**BỘ XÂY DỰNG**

**TRƯỜNG ĐẠI HỌC KIẾN TRÚC HÀ NỘI**

**------- \*\*\* -------**

Logo

Description automatically generated

**BÁO CÁO**

**ĐỒ ÁN TRÍ TUỆ NHÂN TẠO**

**ĐỀ TÀI:**

**NGHIÊN CỨU VÀ XÂY DỰNG TRÒ CHƠI TÌM ĐƯỜNG ĐI TRONG MÊ CUNG CÓ SỬ DỤNG THUẬT TOÁN A\***

**GIẢNG VIÊN HƯỚNG DẪN**

**Ths. Nguyễn Thị Huệ**

**SINH VIÊN THỰC HIỆN**

|  |  |  |
| --- | --- | --- |
| **1. Lê Quang Diễm** | **MSSV: 2155010047** | **Lớp: 21CN2** |
| **2. Trần Thanh Hiền** | **MSSV: 2155010092** | **Lớp: 21CN2** |
| **3. Trần Nhật Nam** | **MSSV: 2155010177** | **Lớp: 21CN2** |
| **4. Hoàng Duy Tưởng** | **MSSV: 2155010242** | **Lớp: 21CN2** |
| **5. Hà Quốc Việt** | **MSSV: 2155010277** | **Lớp: 21CN2** |

***Hà Nội, ngày … tháng 10 năm 2023***

**MỤC LỤC**

[**DANH MỤC HÌNH VẼ** 1](#_Toc149213148)

[**DANH MỤC CÁC TỪ VIẾT TẮT, TỪ TIẾNG ANH** 3](#_Toc149213149)

[**LỜI MỞ ĐẦU** 4](#_Toc149213150)

[**CHƯƠNG 1. CƠ SỞ LÝ THUYẾT** 5](#_Toc149213151)

[**1.1. Tổng quan về trí tuệ nhân tạo** 5](#_Toc149213152)

[**1.1.1. Khái niệm về trí tuệ nhân tạo** 5](#_Toc149213153)

[**1.1.2. Lịch sử hình thành và phát triển của trí tuệ nhân tạo** 5](#_Toc149213154)

[**1.1.3. Các hướng nghiên cứu trí tuệ nhân tạo** 6](#_Toc149213155)

[**1.1.4. Phân loại trí tuệ nhân tạo** 8](#_Toc149213156)

[**1.1.5. Mặt tích cực và tiêu cực của trí tuệ nhân tạo** 8](#_Toc149213157)

[**1.1.6. Ứng dụng của trí tuệ nhân tạo vào thực tế** 9](#_Toc149213158)

[**1.2. Tổng quan về đề tài** 10](#_Toc149213159)

[**1.3. Giới thiệu các công nghệ áp dụng cho đề tài** 11](#_Toc149213160)

[**1.3.1. Ngôn ngữ lập trình Python** 11](#_Toc149213161)

[**1.3.2. Các thư viện sử dụng trong đề tài** 12](#_Toc149213162)

[**CHƯƠNG 2: THUẬT TOÁN ÁP DỤNG ĐỀ TÀI** 14](#_Toc149213163)

[**2.1. Giới thiệu thuật toán AStar: A\*** 14](#_Toc149213164)

[**2.1.1. Mô tả thuật toán** 14](#_Toc149213165)

[**2.1.2. Tính chất của thuật toán** 14](#_Toc149213166)

[**2.1.3. Thuật toán A\*** 14](#_Toc149213167)

[**2.1.4. Ví dụ về thuật toán A\*** 15](#_Toc149213168)

[**2.2. Áp dụng thuật toán vào đề tài** 16](#_Toc149213169)

[**CHƯƠNG 3: XÂY DỰNG VÀ TRIỂN KHAI ỨNG DỤNG** 19](#_Toc149213170)

[**3.1. Sản phẩm nghiên cứu** 19](#_Toc149213171)

[**3.1.1. Cài đặt các thư viện cần thiết** 19](#_Toc149213172)

[**3.1.2 Đưa thuật toán vào các sự kiện của nhân vật.** 20](#_Toc149213173)

[**3.1.3. Giao diện trò chơi** 22](#_Toc149213174)

[**3.1.4 Thiết kế giao diện** 23](#_Toc149213175)

[**3.1.5. Các bước khởi chạy chương trình** 24](#_Toc149213176)

[**3.2. Kết luận** 26](#_Toc149213177)

[**TÀI LIỆU THAM KHẢO** 28](#_Toc149213178)

**DANH MỤC HÌNH VẼ**

[Hình 2.1. Ví dụ về thuật toán A\* 13](#_Toc148879617)

[Hình 3.1. Hình ảnh giao diện bắt đầu trò chơi 20](#_Toc148879618)

[Hình 3.2. Hình ảnh giao diện kết thúc trò chơi 21](#_Toc148879619)

**DANH MỤC CÁC TỪ VIẾT TẮT, TỪ TIẾNG ANH**

|  |  |  |
| --- | --- | --- |
| **STT** | **Kí tự viết tắt** | **Từ viết tắt** |
| 1 | AI | Artificial Intelligence |
| 2 | ANI | Artificial Narrow Intelligence |
| 3 | AGI | Artificial General Intelligence |
| 4 | ASI | Artificial Super Intelligence |
| 5 | Pgzrun | Pygame Zero Run |
| 6 | NLP | Natural Language Processing |

**LỜI MỞ ĐẦU**

Thế giới đang có những bước chuyển mình rõ rệt trong bối cảnh cuộc cách mạng công nghiệp lần thứ tư. Đó là sự ra đời liên tiếp của những phát mình về công nghệ: công nghệ cảm biến, thực tế ảo, trí tuệ nhân tạo… Khoa học, công nghệ dần trở thành một phần không thể thiếu trong cuộc sống thường ngày. Cuộc cách mạng lần thứ tư đã tác động mạnh mẽ đến mọi quốc gia, doanh nghiệp và người dân ở mọi nơi trên thế giới, tạo tiền đề cho sự phát triển mạnh mẽ của khoa học và công nghệ.

Một trong những xu hướng công nghệ mang tính đột phá của cuộc cách mạng công nghiệp 4.0 là Trí tuệ nhân tạo - Artificial Intelligence ( hay còn được viết tắt là AI). AI được sử dụng phổ biến trong nhiều lĩnh vực dưới nhiều hình thức khác nhau. Đặc biệt trí thông minh của trí tuệ nhân tạo được các nhà khoa học sử dụng để tạo ra những trò chơi đặc sắc và không kém phần hấp dẫn.

Nhận thấy ứng dụng đặc biệt của AI trong việc tạo ra các trò chơi, nhóm 8 quyết định lựa chọn đề tài  : “ Nghiên cứu trò chơi tìm đường đi trong mê cung sử dụng thuật toán A\* ”. Qua đó, không chỉ để thấy được ứng dụng của AI trong trò chơi mà còn thấy rõ được cách thuật toán hoạt động trong trò chơi.

# **CHƯƠNG 1. CƠ SỞ LÝ THUYẾT**

## **1.1. Tổng quan về trí tuệ nhân tạo**

Kể từ khi phát minh ra máy tính hoặc máy móc, khả năng thực hiện nhiều nhiệm vụ khác nhau của chúng ngày càng tăng theo cấp số nhân. Con người đã phát triển sức mạnh của hệ thống máy tính về các lĩnh vực làm việc đa dạng, tốc độ ngày càng tăng và kích thước giảm dần theo thời gian. Một nhánh của Khoa học Máy tính có tên Trí tuệ nhân tạo, theo đuổi việc tạo ra các máy tính hoặc máy móc thông minh như con người.

### **1.1.1. Khái niệm về trí tuệ nhân tạo**

Theo cha đẻ của Trí tuệ nhân tạo, John McCarthy, đó là *“Khoa học và kỹ thuật chế tạo ra những cỗ máy thông minh, đặc biệt là các chương trình máy tính thông minh”.*

Trí tuệ nhân tạo(Artificial Intelligence - AI) là một lĩnh vực trong khoa học máy tính và công nghệ liên quan đến việc phát triển máy tính và hệ thống có khả năng thực hiện các nhiệm vụ thông minh mà trước đây chỉ có con người mới có thể thực hiện được. Nó liên quan đến việc xây dựng các chương trình máy tính và hệ thống có khả năng học từ dữ liệu, tự động cải thiện hiệu suất và thực hiện các tác vụ như nhận diện hình ảnh, xử lý ngôn ngữ tự nhiên và ra quyết định dựa trên thông tin thu nhập được. AI được hoàn thiện bằng cách nghiên cứu cách suy nghĩ của bộ não con người cũng như cách con người học hỏi, quyết định và làm việc trong khi cố gắng giải quyết vấn đề, sau đó sử dụng kết quả của nghiên cứu này làm cơ sở để phát triển hệ thống và phần mềm thông minh.

### **1.1.2. Lịch sử hình thành và phát triển của trí tuệ nhân tạo**

Các dấu mốc lịch sử và phát triển trí tuệ nhân tạo đã mang lại những hứa hẹn trong lĩnh vực công nghệ trên toàn thế giới. Để hiểu rõ hơn về lịch sử hình thành, phát triển của trí tuệ nhân tạo, hãy cùng điểm qua một vài cột mốc đáng nhớ trong quá trình phát triển của nó

**Những năm 1950 - Thập kỷ đầu của AI**

1950s: Là thời kỳ đầu của nghiên cứu về AI. Từ các ý tưởng của Alan Turing và John von Neumann, người ta đã bắt đầu xem xét khả năng của máy tính trong việc mô phỏng tư duy con người.

1956: Tổ chức Hội nghị Dartmouth, nơi thuật ngữ "trí tuệ nhân tạo" được đề xuất chính thức. Giai đoạn này thấy sự hứa hẹn về khả năng sáng tạo của máy tính.

**Những năm 1960 - 1970 - Sự phát triển đầu tiên:**

1960s: Các nghiên cứu ban đầu về học máy và hệ thống dựa trên quy tắc bắt đầu. Theo thời gian, AI bắt đầu được xem xét trong các lĩnh vực như lý thuyết trò chơi và xử lý ngôn ngữ tự nhiên.

1963: Theo lời đề nghị của John McCarthy, LISP (ngôn ngữ lập trình đầu tiên được tạo ra để hỗ trợ nghiên cứu AI) ra đời.

1966: Dendral, hệ thống đầu tiên sử dụng luật để giải quyết vấn đề nhận diện hóa chất, được phát triển.

1970s: Đánh dấu sự xuất hiện của hệ thống dựa trên quy tắc và cách tiếp cận mới trong việc xử lý tri thức.

