TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

KHOA KHOA HỌC MÁY TÍNH



BÁO CÁO ĐỒ ÁN MÔN TRÍ TUỆ NHÂN TẠO

BẮN XE TĂNG

Lớp: CS106.I22

Giảng viên hướng dẫn: HUỲNH THỊ THANH THƯƠNG

Nhóm thực hiện:

1. Võ Thành Cường 16521497
2. Nguyễn Phạm Long Duy 16520299
3. Nguyễn Trần Quốc Toàn 16521257
4. Phạm Ngọc Phúc Thuần 16521206

*Tp. Hồ Chí Minh, tháng 07 năm 2018*

**MỤC LỤC**

[LỜI NÓI ĐẦU 1](#_Toc518410989)

[HỢP ĐỒNG NHÓM 1](#_Toc518410990)

[BẢNG PHÂN CÔNG CÔNG VIỆC 1](#_Toc518410991)

[CHƯƠNG 1: GIỚI THIỆU ĐỒ ÁN 1](#_Toc518410992)

[1. LỊCH SỬ TRÒ CHƠI: 1](#_Toc518410993)

[2. MÔ TẢ TRÒ CHƠI: 1](#_Toc518410994)

[CHƯƠNG 2: CƠ SỞ LÝ THUYẾT 1](#_Toc518410995)

[1. NGÔN NGỮ LẬP TRÌNH C#: 1](#_Toc518410996)

[2. LÝ THUYẾT ÁP DỤNG: 1](#_Toc518410997)

[CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ 1](#_Toc518410998)

[1. PHÂN TÍCH BÀI TOÁN: 1](#_Toc518410999)

[2. THIẾT KẾ CẤU TRÚC DỮ LIỆU VÀ TRẠNG THÁI: 1](#_Toc518411000)

[3. CÁC VẤN ĐỀ VÀ THUẬT GIẢI: 1](#_Toc518411001)

[CHƯƠNG 4: ỨNG DỤNG 1](#_Toc518411002)

[1. GIỚI THIỆU CHƯƠNG TRÌNH ỨNG DỤNG: 1](#_Toc518411003)

[CHƯƠNG 5: KẾT LUẬN 1](#_Toc518411004)

[1. KẾT QUẢ ĐẠT ĐƯỢC: 1](#_Toc518411005)

[2. HẠN CHẾ: 1](#_Toc518411006)

[3. HƯỚNG PHÁT TRIỂN: 1](#_Toc518411007)

[TÀI LIỆU THAM KHẢO 1](#_Toc518411008)

[LỜI CẢM ƠN 1](#_Toc518411009)

# LỜI NÓI ĐẦU

Cách mạng công nghiệp lần thứ tư (hay còn gọi là *Cách mạng công nghiệp 4.0*) đánh dấu sự phát triển mạnh mẽ của kỷ nguyên khoa học – kỹ thuật – công nghệ ngày nay. Cuộc cách mạng phát triển ba trụ cột chính là kỹ thuật số, công nghệ sinh học, vật lý. Trong đó, Trí tuệ nhân tạo (AI) đang nổi lên và trở thành tâm điểm của cuộc cách mạng lần này.

Trí tuệ nhân tạo tạo từng được biết đến qua cuộc đua giữa con người và trí tuệ nhân tạo trong các trò chơi như cờ vua, cờ vây, game online, hay thậm chí cả các cuộc đố vui. Nếu như trước đây chúng ta chỉ thấy được những ứng dụng của trí tuệ nhân tạo qua các trò chơi phổ biến, năm 2018, trí tuệ nhân tạo đã có mặt ở tất cả các lĩnh vực và đời sống của con người, ví dụ như các trợ lý ảo được cài đặt trên các thiết bị điện thoại di động thông minh ngày nay. Theo một báo cáo nghiên cứu gần đây, các nhà phân tích thị trường dự báo thị trường trí tuệ nhân tạo sẽ đạt tới 47 tỷ USD vào năm 2020. Vì thế, trí tuệ nhân tạo đang nhanh chóng trở thành nền tảng cốt lõi của các ngành công nghiệp trên thị trường hiện nay.

Nhờ khoa Khoa học Máy tính và cô Huỳnh Thị Thanh Thương, nhóm đã có cơ hội thực hiện đồ án này và thêm một bước gần hơn với lý tưởng máy móc có trí thông minh. Từ những kiến thức nền tảng được truyền tải ở mỗi buổi học và việc tìm tòi, học hỏi, nghiên cứu các kiến thức mới, nhóm đã hoàn thành đồ án Bắn xe tăng. Trong quá trình thực hiện đồ án, những sai sót là không thể tránh khỏi. Chính vì vậy, nhóm mong nhận được những ý kiến đóng góp từ giảng viên để đồ án được hoàn thiện hơn.

# HỢP ĐỒNG NHÓM

1. THÀNH VIÊN NHÓM:

|  |  |  |  |
| --- | --- | --- | --- |
| STT | HỌ VÀ TÊN | MSSV | VAI TRÒ |
| 1 | Nguyễn Trần Quốc Toàn | 16521257 | Nhóm trưởng |
| 2 | Nguyễn Phạm Long Duy | 16520299 | Thành viên |
| 3 | Võ Thành Cường | 16521497 | Thành viên |
| 4 | Phạm Ngọc Phúc Thuần | 16521206 | Thành viên |

1. HỢP ĐỒNG NHÓM:
   1. Mục tiêu:

* Cùng hợp tác, cùng giúp đỡ lẫn nhâu trong tất cả các vấn đề mà nhóm đề ra.
* Có ý thức trách nhiệm với những gì mình làm, rèn luyện kỹ năng sống lẫn thái độ làm việc nhóm.
* Sau khi kết thúc môn học, sản phẩm có thể sử dụng được và các thành viên có thể vận dụng các kỹ năng lập trình và làm việc vào đời sống.
  1. Hiệp định:
* Ra nguyên quyết định theo nguyên tắc đồng thuận.
* Hoàn thành công việc đúng thời hẹn.
* Đề cao sự tôn trọng giữa các thành viên trong nhóm.
* Mục đích cuối cùng đạt được là hoàn thành tốt công việc nhóm.
  1. Các quy định và kỹ cương của nhóm:
* Có mặt đầy đủ trong các buổi họp nhóm.
* Tích cực trao đổi đóng góp ý kiến cá nhân để hoàn thiện đồ án
* Chịu trách nhiệm về những việc mình được phân công, hoàn thành hết khả năng của bản thân.
* Có vấn đề phát sinh lập tức thông báo cho nhóm để kịp thời chỉnh sửa.
* Tôn trọng sự hợp tác của mỗi thành viên trong nhóm.
* Hình thức họp nhóm: chủ yếu họp mặt online bằng Messenger, Teamview, … và cũng có những buổi họp mặt cụ thể (sẽ thông báo buổi họp trước 3 ngày để các bạn chuẩn bị).

