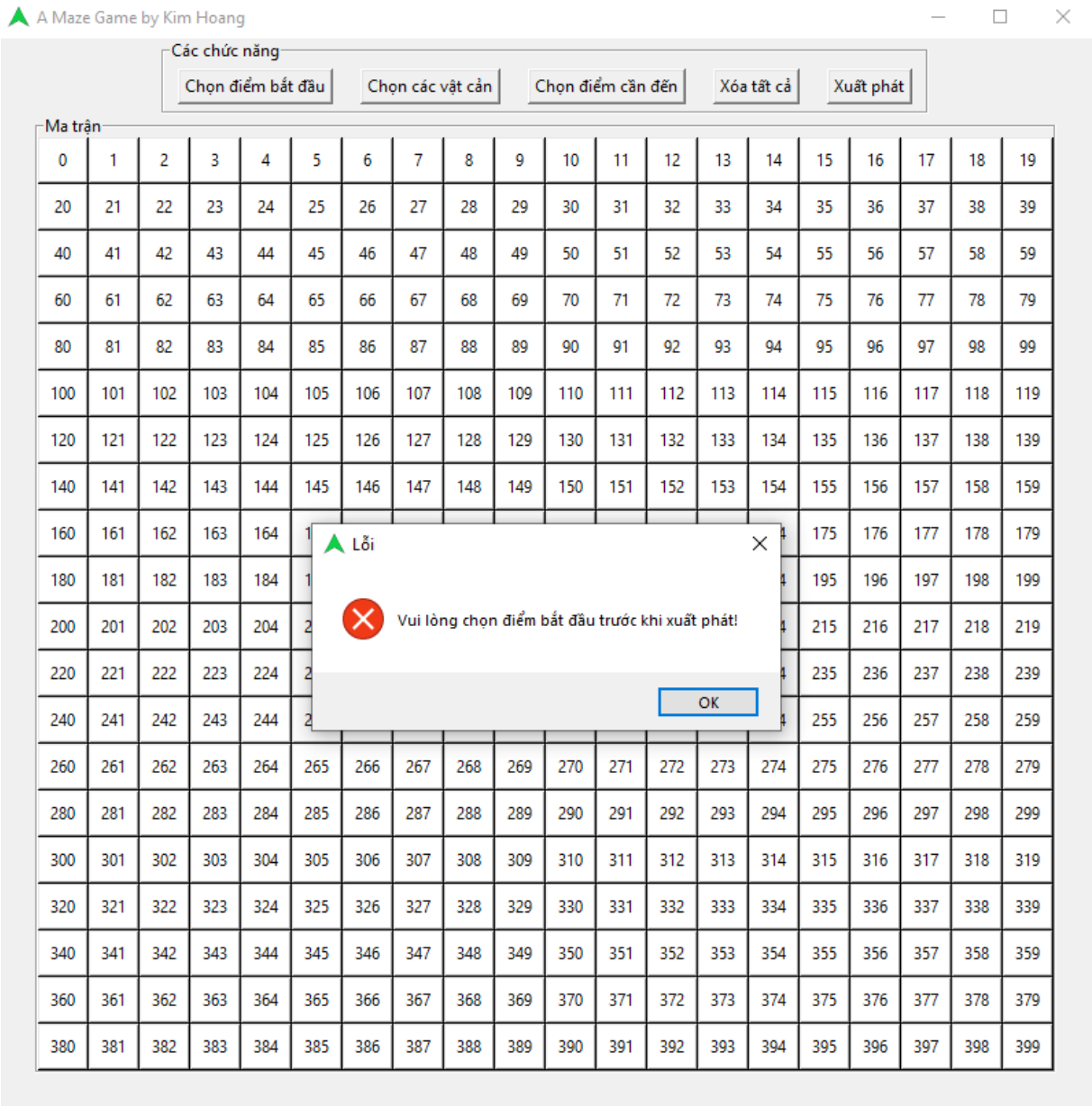
3.4. Giao diện khi người dùng bấm nút Xóa tất cả



Hình 6 : Giao diện khi người dùng bấm nút Xóa tất cả

- Ở giao diện này, khi người dùng bấm nút xóa tất cả, ứng dụng sẽ Clear hết tất cả các điểm cũng như vật cản và đưa giao diện về lại trạng thái ban đầu.