**Những năm 1980 - 1990 - Thành công và khủng hoảng AI:**

1980s: Thập kỷ này thấy sự phát triển của các hệ thống học máy đầu tiên và sự tăng trưởng của AI. Tuy nhiên, cuối thập kỷ, AI đối mặt với khủng hoảng do kỳ vọng quá lớn.

1987: Mạng nơ-ron nhiều tầng (multilayer neural networks) trở nên phổ biến trong học máy.

1988: Sử dụng mã nguồn mở trở nên phổ biến, giúp cộng đồng nghiên cứu AI phát triển nhanh hơn.

**Những năm 2000 - 2010 - Sự hồi sinh của AI:**

2000s: AI trải qua giai đoạn phục hồi nhờ vào sự gia tăng của dữ liệu lớn và tiến bộ trong học sâu. Trợ lý ảo như Siri của Apple và các dự án về xe tự hành bắt đầu xuất hiện.

2006: Học sâu trở thành một phần quan trọng của AI với việc phát triển mạng nơ-ron sâu.

**Những năm 2010 - đến nay - Đỉnh cao và triển vọng:**

2010s: AI đạt đỉnh với các ứng dụng như nhận diện hình ảnh, xử lý ngôn ngữ tự nhiên và tự động hóa nhiều ngành công nghiệp.

2011: IBM Watson giành chiến thắng trong trò chơi Trí Tuệ Nhân Tạo Jeopardy!, là một ví dụ về sự phát triển của AI.

2016: AlphaGo của DeepMind (công ty con của Google) đánh bại nhà vô địch cờ vua thế giới, Lee Sedol. Đây được xem là một trong những thành tựu quan trọng của AI.

Hiện tại và tương lai: AI đang tiếp tục phát triển mạnh mẽ với sự tập trung vào phát triển trí tuệ nhân tạo mạnh (AGI) có khả năng tự hiểu và thực hiện nhiều nhiệm vụ như con người. AI đang thúc đẩy sự thay đổi đáng kể trong cuộc sống và kinh tế toàn cầu, và triển vọng cho tương lai là rất hứa hẹn.

### **1.1.3. Các hướng nghiên cứu trí tuệ nhân tạo**

Trong lĩnh vực trí tuệ nhân tạo (AI), có nhiều hướng nghiên cứu đang được quan tâm và phát triển mạnh mẽ. Những hướng nghiên cứu này đóng vai trò quan trọng trong việc xây dựng và phát triển các ứng dụng AI tiên tiến.

**Học máy: (Machine Learning):** Học máy tập trung vào phân tích và hiểu dữ liệu để máy tính có thể tự động học và cải thiện từ kinh nghiệm. Các thuật toán Machine Learning giúp máy tính nhận biết mô hình, xu hướng và thông tin ẩn trong dữ liệu. Học máy bao gồm các hình thức chính là học có giám sát, học không giám sát và học tăng cường. Trong học có giám sát, hệ thống nhận đầu vào và đầu ra và cần tìm các quy tắc để ánh xạ đầu vào thành đầu ra. Học không giám sát, hệ thống không nhận được đầu ra và phải tìm các mẫu hoặc quy tắc từ thông tin đầu vào. Với học tăng cường, hệ thống chỉ biết kết quả cuối cùng của toàn bộ quá trình chứ không phải kết quả của từng bước riêng lẻ.

**Thị giác máy tính (Computer Vision)**: Đây là lĩnh vực trí tuệ nhân tạo liên quan đến việc thu thập, xử lý, phân tích và nhận dạng thông tin hình ảnh thu được từ các cảm biến hình ảnh như máy ảnh. Mục đích của thị giác máy tính là chuyển đổi thông tin thu được thành dạng biểu diễn ở cấp độ cao hơn để máy tính có thể hiểu được nó. Ví dụ: dựa trên ảnh chụp văn bản, nó phải trả về mã UNICODE của các chữ cái được in trong văn bản đó. Sau đó, có thể sử dụng biểu diễn cấp cao hơn của thông tin cảm biến hình ảnh để đưa ra quyết định. Thị giác máy tính bao gồm các vấn đề chính sau: nhận dạng mẫu, phân tích chuyển động, tạo cảnh 3D, cải thiện chất lượng hình ảnh.

**Xử lý ngôn ngữ tự nhiên (Natural Language Processing - NLP)**: Lĩnh vực này tập trung vào việc phân tích dữ liệu, thông tin nhận được dưới dạng văn bản hay âm thanh và được trình bày dưới dạng ngôn ngữ tự nhiên. Ví dụ, chúng ta có thể ra lệnh cho máy tính bằng cách nói với máy tính như với người thường thay vì phải gõ các lệnh quy ước. Xử lý ngôn ngữ tự nhiên bao gồm ba giai đoạn chính: nhận dạng tiếng nói, xử lý thông tin đã được biểu diễn dưới dạng văn bản, và biến đổi từ văn bản thành tiếng nói .

**Nhận dạng mẫu:** Đây lĩnh vực nghiên cứu quan trọng nhất trong thị giác máy tính. Bản thân nhận dạng mẫu được chia thành nhiều vấn đề nhỏ hơn và cụ thể hơn, chẳng hạn như. Ví dụ: nhận diện khuôn mặt, vân tay, chữ viết; nhận dạng đối tượng trong ảnh hoặc video và xác định đối tượng. Mặc dù việc này tương đối dễ dàng đối với con người nhưng việc nhận dạng tự động thường khó khăn hơn nhiều. Hiện tại, máy tính chỉ có thể nhận dạng một số loại đối tượng nhất định, chẳng hạn như chữ in và khuôn mặt người, với độ chính xác gần như con người.

**Biểu diễn tri thức (knowledge representation):** Nhiều vấn đề về AI đòi hỏi phải suy luận dựa trên sự trực quan hóa của thế giới xung quanh. Để suy luận, các sự kiện, thông tin và kiến ​​thức về thế giới xung quanh phải được thể hiện dưới dạng mà máy tính có thể “hiểu” được, chẳng hạn như logic hoặc ngôn ngữ trí tuệ nhân tạo. Thông thường hệ thống cần có kiến ​​thức về: đối tượng hoặc thực thể, thuộc tính của chúng, phân loại và mối quan hệ giữa các đối tượng, tình huống, sự kiện, trạng thái, thời gian, nguyên nhân và kết quả, kiến ​​thức về kiến ​​thức (chúng ta biết về kiến ​​thức mà người khác có), v.v.. Trong khuôn khổ nghiên cứu về biểu diễn tri thức, một số phương pháp biểu diễn đã được phát triển và ứng dụng như: logic, mạng ngữ nghĩa, frame, các quy tắc, bản thể học (ontology).

### **1.1.4. Phân loại trí tuệ nhân tạo**

**Loại 1: Trí tuệ nhân tạo yếu (Artificial Narrow Intelligence - ANI)**

Trí tuệ nhân tạo yếu còn được gọi là trí tuệ nhân tạo hẹp, là một loại trí tuệ nhân tạo mà hệ thống chỉ tập trung vào một tác vụ hoặc lĩnh vực cụ thể. ANI không có khả năng tự học và thực hiện nhiệm vụ ngoài phạm vi đã đào tạo.Ví dụ điển hình về ANI như là Trợ lý ảo như Siri và Alexa: Các trợ lý ảo này có thể trả lời câu hỏi, thực hiện các tác vụ như đặt hẹn hay đọc tin tức, nhưng chúng không có khả năng thực hiện nhiều loại công việc khác ngoài các nhiệm vụ cụ thể đã được lập trình

**Loại 2: Trí tuệ nhân tạo mạnh (Artificial General Intelligence - AGI)**

Trí tuệ nhân tạo mạnh đại diện cho một dạng trí tuệ nhân tạo có khả năng tự học và thực hiện nhiều loại nhiệm vụ khác nhau tương tự như con người.Không giống như ANI, AI mạnh không bị giới hạn bởi bất kỳ ràng buộc nào. Nghĩa là nó có ý thức tự nhận thức để giải quyết các vấn đề, học và lên kế hoạch cho tương lai. Nói khác đi, nó có thể hoàn toàn tự đưa ra các quyết định mà không cần phải được huấn luyện từ trước. Hiện tại, AGI vẫn là một mục tiêu chưa thực hiện hoàn toàn và đang là đối tượng của nhiều nghiên cứu và phát triển trong lĩnh vực trí tuệ nhân tạo

**Loại 3: Siêu trí tuệ nhân tạo (Artificial Super Intelligence )**

Khái niệm "Siêu Trí Tuệ Nhân Tạo” vẫn là một phần của tương lai và chưa có ví dụ cụ thể về hệ thống ASI trong thế giới thực. Tuy nhiên, nó thường xuất hiện trong văn học và tưởng tượng khoa học như một đề tài cảnh báo về các tác động tiềm năng của một trí tuệ nhân tạo vượt trội đối với nhân loại.

Một trong những ví dụ phổ biến về ASI có thể thấy trong tiểu thuyết và phim là: Máy Tư Duy trong Tiểu Thuyết: Trong tiểu thuyết như "Sự Sống Của Pi" của Yann Martel, tác giả đưa ra hình dung về một máy tính ASI có khả năng tư duy và hiểu biết vượt trội hơn con người. Trong tiểu thuyết, máy tính này có khả năng tạo ra những suy tư phức tạp và sáng tạo.

Tuy nhiên, cần lưu ý rằng ASI vẫn chỉ là một đề tài tưởng tượng và đang nằm ngoài phạm vi của công nghệ hiện tại. Siêu Trí Tuệ Nhân Tạo, nếu tồn tại trong tương lai, sẽ đòi hỏi sự phát triển và nghiên cứu đáng kể và cần quan tâm đến nhiều vấn đề đạo đức, định chế, và an ninh để đảm bảo sự hài hòa và an toàn cho nhân loại

### **1.1.5. Mặt tích cực và tiêu cực của trí tuệ nhân tạo**

**Mặt tích cực:**

**Giải Quyết Vấn Đề Phức Tạp:** AI có khả năng xử lý, phân tích dữ liệu lớn một cách nhanh chóng và hiệu quả, giúp tìm ra thông tin quan trọng và đưa ra quyết định dựa trên dữ liệu, giải quyết các vấn đề phức tạp mà con người không thể thực hiện trong thời gian ngắn. Điều này có ứng dụng trong nghiên cứu khoa học, y tế, và nhiều lĩnh vực khác.

**Tăng Cường Hiệu Suất Công Việc:** AI có thể thực hiện các tác vụ lặp lại và tốn thời gian nhanh hơn và chính xác hơn so với con người, giúp cải thiện hiệu suất làm việc và giảm tải công việc cho con người.