1. BẢNG ĐÁNH GIÁ VỀ MỐI QUAN HỆ HỢP TÁC:
   1. Cách đánh giá thành viên:

* Cách thức tính phần trăm đóng góp của mỗi thành viên dựa vào công việc và mức độ làm việc của các thành viên tính theo tỷ trọng của bảng đánh giá chi tiết bên dưới.
  1. Bảng đánh giá chi tiết:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ***STT*** | ***Tỷ trọng*** | ***Tiêu chí*** | ***Tuyệt vời*** | ***Tốt*** | ***Tạm được*** | ***Kém*** |
| 1 | 40% | Đóng góp về nội dung, chất lượng bài làm | Số lượng bài làm nhiều. Chất lượng bài làm tốt. Đánh giá bài làm của các bạn khác và góp ý để chỉnh sửa. Sẵn sàng hướng dẫn lại cho bạn/sửa lại/làm lại nếu thấy chưa đúng | Số lượng bài làm nhiều. Chất lượng bài làm tốt. Đánh giá bài làm của các bạn khác và góp ý chỉnh sửa | Có chịu khó làm bài.  Chất lượng bài làm không tốt, nhưng biết học hỏi, lắng nghe góp ý của các bạn và chỉnh sửa lại | Không làm bài, hoặc làm ít và chất lượng bài làm kém, sai nhiều nhưng không chỉnh sửa/làm lại theo góp ý của các bạn |
| 2 | 10% | Tính thần trách nhiệm | Hoàn thành tốt công việc được phân công trong thời gian đề ra, giúp đỡ các thành viên khác | Hoàn thành tốt công việc được phân công trong thời gian đề ra | Hoàn thành công việc | Không hoàn thành công việc |
| 3 | 15% | Giải quyết vấn đề (vấn đề có thể là khó khăn trong hoạt động nhóm, khó khăn khi làm bài tập)  Tư duy sáng tạo | Tham gia nhiệt tình đóng góp ý kiến trong các buổi họp nhóm  Luôn đưa ra được cách thức/ ý tưởng mới để giải quyết vấn đề  Giải được các bài tập khó | Đóng góp ý kiến tương đối nhiều | Thỉnh thoảng đóng góp | Không đóng góp ý kiến nào hết |
| 4 | 10% | Tinh thần hợp tác | Dựa trên sự tôn trọng và mục đích cuối cùng là đạt được kết quả tốt trong cộng việc và lợi ích của nhóm | Đạt kết quả tót trong công việc | Đạt được kết quả tạm được | Không đạt kết quả |
| 5 | 10% | Quản lý xung đột | Các bạn hợp tác với nhau tìm ra giải pháp tốt nhất cho cả đôi bên, chú trọng sự đồng thuận. | Cùng nhau tìm ra những giải pháp trung hòa để đôi bên cùng có một phần lợi ích | Quan tâm đến giữ các mối quan hệ chứ không cần quan tâm đến kết quả | Né tránh sự va chạm, sợ đối đầu, không quan tâm đến nhu cầu của các bên |
| 6 | 5% | Họp nhóm | 2 lần / tuần | 1 lần/ tuần | Nảy sinh vấn đề mới họp | Không hợp |
| 7 | 10% | Tìm kiếm thông tin | Tìm kiếm đầy đủ, phong phú các thông tin mà nhóm cần, tổng hợp một cách hiệu quả | Tìm kiếm đầy đủ, phong phú các thông tin | Tìm kiếm thông tin để đối phó | Không tìm báo cáo sơ sài |

# BẢNG PHÂN CÔNG CÔNG VIỆC

|  |  |  |
| --- | --- | --- |
| STT | THÀNH VIÊN | CÔNG VIỆC |
| 1 | Nguyễn Trần Quốc Toàn | * Thiết kế cơ sở dữ liệu cơ bản * Cải tiến các cơ chế của xe tăng (bắn và né đạn) |
| 2 | Nguyễn Phạm Long Duy | * Xây dựng môi trường * Hỗ trợ xây dựng các hàm để đưa vào môi trường |
| 3 | Võ Thành Cường | * Xây dựng các hàm Heuristic cho xe tăng |
| 4 | Phạm Ngọc Phúc Thuần | * Hỗ trợ xây dựng môi trường * Viết báo cáo |

# CHƯƠNG 1: GIỚI THIỆU ĐỒ ÁN

1. LỊCH SỬ TRÒ CHƠI:

*Battle City* (*Bắn xe tăng*) là một trò chơi điện tử bắn súng đa hướng được Namco giới thiệu vào ngày 9 tháng 9 năm 1985. Sự xuất hiện của *Battle City* được xem là một bước cải tiến lớn so với phiên bản tiền nhiệm – *Tank Battalion,* đồng thời, góp phần quan trọng cho sự thành công của phiên bản tiếp theo – *Tank Force.*