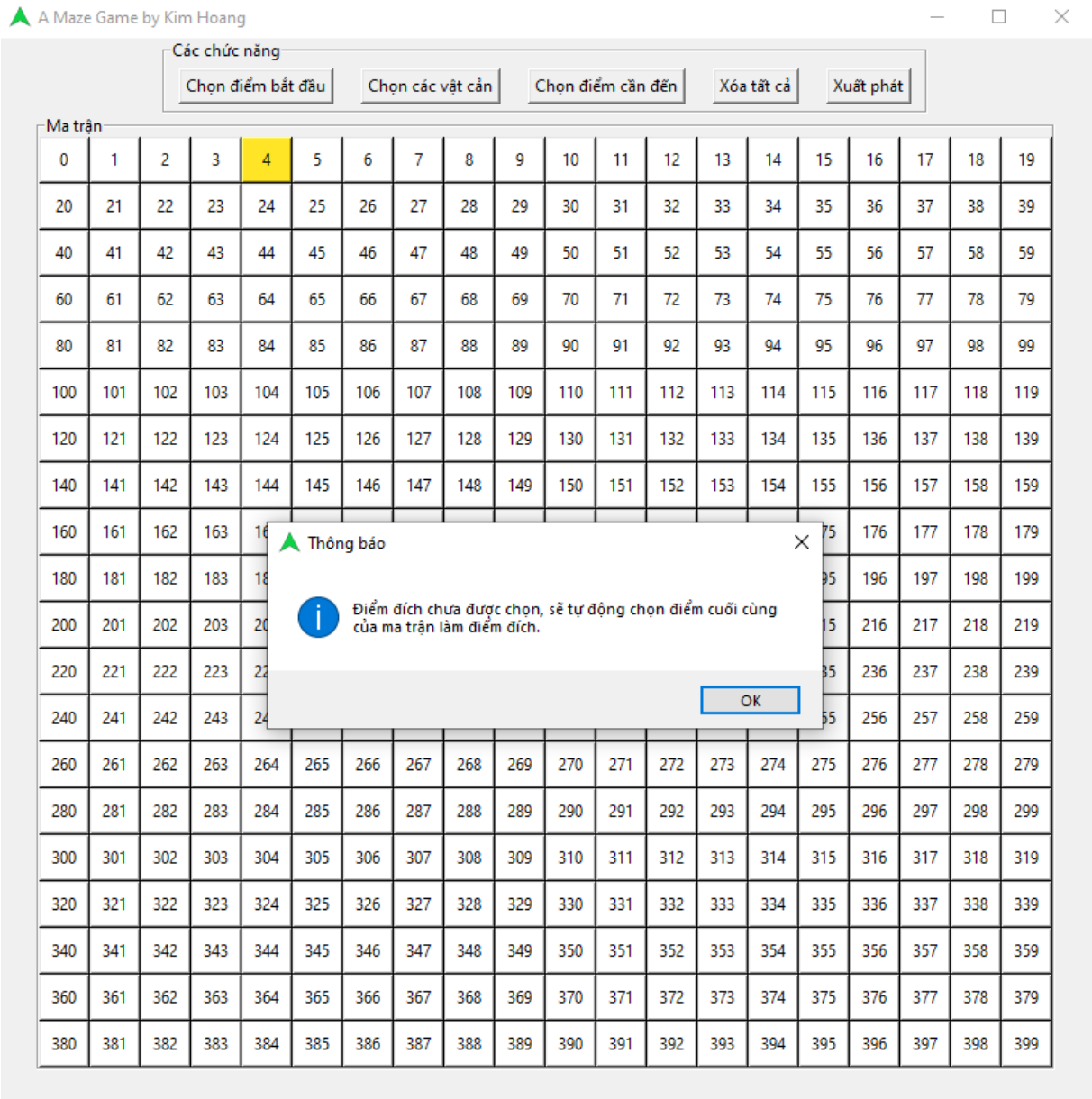
3.5. Giao diện khi người dùng chưa chọn điểm bắt đầu