**Xóa bỏ khoảng cách ngôn ngữ:** Với các phương tiện dịch thuật bằng văn bản, hình ảnh, âm thanh ngày càng phát triển một cách mạnh mẽ, chính xác sẽ giúp con người ở trên mọi quốc gia có thể nói chuyện với nhau, thoải mái tiếp xúc mà không cần đến phiên dịch hay lo ngại về khoảng cách ngôn ngữ. Từ đó, mọi người có thêm nhiều cơ hội để làm quen, học tập, làm việc với bạn bè trên toàn thế giới. Mở rộng cơ hội hòa nhập lớn, xóa bỏ khoảng cách về ngôn ngữ cũng như sự phân biệt chủng tộc, quốc gia

Tuy AI mang lại nhiều lợi ích, nhưng cần quản lý cẩn thận để đảm bảo rằng nó được sử dụng một cách đạo đức và an toàn, và đảm bảo rằng nó không gây ra các vấn đề đạo đức hoặc an ninh mới.

**Mặt tiêu cực**

**Sự lệ thuộc của con người vào AI:** AI đã làm được những điều mà một người bình thường không thể hoàn thành. Tuy nhiên, trong những công việc cụ thể mang tính chuyên sâu và sáng tạo mới AI hiện nay chưa thể đáp ứng. Nếu không có sự kiểm tra, giám sát, thẩm định, chỉnh sửa, hoàn thiện của con người, những kết quả của AI sẽ dẫn tới sự sai lầm trong các quyết định.

**Sử dụng AI vào mục đích không chính đáng:** Sử dụng AI để thu thập và phân tích dữ liệu cá nhân có thể đe dọa quyền riêng tư của cá nhân. Việc lưu trữ và truy cập dữ liệu cá nhân phải được quản lý cẩn thận để đảm bảo an toàn và bảo vệ quyền riêng tư. AI có thể được sử dụng để tạo ra các cuộc tấn công mạng và thâm nhập vào hệ thống an ninh. Điều này đòi hỏi quản lý an ninh mạng tốt hơn để đối phó với các rủi ro này.

**Nguy cơ về trách nhiệm đạo đức:** AI là sản phẩm do con người tạo ra nhằm phục vụ và cải thiện cuộc sống của con người ngày càng phát triển hơn. Song, AI là công cụ của của con người, không có ý thức, nhân cách, tư cách pháp nhân trong xã hội loài người. Với khả năng ra quyết định và hành động tự động, AI đặt ra câu hỏi về trách nhiệm đạo đức. Ai chịu trách nhiệm nếu AI gây ra hậu quả không mong muốn?

### **1.1.6. Ứng dụng của trí tuệ nhân tạo vào thực tế**

**Y tế:** Trong lĩnh vực chẩn đoán y học, AI có thể được sử dụng trong việc phân tích và đưa ra dự đoán dựa trên dữ liệu y tế của bệnh nhân. Một ví dụ cụ thể là hệ thống hỗ trợ chẩn đoán hình ảnh y khoa. AI có thể được áp dụng để phân tích hình ảnh y khoa như chụp X-quang, CT scans. Hệ thống có thể phát hiện tổn thương hay những biểu hiện bất thường trên hình ảnh này. AI có thể giúp bác sĩ tăng cường khả năng chẩn đoán và đưa ra quyết định điều trị nhanh hơn và chính xác hơn.

**Giáo dục:** Sự xuất hiện của trí tuệ nhân tạo đang giúp mang lại những thay đổi lớn trong lĩnh vực giáo dục. Các hoạt động giáo dục như chấm điểm hoặc dạy kèm học sinh có thể được tự động hóa với sự trợ giúp của công nghệ AI. Nhiều trò chơi và phần mềm giáo dục được tạo ra để đáp ứng nhu cầu cụ thể của từng học sinh, giúp học sinh cải thiện việc học theo tốc độ của riêng mình. Các ứng dạy ngoại ngữ dựa trên AI, có thể giúp học sinh phát triển các kỹ năng ngôn ngữ bằng cách cung cấp luyện tập và phản hồi liên tục

**Công nghiệp nặng :** Ứng dụng của trí tuệ nhân tạo trong công nghiệp nặng là vô cùng đa dạng. Từ quá trình tự động hóa sản xuất, quản lý và theo dõi hoạt động của máy móc, cho đến việc phân tích dữ liệu và dự báo để tối ưu hóa quy trình sản xuất. Nhờ vào AI, các doanh nghiệp có thể tiết kiệm thời gian, công sức và nguồn lực để tập trung vào việc cải thiện chất lượng sản phẩm và gia tăng sự cạnh tranh.Ngoài ra, AI còn có khả năng phân tích lượng dữ liệu lớn để tìm ra những xu hướng và mô hình sản xuất hiệu quả. Điều này giúp các doanh nghiệp trong công nghiệp nặng đưa ra các quyết định chiến lược thông minh để tối ưu hóa hoạt động sản xuất và tiếp cận với các thị trường mới.

**Tài chính:** Trong lĩnh vực tài chính, AI đã mang lại những ứng dụng đáng ngạc nhiên trong việc quản lý tài chính. Một trong những ứng dụng chính của AI trong tài chính là việc phân tích dữ liệu. Nhờ vào khả năng học máy và thu thập thông tin từ dữ liệu lớn, AI có thể phân tích xu hướng thị trường, thực hiện các giao dịch tự động. Ngoài ra, AI cũng được sử dụng rộng rãi trong việc kiểm soát rủi ro và phòng ngừa gian lận. Công nghệ này có khả năng theo dõi hàng triệu giao dịch và phát hiện các hoạt động bất thường hoặc không hợp lý. Điều này giúp ngăn chặn các hành vi gian lận và bảo vệ hệ thống tài chính khỏi những rủi ro tiềm ẩn.

## **1.2. Tổng quan về đề tài**

Trò chơi tìm đường đi trong mê cung đã tồn tại từ lâu và luôn thu hút sự chú ý của người chơi. Phát triển của trò chơi tìm đường đi trong mê cung có một lịch sử dài và phức tạp, bắt đầu từ các trò chơi trên giấy, trong đó người chơi cố gắng vẽ một đường đi từ một điểm đến một điểm khác trên một bản đồ mê cung. Sau đó, trò chơi này được phát triển thành trò chơi điện tử phức tạp trên nhiều nền tảng như máy tính, điện từ cầm tay, máy trò chơi…. Có nhiều loại hình trò chơi tìm đường đi trong mê cung, từ trò chơi cổ điển Maze War (1973) được coi là trò chơi đầu tiên trong thể loại tìm đường đi trong mê cung, cho phép người chơi di chuyển trong một mê cung 3D và bắn nhau với những người chơi khác; đến các trò chơi phiêu lưu như The Legend of Zeldan (1986), trò chơi giải đố như Monument valley (2014) , và trò chơi hành động như Bomberman….. Như vậy, trò chơi tìm đường đi trong mê cung đã trải qua một hành trình dài và có sự phát triển đa dạng trên nhiều nền tảng khác nhau. Từ những trò chơi đơn giản trên giấy , thời kì đầu của máy tính, cho đến hiện đại trên điện thoại di động, thể loại này vẫn luôn thu hút sự quan tâm của người chơi và nhà phát triển trò chơi.

Mục tiêu của trò chơi này là tìm ra con đường ngắn nhất hoặc tối ưu nhất để thoát khỏi mê cung. Đây là một trò chơi logic thú vị, trong đó người chơi phải tìm ra con đường để thoát khỏi mê cung. Trong trò chơi này, người chơi sẽ đối mặt với nhiều thách thức và rào cản khác nhau. Họ phải suy nghĩ logic, dự tính và quan sát để tìm ra con đường thông qua các lối đi hẹp và các bức tường không thể xuyên qua.

Trò chơi tìm đường đi trong mê cung có nhiều biến thể khác nhau, từ phiên bản truyền thống với các ô vuông và ký tự cho đến phiên bản 3D sống động với hiệu ứng âm thanh hấp dẫn. Có những trò chơi chỉ gồm một mê cung duy nhất, trong khi có những trò chơi có nhiều mức độ khác nhau với các mê cung phức tạp hơn. Một số phiên bản có thêm yếu tố thời gian hoặc áp lực để người chơi hoàn thành nhanh hơn. Người chơi có thể lựa chọn giữa việc giải quyết các câu đố trong một môi trường yên tĩnh hoặc tranh tài với bạn bè thông qua các tính năng multiplayer. Một số game tìm đường đi trong mê cung cũng kết hợp yếu tố phiêu lưu, cho phép người chơi khám phá các phòng chứa bí mật, thu thập vật phẩm và đối phó với quái vật. Điều này tạo ra sự hấp dẫn và thử thách không chỉ trong việc tìm đường đi mà còn trong việc hoàn thành các nhiệm vụ khác nhau.

Trò chơi tìm đường đi trong mê cung không chỉ giúp người chơi rèn luyện kỹ năng logic, tư duy linh hoạt và khả năng quan sát, mà còn mang lại niềm vui và sự thỏa mãn khi thành công trong việc giải quyết các câu đố khó khăn.

## **1.3. Giới thiệu các công nghệ áp dụng cho đề tài**

### **1.3.1. Ngôn ngữ lập trình Python**

Python là ngôn ngữ lập trình cấp cao được sử dụng cho mục đích lập trình chung, được tạo bởi Guido van Rossum và phát hành lần đầu tiên vào năm 1991. Python được thiết kế để dễ đọc, dễ học và dễ nhớ. Python là ngôn ngữ có hình thức rất bắt mắt, cấu trúc rõ ràng, thuận tiện cho người mới bắt đầu học lập trình, là ngôn ngữ lập trình dễ học, được sử dụng rộng rãi trong phát triển trí tuệ nhân tạo. Cấu trúc của Python cũng cho phép người dùng viết mã với số lần nhấn phím tối thiểu. Vào tháng 7 năm 2018, van Rossum từ chức lãnh đạo cộng đồng ngôn ngữ Python sau 30 năm.

Python hoàn toàn tạo kiểu động và dùng cơ chế cấp phát bộ nhớ tự động; do đó, nó tương tự như Perl, Ruby, Schema, Smalltalk và Tcl. Python được phát triển như một dự án mã nguồn mở và được quản lý bởi tổ chức phi lợi nhuận Python Software Foundation.