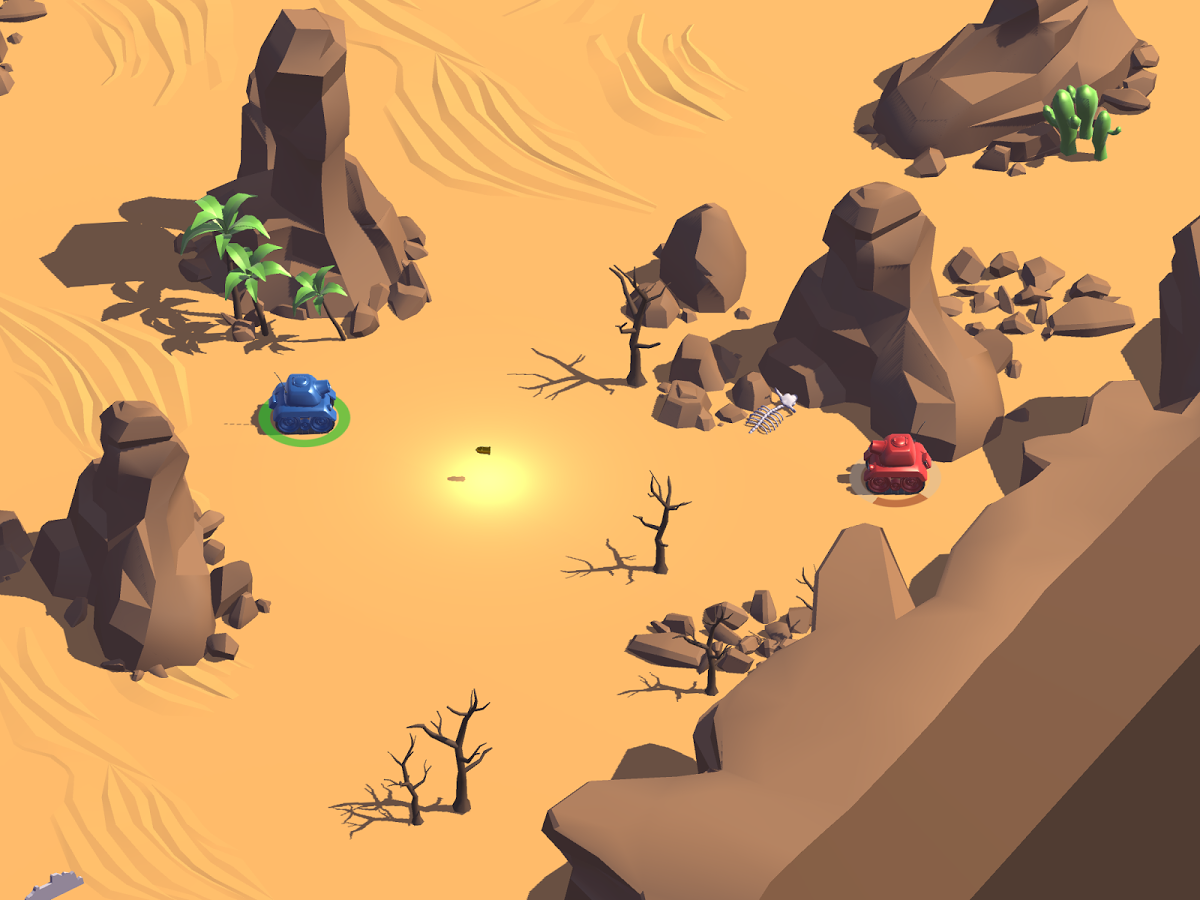


*Hình 1: Bộ sản phẩm trò chơi Battle City cho máy điện tử 4 nút*

**

*Hình 2: Máy điện tử 4 nút Famicom*

Từ nền tảng trò chơi trên máy điện tử 4 nút, Battle City đã được phát triển thêm trên nền tảng *arcade game* (dùng máy chơi game dùng xu). Năm 1991, Nova đã đưa trò chơi vào *Game Boy*, một hệ thống giả lập chơi game cầm tay 8-bit được phát triển và chế tạo bởi Nitendo. Hiện tại, *Battle City* đã được phát triển trên các nền tảng mới với nhiều sự thay đổi về lối chơi và cải tiến về đồ hoạ, điển hình là TANKS! được xây dựng bằng phần mềm Unity.



*Hình 3: Một cảnh từ trò chơi TANKS!*

1. MÔ TẢ TRÒ CHƠI:
   1. Cốt truyện:

Bạn là thành viên cuối cùng của lực lượng chỉ huy xe tăng tinh nhuệ. Bạn không có sự lựa chọn nào khác ngoài bảo vệ pháo đài và chống lại toàn bộ quân địch. Hãy lập ra chiến lược để tiêu diệt được quân địch nhanh nhất có thể. Đừng quên né đạn của quân địch và bảo vệ pháo đài nhé!

* 1. Mô tả sơ lược:
* Bạn xuất hiện bên trái pháo đài ở phần dưới của khung hình. Các xe tăng của quân địch sẽ xuất hiện từ một trong ba vị trí ở phần trên cùng của khung hình.
* Ở mỗi màn của trò chơi, bạn phải loại bỏ tổng cộng 20 xe tăng để đến được màn tiếp theo của trò chơi.
* Bạn có thể bắn đạn theo bốn hướng. Tiêu diệt xe tăng địch chỉ cần một lần bắn. Tuy nhiên, quân địch có một loại xe tăng yêu cầu bạn cần đến bốn lần bắn để tiêu diệt hoàn toàn.
* Nếu bạn bị trúng đạn của quân địch, bạn sẽ mất một mạng, Nếu pháo đài của bạn bị bắn, trò chơi sẽ kết thúc ngay lập tức, đồng nghĩa với việc bạn đã thua.
* Bạn có thể bắn đạn để phá tường. Một bức tường tiêu chuẩn cần 4 lần bắn để phá huỷ.
* Xe tăng địch nhấp nháy màu đỏ sẽ cung cấp *sức mạnh (power-up)* bất cứ khi nào bị bắn trúng. Những *sức mạnh* sẽ xuất hiện ngẫu nhiên ở mọi nơi trong khung hình.
  1. Mô tả chi tiết:
     1. *Các nút điều khiển:*
* Các phím mũi tên (Left, Right, Up Down): dùng để di chuyển xe tăng.
* Phím Space dùng để bắn đạn.
* Phím Esc dùng để tạm dừng trò chơi.
  + 1. *Xe tăng:*
  + Người chơi:

|  |  |
| --- | --- |
|  | Người chơi có thể tiêu diệt xe tăng của quân địch hoặc phá tường bằng cách bắn đạn. Tuy nhiên, người chơi chỉ được bắn lượt đạn tiếp theo khi lượt đạn trước kết thúc. |

* + Quân địch:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Quân địch* | *Loại xe* | *Điểm* | *Máu* | *Tốc độ*  *di chuyển* | *Thông tin thêm* |
|  | Seeker | 100 | 1 | 3 | Tìm đường đi tốt nhất đến người chơi và bắn mục tiêu |
|  | Bulldozer | 200 | 1 | 1 | Tìm đường đi tối nhất đến nhà chính và phá huỷ nó (chỉ có thể càng về phía trước) |
|  | Hunter | 300 | 1 | 3 | Xem xét vị trí tối ưu nhất giữa người chơi và pháo đài và đi đến nó và thủ tiêu (có thể bắn và né đạn một cách linh hoạt) |

* + 1. *Sức mạnh (Power-up):*

Mỗi sức mạnh sẽ xuất hiện tại một trong 16 vị trí ngẫu nhiên trên màn hình bất kì khi nào một trong ba xe tăng nhấp nháy bị bắn. Sức mạnh này sẽ biến mất khi một xe tăng nhấp nháy khác xuất hiện. Chạm vào các sức mạnh để thu nhập chúng (chúng sẽ biến mất) và người chơi có thể nhận được 500 điệm khi sức mạnh được sử dụng.