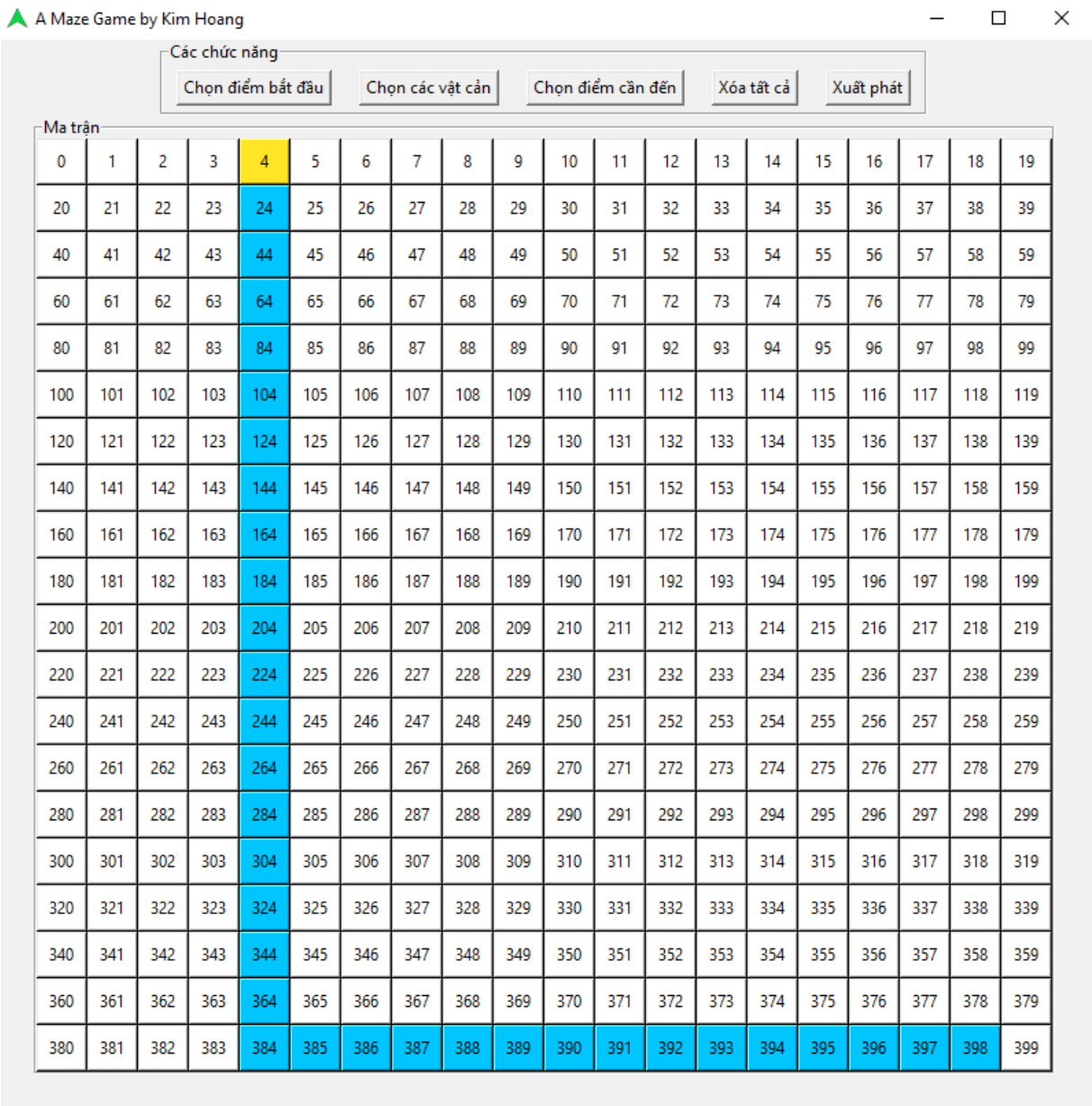
Hình 7 : Giao diện khi người dùng chưa chọn điểm bắt đầu

- Ở giao diện này, khi người dùng chưa chọn điểm bắt đầu cho nó mà bấm vào điểm xuất phát, hệ thống sẽ hiện một MessageBox báo lỗi "Vui lòng chọn điểm bắt đầu trước khi xuất phát" và yêu cầu người dùng phải chọn điểm bắt đầu khi xuất phát.