Ban đầu, Python được phát triển để chạy trên nền tảng Unix. Nhưng theo thời gian, Python dần dần mở rộng sang tất cả các hệ điều hành từ MS-DOS đến Mac OS, OS/2, Windows, Linux và các hệ điều hành họ Unix khác. Mặc dù có nhiều đóng góp cá nhân cho sự phát triển của Python nhưng ngày nay Guido van Rossum vẫn là tác giả chính của Python. Ông đóng vai trò quan trọng trong việc xác định hướng phát triển của Python. Python vẫn là một trong những ngôn ngữ lập trình phổ biến nhất

### **1.3.2. Các thư viện sử dụng trong đề tài**

**Thư viện pgzrun**

Thư viện pgzrun (viết tắt của "Pygame Zero Run") là một công cụ hỗ trợ để phát triển trò chơi và ứng dụng đồ họa sử dụng thư viện Pygame Zero. Pygame Zero là một thư viện Python dựa trên Pygame được tạo ra để giúp đơn giản hóa việc phát triển trò chơi 2D và ứng dụng đồ họa. Thư viện pgzrun có một số công dụng quan trọng trong việc phát triển trò chơi và ứng dụng đồ họa sử dụng Pygame Zero:

Tạo Trò Chơi 2D và Ứng Dụng Đồ Họa: pgzrun cho phép bạn tạo các trò chơi 2D và ứng dụng đồ họa dựa trên Pygame Zero một cách thuận tiện và dễ dàng. Bạn có thể vẽ đồ họa, quản lý sprite, xử lý sự kiện người dùng và thêm âm thanh vào trò chơi của mình.

Quản Lý Vòng Lặp Chính: pgzrun tự động quản lý vòng lặp chính của trò chơi. Điều này bao gồm việc cập nhật trạng thái của trò chơi, xử lý sự kiện người dùng (như bấm phím và chuột), và vẽ màn hình. Bạn không cần phải viết mã lặp chính một cách tường minh.

Đơn Giản Hóa Phát Triển: Sử dụng Pygame Zero và pgzrun, bạn không cần phải quá lo lắng về các chi tiết phức tạp của Pygame. Thay vì phải cấu hình Pygame một cách chi tiết, bạn có thể tập trung vào việc phát triển logic và giao diện của trò chơi hoặc ứng dụng của bạn.

Hỗ Trợ Đa Nền Tảng: Pygame Zero và pgzrun có thể hoạt động trên nhiều nền tảng, bao gồm Windows, macOS và Linux. Điều này giúp bạn tạo ra ứng dụng đa nền tảng dễ dàng hơn.

Tạo Prototypes và Trò Chơi Nhỏ: pgzrun rất hữu ích cho việc tạo ra các bản thử nghiệm (prototypes) nhanh chóng và phát triển các trò chơi nhỏ hoặc ứng dụng đồ họa. Bạn có thể kiểm tra ý tưởng một cách nhanh chóng trước khi tập trung vào phát triển lớn hơn.

**Thư viện heapq**

Thư viện heapq trong Python là một thư viện được sử dụng để thực hiện các thao tác trên các dạng cấu trúc dữ liệu gọi là "heap," trong đó heap thường được hiểu là "binary heap" (heap nhị phân). Thư viện heapq trong Python được sử dụng để thực hiện các thao tác trên các cấu trúc dữ liệu heap. Heap là một cấu trúc dữ liệu cây nhị phân đặc biệt, và heapq hỗ trợ các thao tác trên hai loại heap chính: min heap (heap nhỏ nhất) và max heap (heap lớn nhất). Dưới đây là các công dụng chính của thư viện heapq:

Tìm Phần Tử Tối Ưu: heapq cho phép bạn nhanh chóng tìm và lấy phần tử tối ưu từ heap (phần tử nhỏ nhất trong min heap hoặc phần tử lớn nhất trong max heap) mà không cần phải sắp xếp lại toàn bộ danh sách. Điều này hữu ích trong việc tìm kiếm phần tử tối ưu trong dãy dữ liệu lớn.

Triển Khai Hàng Đợi Ưu Tiên (Priority Queue): Heap thường được sử dụng để triển khai hàng đợi ưu tiên, nơi các phần tử được sắp xếp theo mức ưu tiên và phần tử tối ưu (mức ưu tiên cao nhất trong min heap hoặc mức ưu tiên thấp nhất trong max heap) luôn ở đầu hàng đợi. Điều này hữu ích trong việc quản lý và giải quyết các công việc theo ưu tiên khác nhau.

Sắp Xếp Danh Sách: ‘heapq’ cung cấp hàm ‘nlargest()’ và ‘nsmallest()’ để lấy n phần tử lớn nhất hoặc nhỏ nhất từ danh sách, sử dụng cơ chế heap để tối ưu hóa thời gian thực hiện.

Giải Quyết Bài Toán Liên Quan Đến Sắp Xếp: Heap thường được sử dụng để giải quyết các bài toán liên quan đến sắp xếp như tìm k phần tử lớn nhất hoặc nhỏ nhất, sắp xếp dãy dữ liệu theo thứ tự tăng hoặc giảm dần, và nhiều thuật toán tối ưu khác.

Thao Tác Với Dãy Dữ Liệu Lớn: Heap rất hiệu quả trong việc thao tác với dãy dữ liệu lớn, đặc biệt là khi bạn cần tối ưu hóa thời gian và không muốn sắp xếp toàn bộ danh sách..

**Thư viện Time**

Thư viện time trong Python là một thư viện chuẩn của ngôn ngữ Python, cung cấp các chức năng liên quan đến xử lý thời gian và đồng hồ. Thư viện này cho phép bạn thực hiện các tác vụ như đo thời gian thực hiện mã, tạo độ trễ, đặt thời gian ngưng và nhiều công việc khác liên quan đến thời gian. Dưới đây là một số công dụng chính của thư viện time:

Đo Thời Gian: Thư viện time cho phép bạn đo thời gian thực hiện một phần mã bằng cách ghi lại thời điểm bắt đầu và kết thúc thực hiện mã, sau đó tính toán thời gian trôi qua giữa chúng. Điều này hữu ích để tối ưu hóa mã và tìm hiểu về hiệu suất của ứng dụng.

Ngưng Chương Trình (Sleep): Bạn có thể sử dụng hàm sleep(seconds) để tạm dừng thực thi chương trình trong một khoảng thời gian cố định. Điều này thường được sử dụng trong các tình huống cần đặt độ trễ giữa các tác vụ hoặc để kiểm tra cách ứng dụng của bạn xử lý thời gian.

Thời Gian Hệ Thống: Bạn có thể lấy thời gian hệ thống hiện tại bằng cách sử dụng hàm time(). Thời gian này thường được biểu diễn dưới dạng số nguyên hoặc số thực, thường là số giây kể từ mốc thời gian (epoch time).

Đồng Hồ Epoch: Python sử dụng "epoch time" là một điểm thời gian cố định (thường là 00:00:00 UTC ngày 1 tháng 1 năm 1970) làm tham chiếu cho việc tính toán thời gian. Thư viện time cho phép bạn làm việc với epoch time thông qua hàm time() và gmtime().

Chuyển Đổi Thời Gian: Bạn có thể chuyển đổi giữa các định dạng thời gian khác nhau, ví dụ như chuyển đổi giữa chuỗi thời gian và số nguyên epoch time bằng cách sử dụng hàm strftime() và strftime().

Thư viện time rất hữu ích trong việc xử lý các tác vụ liên quan đến thời gian, theo dõi hiệu suất ứng dụng, đồng bộ hóa thời gian trong các ứng dụng đa tiến trình hoặc đa luồng, và định dạng thời gian cho đầu ra người dùng.

# **CHƯƠNG 2: THUẬT TOÁN ÁP DỤNG ĐỀ TÀI**

## **2.1. Giới thiệu thuật toán AStar: A\***

Thuật toán A\*(A-Star) được phát triển bởi Peter Hart, Nils Nilson và Bertram Raphael vào năm 1968. Trong báo cáo của họ, thuật toán được gọi là thuật toán A, khi sử dụng thuật toán này với một hàm đánh giá heuristic thích hợp sẽ thu được hoạt động tối ưu, do đó mà có tên A\*. Thuật toán A\* đã trở thành một trong những thuật toán phổ biến và mạnh mẽ.

### **2.1.1. Mô tả thuật toán**

Giả sử n là một trạng thái đạt tới ( có đường đi từ trạng thái ban đầu tới đích). Ta xác định được hàm đánh giá: f(n) = g(n) + h(n). Trong đó:

g(n) là giá thành đường đi từ điểm xuất phát tới nút n

h(n) là hàm heuristic ước lượng giá thành đường đi từ n tới đích

f(n) là chi phí tổng thể ước lượng của đường đi từ điểm xuất phát đến đích

Một ước lượng heuristic h(n) được xem là chấp nhận được nếu với mọi nút n: 0 <= h(n) <= h\*(n). Trong đó, h\*(n) là chi phí thật(thực tế) để đi từ nút n đến đích.

### **2.1.2. Tính chất của thuật toán**

A\* là thuật toán đầy đủ (complete) theo nghĩa rằng nó sẽ luôn luôn tìm thấy một lời giải nếu bài toán có lời giải

A\* có tính chất thu nạp được (hay tối ưu) nếu sử dụng một tập đóng. Nếu không sử dụng tập đóng thì hàm h phải có tính chất đơn điệu (hay nhất quán) thì A\* mới có tính chất tối ưu. Nghĩa là nó không bao giờ đánh giá chi phí đi từ một nút tới một nút kề nó cao hơn chi phí thực.

A\* còn có tính chất hiệu quả một cách tối ưu (optimally efficient) với mọi hàm heuristic h, có nghĩa là không có thuật toán nào cũng sử dụng hàm heuristic đó mà chỉ phải mở rộng ít nút hơn A\*, trừ khi có một số lời giải chưa đầy đủ mà tại đó h dự đoán chính xác chi phí của đường đi tối ưu.

Thuật toán Dijkstra là một trường hợp đặc biệt của A\* trong đó đánh giá heuristic là một hàm hằng h(x) = 0 với mọi x.