|  |  |  |
| --- | --- | --- |
| *Sức mạnh* | *Tên gọi* | *Công dụng* |
|  | Lưu đạn  (Grenade) | * Tiêu diệt toàn bộ xe tăng của quân địch đang xuất hiện trên khung hình. * Không áp dụng cho các xe tăng đã bị tiêu diệt |
|  | Mũ sắt (Helmet) | * Tạo một lá chắn tạm thời xung quang xe tăng của người chơi để ngăn chặn đạn của quân địch. |
|  | Xẻng (Shovel) | * Biến các tường gạch xung quanh pháo đài thành tường thép, mang lại khả năng bất tử tạm thời cho các bức tường và ngăn chặn quân địch phá huỷ pháo đài. * Xây dựng lại những bức tường chưa bị phá huỷ hoàn toàn. |
|  | Ngôi sao (Star) | * Tăng sức tấn công của người chơi lên một cấp (có 4 cấp độ: mặc định, cấp độ 2, cấp độ 3, cấp độ 4). * Khi người chơi chết (mất một mạng), các thông số này sẽ trở về mặc định. * Ngôi sao thứ nhất (cấp độ 2): tốc đạn bắn ngang với tốc độ đạn bắn của Power Tanks. * Ngôi sao thứ hai (cấp độ 3): người chơi có thể bắn 2 viên đạn trong một khung hình. * Ngôi sao thứ ba (cấp độ 4): đạn bắn có thể phá huỷ được tường thép (yêu cầu 2 lần bắn) và gây thiệt hại gấp đôi đối với tường gạch! |
|  | Xe tăng (Tank) | * Người chơi được thêm một mạng * Cách duy nhất để có thêm một mạng là phải đạt được 20,000 điểm |
|  | Thời gian (Timer) | * Sức mạnh này làm đóng băng tạm thời thời gian và ngăn mọi sự di chuyển của xe tăng địch. * *Gợi ý:* Hãy sử dụng khả năng này để tiếp cận xe tăng địch và tiêu diệt chúng. |

* + 1. *Môi trường:*

Mỗi màn chơi sẽ có nhiều loại chướng ngại vật khác nhau nằm rải rác xung quanh. Chìa khoá để sống sót và chiến thắng là hãy sử dụng các đặc tính của chúng để tạo lợi thế cho bản thân.

|  |  |  |
| --- | --- | --- |
| *Môi trường* | *Loại môi trường* | *Đặc tính* |
|  | Tường gạch | * Xe tăng và đạn không thể vượt qua tường gạch. * Tường gạch có thể bị phá uỷ bởi 4 lần bắn. |
|  | Tường thép | * Xe tăng và đạn HOÀN TOÀN không thể vượt qua tường thép. |
|  | Rừng | * Xe tăng và đạn có thể di chuyển dưới tán cây. * *Cẩn thận:* quân địch di chuyển dưới tán cây sẽ rất khó phát hiện. Đặc biệt, đạn bay dưới tán cây sẽ càng khó phát hiện hơn! |
|  | Sông | * Xe tăng không thể vượt qua sông. * Nhưng đạn có thể bay qua sông. |
|  | Băng | * Xe tăng có thể di chuyển trên băng. Xe tăng sẽ phải trượt lên một chút để dừng lại hoàn toàn. * *Cẩn thận:* đạn có thể bay trên băng. |

# CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

1. NGÔN NGỮ LẬP TRÌNH C#:

* C# là một ngôn ngữ lập tình đơn giản, được phát triển bỏi đội ngũ kỹ sư của Micrrosoft vào năm 2000, trong đó người dẫn đầu là Anders Hejlsberg và Scott Wiltamunth.
* C# là ngôn ngữ lập trình hiện đại, hướng đối tượng và nó được xây dựng trên nền tảng của hai ngôn ngữ mạnh nhất là C++ và Java. Do đó, C# được miêu tả là ngôn ngữ có sự cân bằng giữa C++, Visual Basic, Delphi và Java.
* C# được thiết kế cho Common Language Infrastructure (CLI), mà gồm Executable Code và Runtime Environment, cho chép người dùng sử dụng các ngôn ngữ high-level đa dạng trên các nền tảng và cấu trúc máy tính khác nhau.
* C# với sự hỗ trợ mạnh mẽ của .NET Framework giúp cho việc tạo một ứng dụng Windows Forms hay WPF (Windows Prensentation Foundation),… trở nên rất dễ dàng.

1. LÝ THUYẾT ÁP DỤNG:

Bài toán trong trò chơi Bắn xe tăng (Battle City) được xây dựng và giải quyết trên cơ sở lý thuyết thuật giải A\*.

* 1. Định nghĩa thuật giải A\*:

A\* là giải thuật tìm kiếm trong đồ thị, tìm đường đi từ một từ một đỉnh hiện tại đến đỉnh đích có sử dụng hàm để ước lượng khoảng cách hay còn gọi là *hàm Heuristic.*

*Heuristic là gì?* Heuristic là phương pháp giải quyết vấn đề dựa trên phỏng đoán, ước chừng, kinh nghiệm, trực giác để tìm ra giải pháp gần như là tốt nhất, nhanh chóng, dễ dàng.

*Hàm Heuristic là gì?* Hàm Hueristic là hàm ứng với mỗi trạng thái hay mỗi sự lựa chọn một giá trị ý nghĩa đối với vấn đề dựa vào giá trị hàm này ta lựa chọn hành động.

Từ trạng thái hiện tại A\* xây dựng tất cả các đường đi có thể đi dùng hàm ước lược khoảng cách (hàm Heuristic) để đánh giá đường đi tốt nhất có thể đi. Theo mỗi dạng bài khác nhau mà hàm Heuristic sẽ được đánh giá khác nhau. A\* luôn tìm được đường đi ngắn nhất nếu tồn tại đường đi như thế. Trong Battle City, nhóm sẽ xây dựng các hàm Heuristic cho xe tăng của quân địch đã được liệt kê ở chương 1, phần 2.

A\* lưu giữ một tập các đường đi qua đồ thị, từ đỉnh bắt đầu đến đỉnh kết thúc. Tập các đỉnh có thể đi tiếp được lưu trong tập Open. Thứ tự ưu tiên cho một đường đi đươc quyết định bởi hàm Heuristic được đánh giá bằng công thức:

|  |  |  |
| --- | --- | --- |
|  | *f(x) = g(x) + h(x)* |  |

Trong đó:

* *g(x)* là chi phí của đường đi từ đỉnh xuất phát cho đến thời đỉnh hiện tại;
* *h(x)* là hàm ước lượng chi phí từ đỉnh hiện tại đến đỉnh đích;
* *f(x)* là độ tốt, dùng để so sánh và chọn đường đi, *f(x)* có giá trị càng thấp thì độ ưu tiên càng cao.
  1. Mô tả thuật giải A\*:
* Tạo tập Open để chứa các đỉnh lân cận nhưng chưa xét đến.
* Tạo tập Close để chứa các đỉnh đã xét qua.
* cost(p, q): khoảng cách giữa p, q.
* g(p): khoảng cách từ đỉnh bắt đầu đến đỉnh hiện tại.
* h(p): giá trị được ước lượng từ đỉnh hiện tại đến đỉnh kết thúc.
* f(p) = g(p) + h(p)
* Bước 1:
  + - Open: = {s} // s là đỉnh bắt đầu
    - Close: = { }
* Bước 2: while (Open != { })
  + - Chọn p thuộc tập Open có f(p) nhỏ nhất.
    - Nếu p là đỉnh kết thúc thì thoát và thông báo kết quả.
    - Chuyển p qua tập Close và mở các q sau p.
      * Nếu q đã có trong Open:
        + Nếu g(q) > g(p) + cost(p, q):

g(q) = g(p) + cost(p, q)

f(q) = g(q) + h(q)

prev(q) = p // prev = đỉnh cha // đỉnh cha của q là p

* + - * Nếu q chưa có trong Open và q chưa có trong Close:
        + g(q) = g(p) + cost(p, q)
        + f(q) = g(q) + h(q)
        + prev(q) = p
        + Thêm q vào Open
      * Nếu q có trong Close:
        + Nếu g(q) > g(p) + cost(p, q)

Bỏ q khỏi Close

g(q) = g(p) + cost(p, q)

f(q) = g(q) + h(q)

prev(q) = p

Thêm q vào Open

Cập nhật các đỉnh chịu ảnh hướng từ sự thay đổi của q

* Bước 3: Không tìm được.
  1. Bài toán mô phỏng tìm đường đi ngắn nhất A\* ĐƠN GIẢN:

h = 6

|  |
| --- |
| 8  3  2  3  8  11  7  4  7  10  5  h = 1  h = 5  h = 7  h = 3  h = 4  h = 3 |

h(A) = 6; h(B) = 5; h(C) = 4; h(D) = 3; h(E) = 3; h(F) =7; h(G) = 1.

Tìm đường đi ngắn nhất từ đỉnh A đến đỉnh F.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Bước* | *p* | *Các đỉnh nối với p* | *Open* | *Close* |
| 0 |  |  | A6 |  |
| 1 | A | B | B5 | A |
| 2 | B | C | C4 | A, B |
| 3 | C | D | D3 | A, B, C |
| 4 | D | B, E, G | E3, G1 | A, B, C, D |
| 5 | E | F | G1, F7 | A, B, C, D, E |
| 6 | G | A, B, F | F7 | A, B, C, D, E, G |
| 7 | F  (dừng) | E |  |  |

Cây tìm kiếm với đồ thị trên:

|  |
| --- |
| h(F) = 7; g(F) = 29  h(E) = 3; g(E) = 26;  f(E) = 26 + 3 = 29  h(G) = 1; g(G) = 31;  f(G) = 31 + 3 = 34  h(D) = 3; g(D) = 20;  f(D) = 20 + 3 = 23  h(C) = 4; g(C) = 13;  f(A) = 13 + 4 = 17  h(B) = 5; g(B) = 8;  f(B) = 8 + 5 = 13  h(A) = 6; g(A) = 0;  f(A) = 0 + 6 =6 |

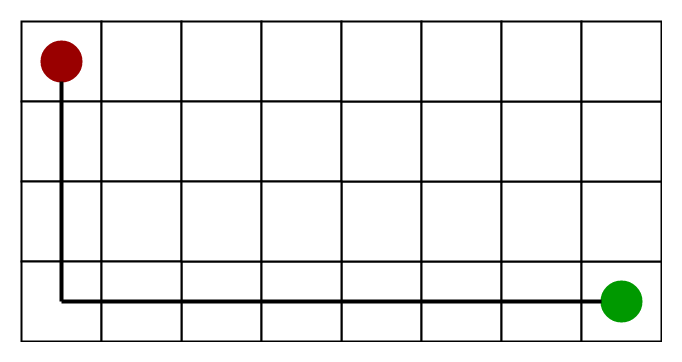
* 1. Các cách tính giá trị Heuristic:

Các bài toán trong thực tế không hề biết trước giá trị heuristic (*h(x)*).

Do đó, việc xây dựng các hàm Heuristic để tính giá trị của *h(x)* là vô cùng quan trọng. Sau đây là ba phương pháp tính *h(x)* thường được sử dụng:

* + 1. *Đường Manhattan:*

Sử dụng đường Manhattan khi chỉ được phép di chuyển theo bốn hướng (trái phải, trên, dưới). Ví dụ như hình bên dưới, đường đi ngắn nhất từ chấm đỏ đến chấm xanh lá theo áp dụng đường Manhattan.



*Hình 4: Đường đi theo phương pháp đường Manhattan*

Công thức tính *h(x)*:

|  |
| --- |
| h = abs(current\_cell.x – goal.x) + abs(current\_cell.y – goal.y) |

* + 1. *Đường Diagonal:*

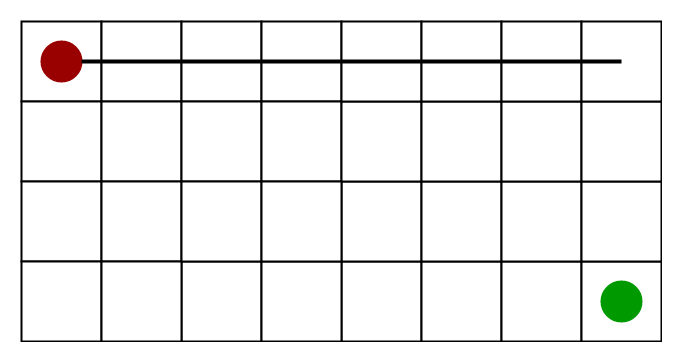
|  |  |  |
| --- | --- | --- |
| X | X | X |
| X | *Điểm hiện tại* | X |
| X | X | X |

Sử dụng đường Diagonal khi được phép di chuyển theo tám hướng (các ô được đánh dấu X) xung quanh điểm hiện tại.

Công thức tính *h(x)*:

|  |
| --- |
| h = max {abs(current\_cell.x – goal.x), abs(current\_cell.y – goal.y)} |

Đường đi ngắn nhất từ chấm đỏ đến chấm xanh lá theo áp dụng đường Diagonal:



*Hình 5: Đường đi theo phương pháp đường Diagonal*

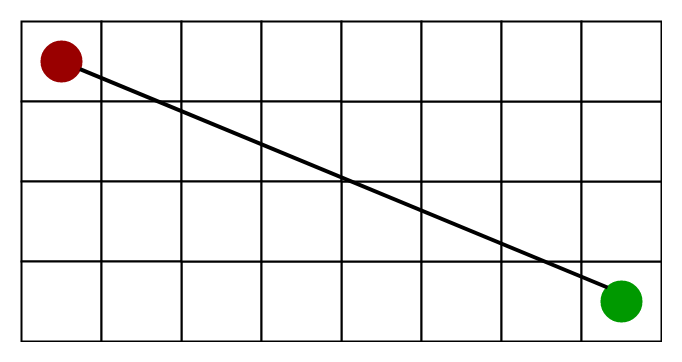
* + 1. *Đường Euclidean:*

Sử dụng đường Euclidean khi được phép di chuyển theo TẤT CẢ các hướng. Tức là đường đi không còn giới hạn là đường thẳng mà có thể là đường chéo!