3.6. Giao diện khi người người dùng không chọn điểm cần đến



Hình 8 : Giao diện khi người người dùng không chọn điểm cần đến



Hình 9 : Giao diện tìm đường đi khi không có điểm đích đến

- Ở giao diện này, khi người dùng chọn điểm bắt đầu (màu vàng) mà chưa chọn điểm đích mà bấm xuất phát thì hệ thống sẽ thông báo một MessageBox thông báo " Điểm đích chưa được chọn, sẽ tự động chọn điểm cuối cùng của ma trận làm điểm đích" và nó sẽ tự động tìm đường đi ngắn nhất tới điểm cuối cùng của ma trận.

KẾT LUẬN

1. Kết quả đạt được

- Sau 4 tuần thực hiện, với nhiều nỗ lực và cũng nhận được nhiều sự giúp đỡ của thầy cô và bạn bè, em thực hiện đề tài đã thực hiện được các chức năng cơ bản như đã đặt ra từ ban đầu của báo cáo. Qua đó, phần nào cũng nâng cao được khả năng lập trình cũng như việc tự nghiên cứu, tìm hiểu thông các tài liệu khác nhau, thực sự giúp ích cho công việc của bản thân sau khi ra trường. Về phần mềm xây dựng game a maze sử dụng thuật toán Dijkstra, có thể nói chương trình đã có thể vận hành được một cách cơ bản
- Kết quả đạt được là việc tìm được đường đi tối ưu từ điểm xuất phát đến điểm đích trong mê cung, tránh được các vật cản. Đoạn đường tối ưu này không chỉ là một đường đi đơn thuần mà còn là con đường ngắn nhất và an toàn nhất giữa hai điểm trong mê cung.
- Khi thuật toán Dijkstra hoàn thành tốt nhiệm vụ của mình, người dùng sẽ thấy đường đi tối ưu được đánh dấu rõ ràng trên bản đồ mê cung, giúp họ dễ dàng nhận biết và tiếp tục hành trình một cách hiệu quả. Sự thành công của việc tìm đường đi này không chỉ mang lại cảm giác tự hào và hài lòng cho người chơi mà còn thể hiện sức mạnh và khả năng của thuật toán trong việc giải quyết vấn đề phức tạp như tìm đường trong mê cung.

2. Hướng phát triển của đề tài

- Tối ưu hóa thuật toán: Cải thiện hiệu suất của thuật toán Dijkstra bằng cách áp dụng các phương pháp tối ưu hóa như thu gọn đồ thị, sử dụng các cấu trúc dữ liệu hiệu quả hơn.
- Mở rộng tính năng: Phát triển thêm các tính năng cho ứng dụng như tạo mê cung ngẫu nhiên, cho phép người dùng tùy chỉnh kích thước và độ phức tạp của mê cung.
- Giao diện người dùng: Cải thiện giao diện người dùng để tạo trải nghiệm người dùng tốt hơn, bao gồm thêm hướng dẫn sử dụng, giao diện đồ họa thú vị và trực quan.

- Tích hợp trí tuệ nhân tạo: Kết hợp thuật toán Dijkstra với các phương pháp trí tuệ nhân tạo khác như học máy để tạo ra một hệ thống thông minh hơn có khả năng học và cải thiện qua thời gian.
- Ứng dụng thực tế: Phát triển ứng dụng này thành một sản phẩm thực tế có thể được sử dụng trong nhiều lĩnh vực như điều hướng robot, trò chơi điện tử hoặc giáo dục.

TÀI LIỆU THAM KHẢO

Tài liệu website

[1 https://www.youtube.com/watch?v=MHva487-DH0- Lập trình giao diện đồ hoạ đa luồng với Tkinter trên Python - Mì AI

[2] https://v1study.com/study-python-gioi-thieu-ve-tkinter.html – Python : Giới thiệu về Tkinter

[3] https://reeborg.ca/docs/en/reference/mazes.html - Nguồn tài nguyên xây dựng Game