### **2.1.3. Thuật toán A\***

A = (Q,S,G,P, c,h)

Q: Không giá trạng thái; S: Trạng thái bắt đầu; G: Đích; P:hành động,c: giá, h: heuristic)

Đầu vào: Bài toán tìm kiếm, hàm heuristic h

Đầu ra: Đường tới nút đích

Khởi tạo: tập các nút biên (nút mở) O = S

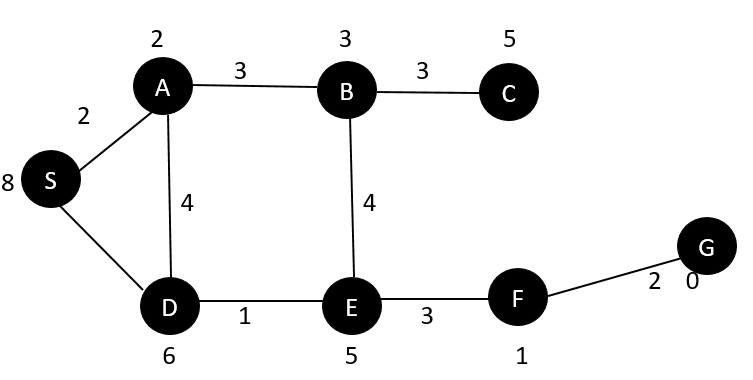
**while** (O **!= ∅** ) do

1. Lấy nút n có f(n) là nhỏ nhất ra khỏi O.
2. **if** n∈ G, return (đường đi tới n)
3. Với mọi m∈ P(n)
4. g(m) = g(n) + c(n, m)
5. f(m) = g(m) + h(m)
6. Thêm m vào O cùng với giá trị f(m)

**return** Không tìm được đường đi

### **2.1.4. Ví dụ về thuật toán A\***

Tìm đường đi ngắn nhất từ S -> G và chi phí bằng thuật toán A\*

****

Hình 2.1. Ví dụ về thuật toán A\*

**Lời giải**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Bước lập | U | Kề U | g(v)=g(u)+k(u,v) | f(v)=g(v)+h(v) | Open |
| 0 |  |  |  |  | S8 |
| 1 | S8 | A2 ,D6 | g(A)=0+2=2  g(D)=0+3=3 | f(A)=2+2=4  f(D)=3+6=9 | A4 ,D9 |
| 2 | A4 | D6 ,B3 | g(D)=2+4=6  g(B)=2+3=5 | f(D)=6+6=12  f(B)=5+3=8 | B8 ,D9 |
| 3 | B8 | C5 ,E5 | g(C)=5+3=8  g(E)=5+4=9 | f(C)=8+5=13  f(E)=9+5=14 | D9 ,C13 ,E14 |
| 4 | D9 | E5 | g(E)=3+1=4 | f(E)=4+5=9 | E9 ,C13 |
| 5 | E9 | F1 | g(F)=4+3=7 | f(F)=7+1=8 | F8 ,C13 |
| 6 | F8 | G0 | g(G)=7+2=9 | f(G)=9+0=9 | G9 ,C13 |
| 7 | Đích:G9 |  |  |  | C13 |

## **2.2. Áp dụng thuật toán vào đề tài**

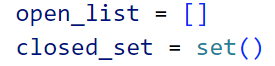
Hàm a\_star nhận vào hai đối số: start :là điểm bắt đầu và goal :là điểm đích.

Đây là một hàm thực hiện thuật toán tìm đường A\* để tìm đường ngắn nhất giữa hai điểm trên một lưới.



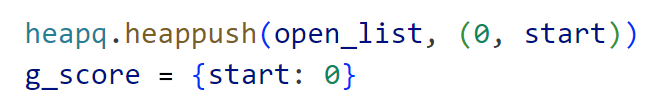
Ở đây, chúng ta khởi tạo open list :là một danh sách ưu tiên (priority queue) để theo dõi các ô chưa xem xét.

Closed set :là một tập hợp để lưu trữ các ô đã xem xét.



Chúng ta đẩy điểm bắt đầu vào open list với chi phí tới nó là 0.

g score :là một từ điển để lưu trữ chi phí tới các điểm.



Chúng ta bắt đầu một vòng lặp chạy cho đến khi open list trống.



Ở đây, chúng ta lấy ra điểm có giá trị dự kiến tổng chi phí thấp nhất từ open list.



Kiểm tra xem điểm hiện tại có phải là điểm đích không. Nếu có, ta quay lại từ điểm đích về điểm bắt đầu để xây dựng và trả về đường đi

1. Kiểm tra điểm hiện tại có phải là điểm đích không:

if current == goal: kiểm tra xem điểm hiện tại có phải là điểm đích không.

2. Xây dựng đường đi ngược từ điểm đích về điểm bắt đầu:

path = []: Khởi tạo một danh sách để lưu trữ đường đi.

while current != start:: Vòng lặp chạy cho đến khi đạt được điểm bắt đầu.

path.append(current): Thêm điểm hiện tại vào danh sách đường đi.

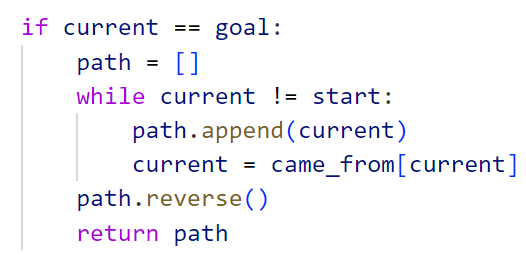
current = came from[current]: Di chuyển đến điểm trước đó theo thông tin trong came\_from.

3. Đảo ngược đường đi:

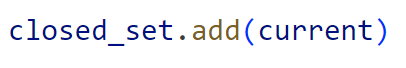
path.reverse(): Đảo ngược danh sách để có thứ tự từ điểm bắt đầu đến điểm đích.

4. Trả về đường đi:

return path: Trả về danh sách đường đi từ điểm bắt đầu đến điểm đích.



Đánh dấu điểm hiện tại là đã xem xét.



Duyệt qua 4 hướng lân cận: lên, xuống, trái, phải.



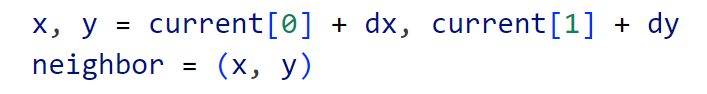
1. [(0, -1), (0, 1), (-1, 0), (1, 0)]: Đây là một danh sách chứa 4 cặp giá trị, mỗi cặp biểu diễn một hướng lân cận trong không gian 2D. Cụ thể, (0, -1) biểu diễn di chuyển lên, (0, 1) là di chuyển xuống, (-1, 0) là di chuyển sang trái, và (1, 0) là di chuyển sang phải.

2. for dx, dy in [(0, -1), (0, 1), (-1, 0), (1, 0)]: Dòng này tạo một vòng lặp for để duyệt qua mỗi cặp giá trị trong danh sách. dx và dy là biến lặp, và trong mỗi lần lặp, chúng sẽ nhận giá trị của cặp tương ứng trong danh sách.

3. Tính toán tọa độ mới (x, y): Dòng code x, y = current[0] + dx, current[1] + dy sử dụng giá trị dx và dy để tính toán tọa độ mới (x, y) dựa trên tọa độ của điểm hiện tại (current).

4. Sử dụng tọa độ mới để xử lý ô lân cận: Dòng code neighbor = (x, y) tạo một tuple neighbor chứa tọa độ mới, biểu diễn ô lân cận của điểm hiện tại. Các bước tiếp theo của vòng lặp sẽ xử lý ô lân cận này.

Tính toán tọa độ của ô lân cận.



Đây là cách tính toán tọa độ mới (x, y) của ô lân cận dựa trên tọa độ của điểm hiện tại (current) và một cặp giá trị (dx, dy). Dưới đây là chi tiết từng dòng:

1. x, y = current[0] + dx, current[1] + dy:

Current[0] là tọa độ x của điểm hiện tại.

Current[1] là tọa độ y của điểm hiện tại.

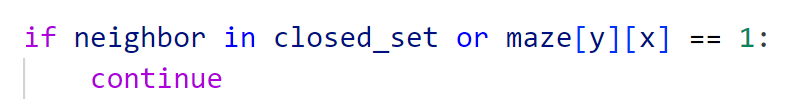
dx và dy là cặp giá trị từ danh sách [(0, -1), (0, 1), (-1, 0), (1, 0)].

Dòng này thực hiện phép cộng để di chuyển tọa độ của điểm hiện tại theo giá trị của dx và dy.

Kết quả là tọa độ mới của ô lân cận được lưu vào x và y.

2. Neighbor = (x, y): Sau khi tính toán tọa độ mới, chúng ta tạo một tuple neighbor chứa tọa độ mới đó. Neighbor biểu diễn ô lân cận của điểm hiện tại trong mê cung. Điều này giúp chúng ta dễ dàng xác định tọa độ của ô lân cận dựa trên tọa độ của điểm hiện tại và hướng di chuyển được xác định bởi dx và dy. Điều này làm cho quá trình kiểm tra và xử lý ô lân cận trở nên linh hoạt trong vòng lặp chính của thuật toán A\*.

Kiểm tra xem ô lân cận đã được xem xét hay có thể đi qua không.

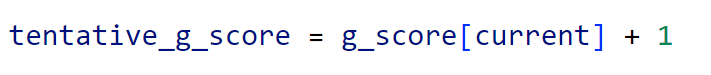


1. neighbor in closed\_set: Neighbor in closed\_set kiểm tra xem ô lân cận đã được xem xét trước đó hay chưa. Nếu neighbor đã nằm trong closed\_set, có nghĩa là nó đã được kiểm tra và không cần phải xem xét lại. Do đó, chúng ta sử dụng continue để bỏ qua phần còn lại của vòng lặp và chuyển sang ô lân cận khác.

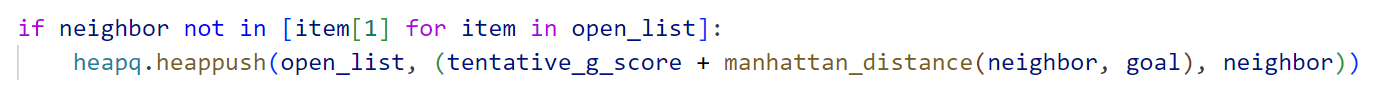
2. maze[y][x] == 1: maze[y][x] == 1 kiểm tra xem ô lân cận có thể đi vào được không. Trong ngữ cảnh của thuật toán A\*, maze có thể là một ma trận biểu diễn mê cung, trong đó giá trị 1 có thể biểu thị các ô không thể đi vào.

Nếu ô lân cận không thể đi vào được (maze[y][x] == 1), chúng ta cũng sử dụng continue để bỏ qua phần còn lại của vòng lặp và xem xét ô lân cận tiếp theo.

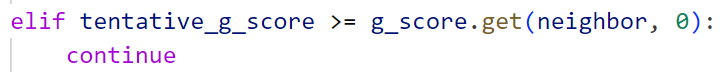
Tính toán chi phí tạm thời từ điểm bắt đầu tới ô lân cận.



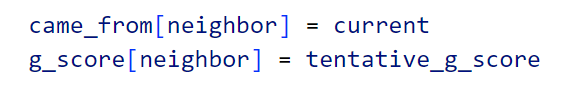
Nếu ô lân cận chưa được xem xét, đẩy nó vào open list với giá trị dự kiến tổng chi phí.



Nếu ô lân cận đã được xem xét và chi phí tạm thời lớn hơn chi phí đã ghi nhớ, bỏ qua.