Công thức tính *h(x)*:

|  |
| --- |
| h = sqrt((current\_cell.x – goal.x)\*2 + (current\_cell.y – goal.y)\*2) |

Đường đi ngắn nhất từ chấm đỏ đến chấm xanh lá theo áp dụng đường Euclidean:



*\* Chú giải các phép toán:*

* abs(x): trị tuyệt đối của x.
* sqrt(x): căn bậc hai của x.
* max(x, y): giá trị lớn nhất giữa x và y.
  1. Nhận xét về thuật giải A\*:
     1. *Ưu điểm:*
* Linh động, tổng quát.
* Nhanh chóng tìm được lời giải.
  + 1. *Hạn chế:*
* Tốn khá nhiều bộ nhớ để lưu lại những trạng thái (đỉnh) đã đi qua.

# CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ

PHÂN TÍCH BÀI TOÁN:

* 1. Đặt vấn đề:

Trò chơi *Battle City* (Bắn xe tăng) là một trò chơi điện tử kinh điển của thế giới. Do đó, việc thiết kế và cải tiến trò chơi yêu cầu nhiều thời gian xây dựng và khả năng xử lý bug (lỗi) tốt. Bên dưới là yêu cầu chi tiết đối với từng công việc.

* Về thiết kế trò chơi:
  + Phần mềm thiết kế: Unity.
  + Ngôn ngữ lập trình sử dụng: C#.
  + Thiết kế môi trường, hình ảnh, phông chữ theo hướng trò chơi Battle City truyền thống (game 8-bit).
  + Hàm heuristic (cách tính *h(x)*) để tìm đường đi ngắn nhất đến mục tiêu (tuỳ thuộc vào từng loại xe tăng).
  + Xử lý đụng độ.
* Về cải tiến trò chơi:
  + Cải tiến cơ chế bắn của xe tăng:
    - Phát hiện xe tăng của người chơi.
    - Tiết kiệm đạn: khi phát hiện ra xe tăng của người chơi, xe tăng sẽ quyết định bắn hay không.
  + Cải tiến cơ chế né đạn của xe tăng:
    - Phát hiện ra hướng đạn đến.
    - Tìm ra cách né đạn (nếu có thể).

1. THIẾT KẾ CẤU TRÚC DỮ LIỆU VÀ TRẠNG THÁI:
   1. Không gian trạng thái của bài toán: Vị trí của các xe tăng trên bản đồ.
   2. Thông tin dữ liệu:

* Xe tăng: bao gồm xe tăng của người chơi và xe tăng của quân địch (Bulldozer, Seeker, Hunter).
* Bản đồ: bao gồm các node.
* Giao diện: nơi thể hiện hình ảnh và tương tác với người chơi.
  1. Sơ đồ cấu trúc chương trình (sơ đồ lớp):

|  |
| --- |
| Màn chơi  Stage one  Scene  Bulldozer  Seeker  Player  player  Hunter  About  Main menu  Giao diện  Tường thép  Tường gạch  Node  Bản đồ  Help  Tank  Battle City |

1. CÁC VẤN ĐỀ VÀ THUẬT GIẢI:
   1. Hàm heuristic cho các xe tăng:
      1. *Đối với xe Bullozer:* sẽ tìm đường đi ngắn nhất đến pháo đài và tiêu diệt nó.
         1. *Mã giả:*

Bước 1:

* Gán trị khởi tạo: Node startNode, currentNode, target. Trong đó, startNode vị trí bắt đầu của xe được gán như là đỉnh bắt đầu. currentNode vị trí hiện tại của xe khi di chuyển. target là mục tiêu và đối với Bullozer là pháo đài của kẻ thù.
* Khởi tạo các giá trị sau: g = new float[180]; f = new float[180]; pre = new Node[180]; cùng với tập open và close List<Node> open = new List<Node>(); List<Node> close = new List<Node>();
* Gán giá trị g[currentNode] = 0; f[currentNode] = calHeuristic(currentNode, target) (Với hàm heuristic là đường mahattan giữa đỉnh current và target đã được đề cập ở phần 2.4.1).
* Thêm đỉnh hiện hành vào trong tập Open. open.Add(currentNode);

Bước 2: Thiết lập giải thuật A sao

* Kiểm tra tập Open có là tập rỗng hay không
* Nếu đúng: Không tìm được đường đi đến mục tiêu và kết thúc chương trình
* Nếu sai: Gán giá trị currentNode bằng giá trị nhỏ nhất trong tập open và kiểm tra tiếp điều kiện nếu đỉnh hiện hành trùng với mục tiêu thì truy xuất đường đi từ trong danh sách list và kết thúc chương trình. Nếu không trùng với mục tiêu thì ta tiếp tục loại bỏ đỉnh hiện hành ra khỏi tập open, thêm nó vào tập close và duyệt tất cả đỉnh phù hợp so với đỉnh hiện hành vào danh sách list nextNode mới.

Bước 3: Duyệt tất cả đỉnh q kề với đỉnh p có trong danh sách nextNode

Nếu q đã có trong Open: Kiểm tra điều kiện nếu g(q) > g(p) + cost(p, q) như sau:

|  |
| --- |
| if (g[q.getName()] > g[currentPoint.getName()] + calculateHValue(currentPoint.getGameobj(), q.getGameobj()))  {  g[q.getName()] = g[currentPoint.getName()] + calculateHValue(currentPoint.getGameobj(), q.getGameobj()) + q.obstacle;  f[q.getName()] = g[q.getName()] + calculateHValue(target, q.getGameobj());  pre[q.getName()] = currentPoint;  } |

Nếu q chưa có trong Open và q chưa có trong Close:

|  |
| --- |
| g[q.getName()] = g[currentPoint.getName()] + calculateHValue(currentPoint.getGameobj(), q.getGameobj()) + q.obstacle;  f[q.getName()] = g[q.getName()] + calculateHValue(target, q.getGameobj());  pre[q.getName()] = currentPoint;  open.Add(q); |

Nếu q có trong Close: kiểm tra điều kiện nếu g(q) > g(p) + cost(p, q) như sau

|  |
| --- |
| if (g[q.getName()] > g[currentPoint.getName()] + calculateHValue(currentPoint.getGameobj(), q.getGameobj())  {  close.Remove(q);  open.Add(q);  g[q.getName()] = g[currentPoint.getName()] + q.obstacle;  f[q.getName()] = g[q.getName()] + calculateHValue(target, q.getGameobj());  pre[q.getName()] = currentPoint;  } |

Trong đó, q.obstacle là trọng số của node q.