Ghi nhớ điểm hiện tại là điểm trước của ô lân cận và cập nhật chi phí đã biết tới ô lân cận.



Nếu sau khi lặp mà không tìm thấy đường đi, trả về None.



# **CHƯƠNG 3:**

# **XÂY DỰNG VÀ TRIỂN KHAI ỨNG DỤNG**

## **3.1. Sản phẩm nghiên cứu**

Sản phẩm nghiên cứu trò chơi tìm khóa trong mê cung mang đến một trải nghiệm chơi khá là thích thú, khi mà nhân vật chính sẽ tự động tìm đường để thu thập chìa khoá mà không có sự điều khiển hoặc can thiệp từ người chơi. Bằng cách tích hợp giải thuật tìm đường A\* vào trong mã nguồn của trò chơi, sản phẩm như là một minh chứng cho biết khả năng ứng dụng linh hoạt của trí tuệ nhân tạo trong giải quyết vấn đề.

Trong trò chơi, môi trường được biểu diễn dưới dạng bản đồ lưới, và nhân vật di chuyển thông qua các ô để thu thập chìa khóa. Sử dụng thuật toán A\* thông minh, nhân vật có khả năng tự động đưa ra quyết định về đường đi tối ưu nhất đến chìa khóa, tạo nên một trải nghiệm chơi độc đáo và thú vị.

Sản phẩm không chỉ mang lại trải nghiệm giải đố sáng tạo mà còn mở ra tiềm năng trong việc áp dụng các giải thuật tìm đường tự động vào nhiều lĩnh vực khác nhau, như tự động hóa và trí tuệ nhân tạo. Điều này thể hiện sức mạnh của nghiên cứu trong việc giải quyết các thách thức thực tế và tạo ra những ứng dụng đa dạng.

### **3.1.1. Cài đặt các thư viện cần thiết**

Để triển khai trò chơi tìm khóa trong mê cung trên, Cần phải tích hợp một số thư viện quan trọng để hỗ trợ việc lập trình và thực hiện giải thuật tìm đường tự động A\*. Thứ nhất sử dụng thư viện pgzrun để tạo ra môi trường đồ họa và quản lý vòng lặp chính của trò chơi. Thư viện này cung cấp các tiện ích cần thiết để vẽ đồ họa, xử lý sự kiện và cập nhật trạng thái của trò chơi.

Để cài đặt thư viện pgzrun có nhiều cách, dưới đây là một vài cách đơn giản:

Bước 1: Mở Terminal hoặc Command Prompt: Trước hết, hãy mở terminal hoặc command prompt trên máy tính

Bước 2: Cài đặt “Pgzrun”: Sử dụng lệnh sau để cài đặt pgzrun “pip install pgzero”. Lệnh này sẽ tự động tải về và cài đặt pgzrun cùng với các phụ thuộc của nó từ Python Package Index (PyPI).

Bước 3: Kiểm tra việc cài đặt: Kiểm tra xem pgzrun đã được cài đặt thành công chưa bằng cách kiểm tra phiên bản. Sử dụng lệnh sau để kiểm tra phiên bản “pgzrun --version”. Nếu lệnh này không trả về lỗi và hiển thị phiên bản của pgzrun, điều đó có nghĩa là bạn đã cài đặt thành công.

Để xử lý đồ họa ta sử dụng thư viện pgzrun, do trò chơi sử dụng đồ hoạ 2D mà thư viện pygame được tích hợp sẵn trong pgzrun. pygame là một thư viện mạnh mẽ cho phát triển trò chơi và đồ họa, cung cấp các chức năng về vẽ và xử lý sự kiện.

Đối với việc triển khai giải thuật tìm đường A\*, bổ sung thêm thư viện heapq. Thư viện heapq được sử dụng để quản lý hàng đợi ưu tiên trong quá trình tìm kiếm đường đi trong thuật toán A\*. Hàng đợi ưu tiên này được sử dụng để lựa chọn nút tiếp theo để kiểm tra dựa trên chi phí tạm thời và ước lượng khoảng cách còn lại.

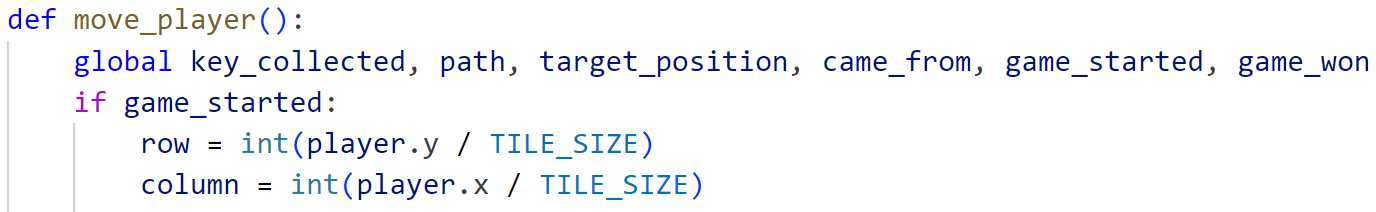
Thư viện time được sử dụng để thêm độ trễ vào quá trình di chuyển của nhân vật, tạo ra một trải nghiệm trò chơi động và thú vị.

Tất cả những thư viện này đều là những công cụ hữu ích giúp chúng tôi dễ dàng thực hiện và hiển thị sản phẩm nghiên cứu một cách hiệu quả. Sự tích hợp thông minh giữa các thư viện này không chỉ tạo ra một sản phẩm mạnh mẽ mà còn thể hiện sự linh hoạt và sáng tạo trong quá trình phát triển.

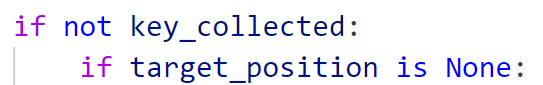
### **3.1.2 Đưa thuật toán vào các sự kiện của nhân vật.**

Để nhân vật có thể tư duy chuyển và tìm đường đến vị trí chìa khoá, cần có các thuật toán về xác định vị trí của nhân vật, điều kiện di chuyển đúng của nhân vật. Dưới đây là cách mà nhân vật di chuyển được đến vị trí chìa khoá.

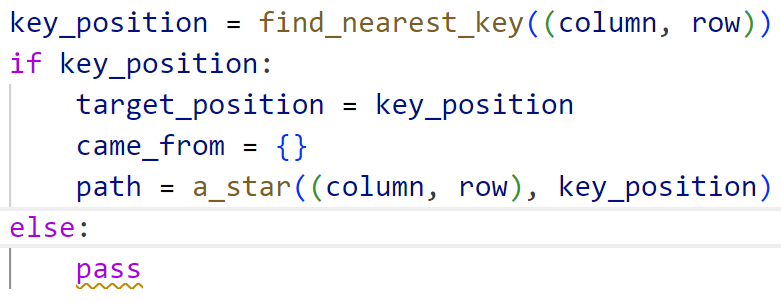
Xác định vị trí hiện tại: để xác định hàng và cột của nhân vật trong mảng maze dựa trên vị trí hiện tại của nhân vật.



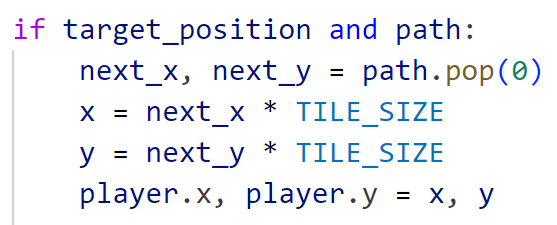
Kiểm tra việc thu thập chìa khoá: Nhận biết xem nhân vật đã thu thập chìa khóa hay chưa. Nếu chưa, thực hiện các bước để tìm và thu thập chìa khóa.



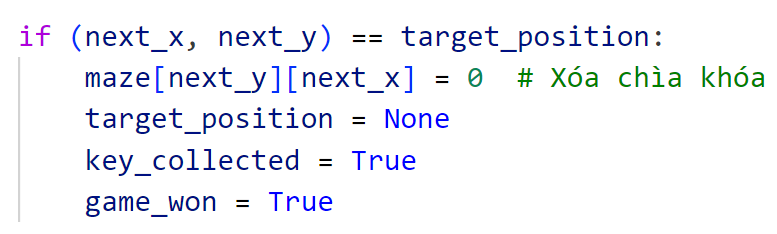
Tìm đường đi đến chìa khoá: Tìm vị trí của chìa khóa gần nhất sử dụng hàm “find\_nearest\_key” và sau đó tính toán đường đi tới chìa khóa bằng thuật toán A\*.



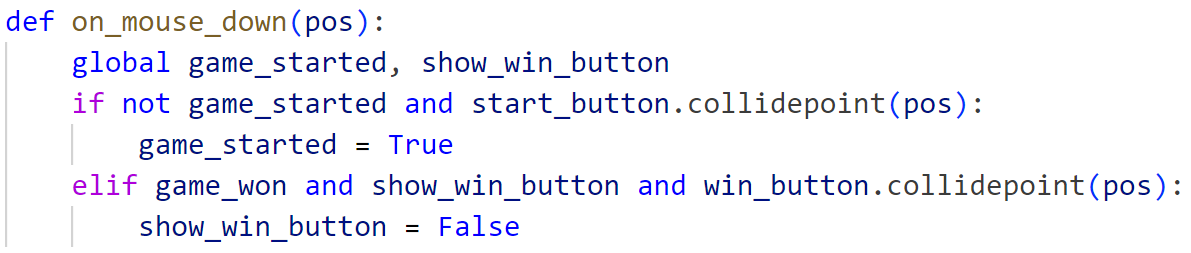
Di chuyển theo đường đi: Nếu có đường đi (“path” không rỗng) và mục tiêu di chuyển chưa được đạt đến (“(next\_x, next\_y) != target\_position”), nhân vật sẽ di chuyển đến vị trí tiếp theo trên đường đi.



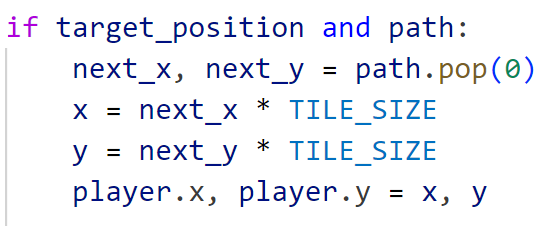
Xoá chìa khoá nếu đã thu thập: Nếu nhân vật đến được chìa khóa (“(next\_x, next\_y) == target\_position”), thì xóa chìa khóa khỏi mảng “maze” và đặt cờ “key\_collected” và “game\_won” là True.



Điều kiện di chuyển đúng: Di chuyển nhân vật chỉ xảy ra khi trò chơi đã bắt đầu (“game\_started == True”).



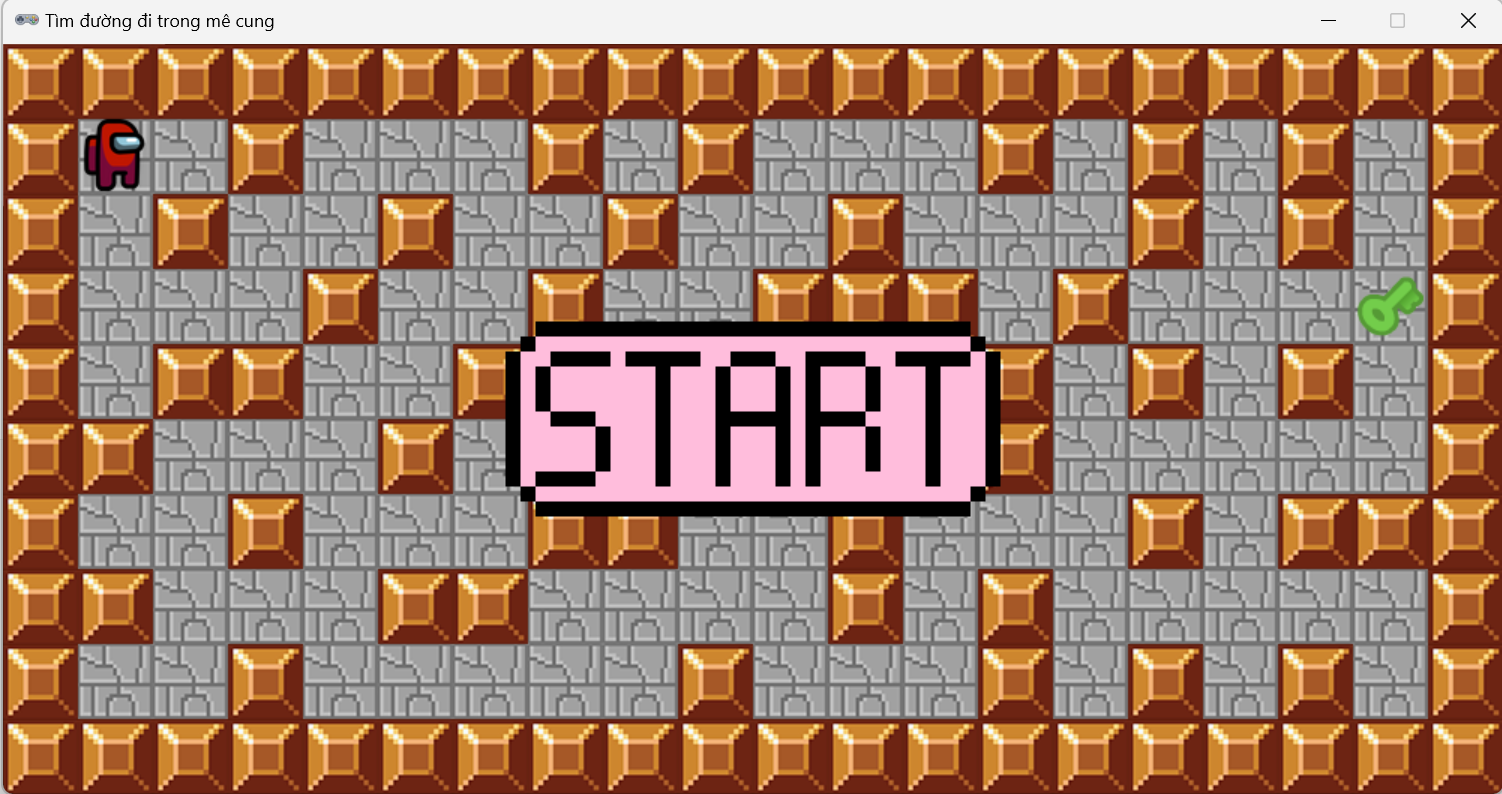
Kiểm tra điều kiện, điều kiện đúng cho di chuyển: Nhân vật chỉ di chuyển khi đang trên đường đi và còn đường để đi (path không rỗng).Mọi hành động đều được kiểm tra và thực hiện theo thứ tự logic, đảm bảo nhân vật di chuyển đúng và các điều kiện được xác định đúng.

****

### **3.1.3. Giao diện trò chơi**

Giao diện trò chơi được thiết kế đơn giản và thân thiện, mang lại trải nghiệm người chơi thú vị và dễ tiếp cận. Màn hình chính của trò chơi hiển thị một bản đồ mê cung với các ô vuông, mỗi ô đại diện cho một phần của môi trường trò chơi. Nhân vật của người chơi được biểu diễn bằng hình ảnh ngộ nghĩnh, trong trường hợp này, là hình ảnh "among\_us". Bạn có thể thấy nhân vật di chuyển linh hoạt trên bản đồ mê cung khi người chơi tham gia vào trò chơi.

Ở giữa màn hình, có một nút "Start" có hình ảnh thích hợp, chờ đợi người chơi nhấp vào để bắt đầu cuộc phiêu lưu. Nút "Start" này giúp tạo ra một trải nghiệm tương tác và khám phá, làm cho người chơi trở nên tích cực và sẵn sàng tham gia vào trò chơi.



Hình 3.1. Hình ảnh giao diện bắt đầu trò chơi

Khi người chơi di chuyển nhân vật, giao diện hiển thị một loạt các ô vuông, mỗi ô có một ý nghĩa khác nhau. Các ô trống thể hiện các khu vực mà nhân vật có thể đi qua, trong khi các ô tường đại diện cho các phần của mê cung không thể vượt qua. Các ô có chìa khóa hoặc mục tiêu được biểu thị bằng biểu tượng phù hợp, tạo ra một cảm giác hứng thú và mong đợi.

Sau khi người chơi thu thập chìa khóa và đạt được mục tiêu, chữ “Victory” xuất hiện, mời gọi người chơi nhấn vào để hoàn thành và kết thúc trò chơi. Chữ “Victory” này giúp tăng tính tương tác và hứng thú cho người chơi, tạo nên một trải nghiệm hoàn chỉnh và thỏa mãn khi hoàn thành mục tiêu của trò chơi.

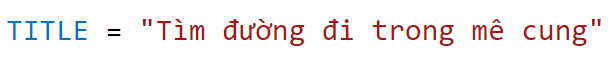


Hình 3.2. Hình ảnh giao diện kết thúc trò chơi

Tổng thể, giao diện trò chơi được thiết kế sao cho đơn giản, rõ ràng và thân thiện với người chơi, giúp họ dễ dàng tham gia và trải nghiệm một cuộc phiêu lưu thú vị trong mê cung.

### **3.1.4 Thiết kế giao diện**

Đặt tiêu đề cho trò chơi: Trong trường hợp này, tiêu đề được đặt là "Tìm đường đi trong mê cung ". Điều này sẽ hiển thị ở phần đầu của cửa sổ trò chơi hoặc cửa sổ trình duyệt nơi trò chơi được chạy. Thông qua biến `TITLE`, bạn có thể đặt tên cho trò chơi hoặc cung cấp thông tin giới thiệu về nội dung của nó.

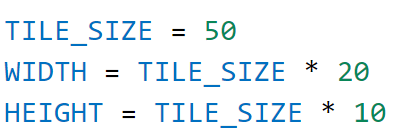


Cài đặt kích thước màn hình: Xác định không gian màn hình và kích thước của mỗi ô, làm cơ sở cho việc vẽ và xử lý vị trí của các đối tượng trong trò chơi. Mỗi ô có kích thước là 50x50 pixels, và màn hình có kích thước là 1000x500 pixels. Đoạn mã trên xác định kích thước của màn hình trò chơi. Cụ thể:

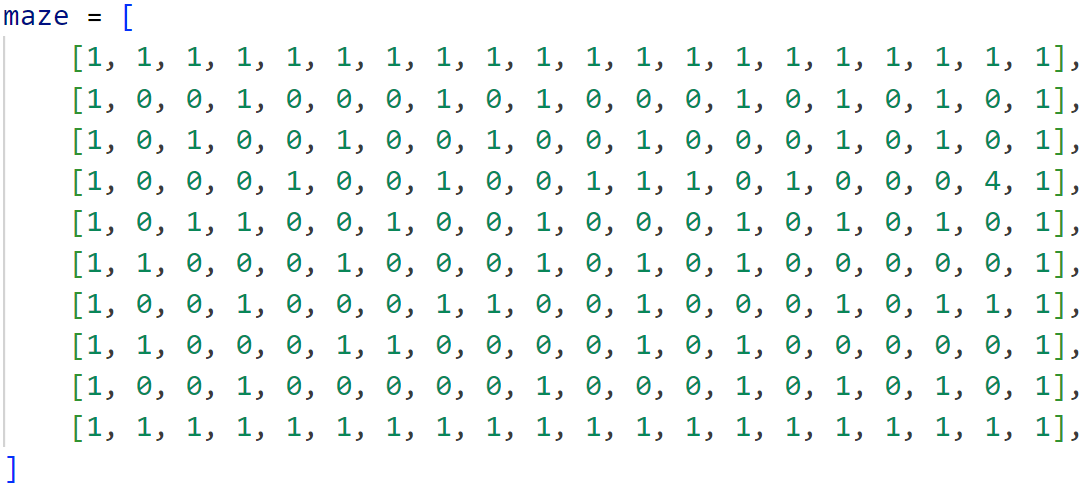
TILE\_SIZE: Kích thước của mỗi ô trong ma trận (50 pixels).

WIDTH: Tổng chiều rộng của màn hình (20 ô theo chiều ngang).

HEIGHT: Tổng chiều cao của màn hình (10 ô theo chiều dọc).



Mảng mê cung: Bài toán ma trận trong trò chơi này đã được giải quyết thông qua việc sử dụng một mảng hai chiều để biểu diễn bản đồ mê cung. Mỗi ô trong mảng đại diện cho một phần tử của môi trường trò chơi. Sự lựa chọn các giá trị trong mảng, như số 1 đại diện cho ô tường, số 0 đại diện cho ô trống mà nhân vật có thể di chuyển được , số 4 đại diện cho ô chứa chìa khóa - mục tiêu mà nhân vật , giúp định nghĩa chi tiết từng phần của mê cung và các yếu tố tương tác trong trò chơi.



Hình ảnh nhân vật: Thiết kế đồ hoạ của nhân vật được thực hiện bằng cách sử dụng hình ảnh "among\_us". Hình ảnh này được đặt ở vị trí góc trái trên của ô vuông, với anchor point được thiết lập một cách chính xác để định vị nhân vật đúng giữa ô vuông. Điều này tạo ra sự mượt mà khi nhân vật di chuyển trong mê cung. Nhân vật được đặt ở vị trí ban đầu (1, 1) trong mê cung.