* + - 1. *Lưu đồ:*
      2. *Code:*

*Phần khai báo khởi tạo:*

Node startNode, currentNode, target

g = new float[180];

f = new float[180];

pre = new Node[180];

List<Node> open = new List<Node>();

List<Node> close = new List<Node>();

g[currentNode] = 0;

f[currentNode] = calH(currentNode, target);

open.Add(currentNode);

Phần thiết lập giải thuật A sao:

Gán giá trị currentNode bằng giá trị nhỏ nhất trong tập open

True

False  
False

open.Count !=0

If (target.name == currentPoint.name)

True

False  
False

open.Remove(currentPoint);

close.Add(currentPoint);

thêm vào list các đỉnh hợp lệ quanh currentPoint vào list NextPoint;

Duyệt mỗi Node trong tập NextPoint

Xét từ đỉnh q kề với p trong NextPoint:

True

g[q.getName()] = g[currentPoint.getName()] + calculateHValue(currentPoint.getGameobj(), q.getGameobj()) + q.obstacle;

f[q.getName()] = g[q.getName()] + calculateHValue(target, q.getGameobj());

pre[q.getName()] = currentPoint;

open.Add(q);

q có không có trong close và open

False

if (g[q.getName()] > g[currentPoint.getName()] + calculateHValue(currentPoint.getGameobj(), q.getGameobj())) {

g[q.getName()] = g[currentPoint.getName()] + calculateHValue(currentPoint.getGameobj(), q.getGameobj()) + q.obstacle;

f[q.getName()] = g[q.getName()] + calculateHValue(target, q.getGameobj());

pre[q.getName()] = currentPoint; }

q có trong open

True

True

if (g[q.getName()] > g[currentPoint.getName()] + calculateHValue(currentPoint.getGameobj(), q.getGameobj()) {

close.Remove(q);

open.Add(q);

g[q.getName()] = g[currentPoint.getName()] + q.obstacle;

f[q.getName()] = g[q.getName()] + calculateHValue(target, q.getGameobj());

pre[q.getName()] = currentPoint; }

q có trong close

c/

* + 1. Cơ chế bắn của xe tăng:
* Chỉnh hướng nòng súng (hướng bay của đạn) theo hướng đi (quay mặt) của xe tăng hiện tại.

|  |
| --- |
| float x = (firePosition.position.x – transform.position.x);  float y = (firePosition.position.x – transform.position.x);  if(!(x==0 && y==0))  {  dx = (Mathf.Abs(x) >= Mathf(y)) ? 1 : 0;  dy = 1 – (dx\*1);  if(x>0)  dx\*=1;  else dx\*=-1;  if(y>0)  dy\*=1;  else dy\*=-1;  } |

* Sử dụng tính năng của *Raycast* để phát hiện xe tăng của người chơi:
  + Chỉnh hướng quét theo hướng của nòng súng.

|  |
| --- |
| RaycastHit2D hit = Physics2D.Raycast (firePosition.position, new Vector2(dx, dy)); |

* + Nếu phát hiện có người chơi trên đường quét của tia (*ray*), xe tăng sẽ đưa ra quyết định có bắn hay không.

|  |
| --- |
| float R = 10; // tầm bắn  if (hit.collider != null) //phát hiện chướng ngại vật trên đường quét  {  //kiểm tra chướng ngại vật có phải người chơi hay không  if ((hit.rigidbody != null)&&(hit.rigidbody.tag == “Player”))  {  float DX = Mathf.Abs(hit.rigidbody.position.x – transform.position.x)  float DY = Mathf.Abs(hit.rigidbody.position.y – transform.position.y)  // Kiểm tra người chơi có nằm trong tầm bắn R  if (((DX<=R) && (dx!=0)) || ((DY<=R) && (dy!=0)))  {  Debug.Log(“Pew”);  Debug.Log(hit.collider.name);  Shoot(); //bắn đạn  }  }  } |

* + DX và DY là khoảng cách mà tia đã quét được, đây cũng là khoảng cách của đạn bắn ra để tiêu diệt người chơi.
  + Người chơi phải nằm trong tầm bắn R (tầm nhất định sao cho người chơi có ít thời gian né đạn nhất) thì xe tăng mới bắn đạn.
* Nhờ đó, xe tăng có thể tiết kiệm số lượng đạn bắn ra.
  1. Cơ chế né đạn của xe tăng:
* Tương tự như cơ chế bắn, cơ chế né đạn sử dụng tính năng của *RayCast* để phát hiện đạn đang hướng tới xe tăng.
* Xây dựng hàm phát hiện nguy hiểm (đạn):

|  |
| --- |
| // hit là hướng đang xét  public int Threat\_Detection(RaycastHit2D hit, ref float Clear\_Time)  {  if (hit.collider != null)  {  if((hit.rigidbody != null) && (hit.rigidbody.tag == "bullet"))  {  Debug.Log(hit.rigidbody.name);  float Impact\_Time = hit.distance / 10;  float Clear\_Time = (hit.distance + 1) /10;  float Evade\_Time =(float)(1.2) / this.speed;  if (Impact\_Time > Evade\_Time)  return 1;  else return 2;  }  }  return 0;  } |

* + Impact\_Time: thời gian sẽ xảy ra va chạm.

Impact\_Time = khoảng cách từ vị trí đạn đến xe tăng / vận tốc của viên đạn

* + Clear\_Time: thời gian đạn bay qua vị trí hiện tại của xe tăng.

Clear\_Time = (khoảng cách từ vị trí đạn đến xe tăng +1) / vận tốc của viên đạn

* + Evade\_Time: thời gian xe tăng né đạn.