### **3.1.5. Các bước khởi chạy chương trình**

Trò chơi “Tìm đường đi trong mê cung” là một ứng dụng thú vị được xây dựng bằng Python và Pygame Zero. Để khởi chạy và trải nghiệm trò chơi này, người chơi cần thực hiện một số bước quan trọng

**Bước 1: Cài đặt môi trường phát triển.** Trước tiên, để chơi trò chơi “Tìm đường đi trong mê cung”, người chơi cần phải có mô trường phát triển Python trên máy tính của họ. Điều này đòi hỏi cài đặt Python và Pygame Zero. Sau khi đã cài đặt chúng, người chơi đã sẵn sàng tiến hành các bước tiếp theo

**Bước 2: Tạo thư mục và chuẩn bị hình ảnh**. Trò chơi cần sử dụng nhiều hình ảnh khác nhau, bao gồm:

Hình ảnh cho nhân vật: Đây là hình ảnh đại diện cho nhân vật trong trò chơi (among\_us.png). Đảm bảo rằng hình ảnh nhân vật có kích thước phù hợp và định dạng hình ảnh (thường là PNG hoặc JPG).

Hình ảnh cho các ô lưới: Trong trò chơi, có các loại ô lưới khác nhau như "empty" (ô chữ chữ “Victory”“Victory”trống), "wall" (tường), "goal" (mục tiêu), "spikes" (cạm) và "key" (chìa khóa). Bạn cần chuẩn bị hình ảnh cho mỗi loại ô lưới này. Ví dụ: "empty.png," "wall.png," "goal.png," "spikes.png," và "key.png." Đảm bảo rằng mỗi hình ảnh có kích thước và định dạng phù hợp.

Hình ảnh cho các nút "Start" và "Win Game" : Trò chơi có hai nút quan trọng là "Start" (Bắt đầu) và "Win Game" (Chiến thắng trò chơi). Hình ảnh cho các nút này là "start\_button.png" và "win\_button.png." Đảm bảo rằng chúng phù hợp với giao diện người dùng và có kích thước hợp lý.

**Bước 3: Viết mã trò chơi.** Viết mã trò chơi là một phần quan trọng trong quá trình phát triển trò chơi. Sử dụng Python và thư viện Pygame để viết mã trò chơi. Đảm bảo rằng bạn đã viết mã để tạo mê cung, di chuyển nhân vật, tìm đường đi và kiểm tra kết thúc trò chơi

Khởi tạo : Phần này bắt đầu với việc khởi tạo trạng thái ban đầu của trò chơi, bao gồm tạo các biến và thiết lập các giá trị ban đầu. Điều này có thể bao gồm việc tạo nhân vật chính, đối tượng trò chơi, và thiết lập các thông số như điểm số.

Vòng lặp chính (main loop): Trò chơi chạy trong một vòng lặp vô hạn, cập nhật và vẽ trạng thái của trò chơi lên môi trường đồ họa. Ta có một vòng lặp while hoặc vòng lặp tương tự để kiểm tra và xử lý sự kiện, cập nhật trạng thái trò chơi và vẽ trạng thái lên màn hình.

Xử lý sự kiện: Trong phần này, ta sẽ kiểm tra các sự kiện người dùng. Điều này bao gồm việc xử lý sự kiện như bấm nút bàn phím, di chuyển chuột, bấm nút cửa sổ (nếu có), và bất kỳ tương tác nào từ người chơi. Xử lý sự kiện sẽ dẫn đến các hành động cụ thể trong trò chơi.

Cập nhật trạng thái trò chơi: Dựa trên sự kiện và tương tác từ người chơi, bạn sẽ cập nhật trạng thái trò chơi. Điều này bao gồm di chuyển nhân vật, kiểm tra va chạm, tính điểm, và thay đổi các biến để điều chỉnh trạng thái trò chơi.

Vẽ đồ họa: Sử dụng tích hợp đồ họa hoặc thư viện đồ họa của ngôn ngữ lập trình để vẽ trạng thái của trò chơi lên màn hình. Điều này bao gồm vẽ nhân vật, đối tượng, môi trường và giao diện người dùng. Bạn sẽ thường cần xác định tọa độ và kích thước của các đối tượng để vẽ chúng lên màn hình.

Kiểm tra điều kiện kết thúc: Trong vòng lặp chính, bạn cần kiểm tra điều kiện kết thúc trò chơi. Điều này có thể bao gồm kiểm tra xem người chơi đã chiến thắng, thua cuộc hoặc bất kỳ điều kiện nào xác định kết thúc trò chơi.

Kết Thúc Vòng Lặp: Khi một điều kiện kết thúc trò chơi xảy ra, bạn sẽ thoát khỏi vòng lặp chính và thực hiện các tác vụ kết thúc như hiển thị thông báo về kết quả, điểm số, hoặc cơ hội chơi lại.

Kiểm Tra Lỗi và Sửa Lỗi: Kiểm tra và sửa lỗi trong quá trình phát triển và kiểm tra trò chơi. Kiểm tra các tình huống đặc biệt, xử lý các lỗi tiềm năng và đảm bảo trò chơi hoạt động một cách ổn định.

Thử Nghiệm và Hiệu Chuẩn: Thử nghiệm trò chơi trên nhiều nền tảng và thiết bị để đảm bảo tính ổn định, độ tin cậy và hiệu suất tốt.

**Bước 4: Khởi chạy trò chơi**. Sau khi hoàn thành xong việc viết mã và thư mục trò chơi đã được xác định, người chơi có thể khởi động trò chơi bằng cách mở một bảng điều khiển hoặc dòng lệnh trên máy tính. Điều này đòi hỏi sử dụng lệnh để điều hướng đến thư mục của trò chơi bằng dòng lệnh. Sau đó, người chơi có thể chạy trò chơi bằng lệnh sau: “python GameAmong.py”. Lệnh này sẽ khởi động trò chơi và màn hình trò chơi sẽ hiển thị.

**Bước 5: Tham Gia Trò Chơi.** Sau khi đã mở màn hình trò chơi, người chơi sẽ thấy một mê cung và nhân vật hiện lên trên màn hình. Người chơi có thể sử dụng bàn phím hoặc chuột để điều khiển nhân vật trong mê cung. Nhiệm vụ là tìm chìa khóa và đến mục tiêu mà chìa khóa mở cửa.

**Bước 6: Tận hưởng trò chơi.** Khi đã tìm thấy chìa khóa và hoàn thành nhiệm vụ, người chơi có thể tận hưởng trò chơi " Tìm Khóa Nhà." Trò chơi không chỉ mang lại niềm vui và giải trí mà còn giúp rèn luyện kỹ năng tìm đường thông qua việc sử dụng thuật toán tìm kiếm A\*.

Các bước khởi chạy chương trình trò chơi "Tìm đường đi trong mê cung" tương đối đơn giản, đòi hỏi môi trường phát triển Python, tải trò chơi, khởi động

**3.2. Kết luận**

Trên đây là tổng quan về sản phẩm bao gồm những thư viện, thuật toán được sử dụng, cùng với giao diện và hướng dẫn các bước khởi chạy chương trình. Qua đó cũng thấy được sản phẩm có những ưu và nhược điểm như sau:

**Ưu điểm:** Sản phẩm đã được thiết kế một giao diện đơn giản nhưng thân thiện và mang lại trải nghiệm thú vị, tạo ra sự kết nối giữa người chơi và trò chơi. Áp dụng được những thuật toán được thiết kế để tìm kiếm đường đi tối ưu giữa hai điểm, đảm bảo rằng nó sẽ tìm ra được lời giải ngắn nhất, tiết kiệm được tài nguyên và thời gian tính toán. Cùng với các thư viện phù hợp giúp giảm thời gian và công sức cần thiết để phát triển tính năng tìm đường đi trong trò chơi và tạo ra những yếu tố độc đáo, sáng tạo trong trò chơi, giúp trò chơi nổi bật và thu hút người chơi. Trò chơi có thể thúc đẩy khả năng giải quyết vấn và tư duy logic của người chơi qua việc người chơi phải giải quyết các vấn đề như tìm đường đi và tránh va chạm để đạt được mục tiêu.

**Nhược điểm:** sản phẩm cũng có những mặt hạn chế như chưa tạo ra được nhiều chướng ngại vận, cũng như độ phức tạp của trò chơi. Chưa tạo ra được các mức độ khó, dễ khác nhau để phù hợp sở thích của từng người chơi. Mê cung còn đơn giản, dễ dàng đoán được cách ra khỏi mê cung, chưa cho phép người chơi tương tác được với môi trường xung quanh mê cung làm giảm tính tương tác của trò chơi, dẫn đến thiếu hấp dẫn. Sản phẩm cần được tạo thêm nhiều giao diện phong phú, bắt mắt nhằm thu hút người chơi và tăng sự hấp dẫn, của trò chơi. Cũng như tạo ra nhiều dạng mê cung có kích thước lớn nhỏ khác nhau để tăng thử thách và phục vụ người chơi. Trò chơi có thể bị dẫn đến quá trình lặp đi lặp lại khi người chơi phải tiếp tục thử nhiều lần để tìm đúng đường. Điều này làm mất hứng thú và tạo cảm giác mất thời gian cho người chơi.

Qua đây sản phẩm sẽ tiếp tục được chỉnh sửa, nâng cấp để tạo ra một trò chơi hoàn thiện, tối ưu nhất tăng trải nghiệm thú vị và thách thức, làm cho người chơi cảm thấy hứng thú và quyết tâm giải quyết các mê cung.

# **TÀI LIỆU THAM KHẢO**

1. PGS.TS Từ Minh Phương, *Nhập môn Trí tuệ nhân tạo*, Học Viện Công Nghệ Bưu Chính Viễn Thông, 2014
2. Stuart Rusell & Peter Norvig, *Artificial Intelligence: A modern approach*, 2009
3. Đinh Mạnh Tường, *Giáo trình trí tuệ nhân tạo*. ĐHQGHN, 2006
4. Nguyễn Thanh Thủy, *Trí tuệ nhân tạo: các phương pháp giải quyết vấn đề và xử lý tri thức*, NXB GD 1995
5. <https://tenten.vn/tin-tuc/ung-dung-cua-tri-tue-nhan-tao/>
6. <https://viblo.asia/p/a-pathfinding-nwmkyEjlkoW>
7. <https://quantrimang.com/hoc/phat-trien-game-de-dang-hon-voi-pygame-zero-194108>