Evade\_Time = (1 + 0.2) / tốc độ di chuyển của xe tăng

*Trong đó:*

*1 là khoảng cách để di chuyển qua vùng có đạn.*

*0.2 là khoảng cách trừ hao để đưa ra quyết định vị trí di chuyển đến.*

* + Vận tốc vủa viên đạn mặc định là 10.
* Từ hàm phát hiện nguy hiểm, xe tăng sẽ rơi vào 3 trường hợp:
  + Trường hợp Threat\_Detection(hit) = 0: không phát hiện ra nguy hiểm (đạn).
  + Trường hợp Threat\_Detection(hit) = 1: phát hiện nguy hiểm (đạn) và có thể né được.
  + Trường hợp Threat\_Detection(hit) = 2: phát hiện nguy hiểm (đạn) nhưng không thể né được.
* Nếu xe tăng rơi vào trường hợp Threat\_Detection(hit) = 1, xe tăng sẽ né đạn như sau:
  + Nếu đạn bay tới theo phương ngang (từ phía bên trái hoặc bên phải) thì xe tăng sẽ né bằng cách di chuyển lên trên hoặc xuống dưới:

|  |
| --- |
| if (C % 4 == 0)  {  Node u = this.currentNode.TopNode;  if (Reachable(u))  {  nextNode = u;  }  else {  u = this.currentNode.BottomNode;  if (Reachable(u))  {  nextNode = u;  }  }  Wait(Clear\_Time);  Debug.Log("Up Or Down");  } |

* + Nếu đạn bay tới theo phương dọc (từ phía bên trên hoặc bên dưới) thì xe tăng sẽ né bằng cách di chuyển sang bên trái hoặc bên phải:

|  |
| --- |
| else // if ( C % 4 != 0 )  {  Node u = this.currentNode.LeftNode;  if (Reachable(u))  {  nextNode = u;  }  else  {  u = this.currentNode.RightNode;  if (Reachable(u))  {  nextNode = u;  }  }  Wait(Clear\_Time);  } |

* + Trong đó:
    - C là hướng đạn bay của đạn sẽ được xác định chi tiết phía dưới.
    - Reachable(u) là hàm kiểm tra node đó có thể di chuyển đến không (an toàn không).

|  |
| --- |
| bool Reachable(Node u)  {  if (u == null || u.obstacle == 9999) // 9999 là tường thép  {  return false;  }  return true;  } |

* + - Wait(Clear\_Time) là hàm đợi cho đến khi hết thời gian Clear\_Time, nghĩa là xe tăng đã an toàn và có thể tiếp tục di chuyển.

|  |
| --- |
| IEnumerator Wait (float Time)  {  yield return new WaitForSeconds(Time);  } |

* Xác định hướng bay của đạn (C) bằng cách sử dụng bitmap:

|  |  |  |
| --- | --- | --- |
|  | 0001  (1) |  |
| 0100  (4) | Vị trí xe tăng  0000 | 1000  (8) |
|  | 0010  (2) |  |

Nhờ vào việc sử dụng bitmap, xe tăng có thể nhanh chóng phát hiện hướng đi của viên đạn. Ví dụ, nếu viên đạn bay đến từ phía bên phải:

|  |
| --- |
| int code = 0; // để lưu hướng đạn đến  bool Died = false;  float tmp = -1;  float Clear\_Time = 0;  switch (Threat\_Detection(R,ref tmp))  {  case 1:  {  code |= 8;  if (Clear\_Time < tmp)  Clear\_Time = tmp;  break;  }  case 2:  {  Died = true;  break;  }  default:  break;  } |

* 1. Xử lý đụng độ:

# CHƯƠNG 4: ỨNG DỤNG

1. GIỚI THIỆU CHƯƠNG TRÌNH ỨNG DỤNG:
   1. Các thông tin cơ bản của chương trình:

* Chương trình thiết kế cho một người chơi.
* Trò chơi có tổng cộng 10 màn (level).
* Hai trường hợp kết thúc trò chơi:
  + Người chơi chiến thắng: là khi người chơi tiêu diệt được tất cả xe tăng của quân địch.
  + Người chơi thua cuộc: là khi người chơi hoặc pháo đài bị tiêu diệt.
  1. Giao diện tương tác:
* Giao diện trò chơi được thiết kế theo giao diện Battle City truyền thống:



* Giao diện Main menu có ba sự lựa chọn: *Play*, *Help*, *About*.
  + *Play:* dẫn đến màn (level) 1 để người chơi bắt đầu chơi.
  + *Help*: hướng dẫn cách chơi cho người chơi.



* + *About*: giới thiệu về các thành viên của nhóm.



* 1. Hướng dẫn cài đặt và sử dụng:

# CHƯƠNG 5: KẾT LUẬN

1. KẾT QUẢ ĐẠT ĐƯỢC:

* Xây dựng được trò chơi với giao diện, hiệu ứng theo kiểu game 8-bit, tạo được được sự gần gũi và quen thuộc đối với người chơi.
* Cách chơi đơn giản, hoàn toàn không cần sử dụng chuột.
* Áp dụng được kiến thức vào trong trò chơi.
* Tiếp nhận thêm được nhiều kiến thức mới.
* Trò chơi có thêm nhiều cải tiến mới, tăng độ khó và hứng thú cho người chơi.

1. HẠN CHẾ:
   1. Quá trình thực hiện:
   2. Ứng dụng:
2. HƯỚNG PHÁT TRIỂN:

TÀI LIỆU THAM KHẢO

* Chương 1: Giới thiệu đồ án

Link: <https://strategywiki.org/wiki/Battle_City>

* Chương 2: Cơ sở lý thuyết:
  + Ngôn ngữ lập trình C#:
* <https://vietjack.com/csharp/csharp_la_gi.jsp>
* <http://it.die.vn/n/ngon-ngu-lap-trinh-c-thang/>
* <https://freetuts.net/c-sharp-la-gi-tong-quan-ve-c-sharp-1045.html>
  + Lý thuyết áp dụng: Thuật toán A\*:

Link:

* <https://www.geeksforgeeks.org/a-search-algorithm/>
* <https://www.stdio.vn/articles/thuat-giai-a-262>

# LỜI CẢM ƠN

Đầu tiên, Nhóm xin gửi lời cảm ơn chân thành đến tập thể quý Thầy, Cô Trường Đại học Công nghệ Thông tin – Đại học Quốc gia Thành phố Hồ Chí Minh.

Đặc biệt, nhóm xin gửi lời cảm ơn và lòng biết ơn sâu sắc đến cô Huỳnh Thị Thanh Thương – giảng viên môn Trí tuệ nhân tạo. Cô đã trực tiếp hướng dẫn và giải đáp các thắc mắc cho nhóm trong quá trình làm đồ án.

Trong suốt thời gian thực hiện đề tài, nhóm đã vận dụng được những kiến thức nền tảng đã được tích luỹ từ các buổi học và phối hợp với việc tự học hỏi và nghiên cứu kiến thức mới. Đồng thời, những ý kiến đóng góp của cô đã giúp đồ án của nhóm ngày càng hoàn thiện hơn. Nhờ đó, đồ án đã thuận lợi hoàn thành.

Nhóm tin tưởng rằng những kiến thức này sẽ là thành trang cho mỗi thành viên trong tương lai.

*Thành phố Hồ Chí Minh, ngày 04 tháng 07 năm 2018*