**BỘ CÔNG THƯƠNG**

**TRƯỜNG ĐẠI HỌC CÔNG THƯƠNG TP.HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**

-----o0o----

****

**BÀI TẬP TIỂU LUẬN HỌC PHẦN: CẤU TRÚC RỜI RẠC**

**TÊN ĐỀ TÀI: Mini-Game tìm đường đi trên đồ thị sử dụng thuật toán dijkstra**

**NHÓM: 11**

**Thành phố Hồ Chí Minh, tháng 11 năm 2023**

**BỘ CÔNG THƯƠNG**

**TRƯỜNG ĐẠI HỌC CÔNG THƯƠNG TP.HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**

-----o0o----

****

**TÊN ĐỀ TÀI:**

|  |  |
| --- | --- |
| **Nhóm: 11**  Trưởng nhóm:  Nguyễn Hoàng Long 2001222438  Thành viên:  1. Tô Minh Lợi 2001222485  2. Ngô Thanh Tâm 2001202230  3. Thái Ngọc Thạch 2001224637 | **Giảng viên hướng dẫn:**  **Nguyễn Thị Thùy Trang** |

**Thành phố Hồ Chí Minh, tháng 11 năm 2023**

|  |  |  |  |
| --- | --- | --- | --- |
| **Họ và Tên** | **PCCV** | **Mức Độ Hoàn Thành (%)** | **Tỉ Lệ Đóng Góp(%)** |
| Nguyễn Hoàng Long | Code chính + word | 100% | 31% |
| Tô Minh Lợi | Hỗ trợ code | 100% | 23% |
| Ngô Thanh Tâm | Hỗ trợ code | 100% | 23% |
| Thái Ngọc Thạch | Hỗ trợ code | 100% | 23% |

**MỤC LỤC**

[**CHƯƠNG I: GIỚI THIỆU** 1](#_Toc152442732)

[1. Lý do chọn đề tài 1](#_Toc152442733)

[2. Mục tiêu và đối tượng nghiên cứu 1](#_Toc152442734)

[2.1 Mục tiêu nghiên cứu 1](#_Toc152442735)

[2.2 Đối tượng nghiên cứu 1](#_Toc152442736)

[3. Phạm vi nghiên cứu 2](#_Toc152442737)

[4. Phương pháp nghiên cứu 2](#_Toc152442738)

[**CHƯƠNG II: THUẬT TOÁN DIJKSTRA** 4](#_Toc152442739)

[1. Lịch sử 4](#_Toc152442740)

[2. Thuật toán Dijkstra 5](#_Toc152442741)

[3. Ứng dụng của thuật toán Dijkstra 6](#_Toc152442742)

[**CHƯƠNG III: ỨNG DỤNG THUẬT TOÁN DIJKSTRA CHO MINIGAME TÌM ĐƯỜNG ĐI TRÊN ĐỒ THỊ** 8](#_Toc152442743)

[1. Khái quát chương trình 8](#_Toc152442744)

[2. Mô tả chi tiết 9](#_Toc152442745)

[PHẦN KẾT LUẬN 21](#_Toc152442746)

# **CHƯƠNG I: GIỚI THIỆU**

1. Lý do chọn đề tài

Lý thuyết đồ thị nói chung và các thuật toán tìm kiếm đường đi ngắn nhất nói riêng là một phần gần như bất kỳ một người học công nghệ thông tin nào cũng phải biết. Bởi lẻ tính ứng dụng và nghiên cứu của các thuật toán này là rất cao. Chúng không chỉ giúp người học hiểu được về đồ thị có hướng hay vô hướng, mà còn hiểu đường cách mà một bài toán thực tế có thể được biểu diễn dưới dạng một đồ thị. Vì vậy, việc hiểu lý thuyết đồ thị sẽ giúp ích cho chúng ta rất nhiều trong việc giải các bài toán tìm đường đi và các bài toán nhiên liệu, chi phí thực tiễn ngày nay. Nổi bật trên hết là thuật toán Dijkstra, là một trong các thuật toán lý thuyết đồ thị quan trọng nhất, được sử dụng rộng rãi và khả năng ứng dụng cao của thuật toán. Cho nên, bài tiểu luận này sẽ tập trung nghiên cứu về cách thuật toán Dijkstra hoạt động cũng như khả năng ứng dụng của nó.

1. Mục tiêu và đối tượng nghiên cứu
   1. Mục tiêu nghiên cứu

Mục tiêu nghiên cứu của tiểu luận là làm rõ sơ bộ và cách vận hành của thuật toán Dijkstra. Hiểu được tầm quan trọng của Dijkstra hay các thuật toán lý thuyết đồ thị khác trong nhiều lĩnh vực khác nhau. Từ đó ứng dụng nó vào các bài toán lớn nhằm giải quyết vấn đề tìm đường đi sao cho tiết kiệm chi phí nhất.

### Đối tượng nghiên cứu

Đối tượng nghiên cứu chính là thuật toán Dijkstra và khả năng ứng dụng của Dijkstra trong nhiều lĩnh vực khác nhau. Song song với đó là khả năng mở rộng và giải quyết các bài toán lớn khi đưa về dạng đồ thị một cách hiệu quả.

1. Phạm vi nghiên cứu

* Về lý thuyết đồ thị:

Tổng quan về lý thuyết đồ thị, các khái niệm cơ bản, loại đồ thị (vô hướng, có hướng, trọng số, v.v.).

Các thuật ngữ và khái niệm quan trọng như đỉnh, cạnh, chu trình, đồ thị liên thông, v.v.

* Về thuật toán Dijkstra:

Chi tiết về thuật toán Dijkstra, cách nó hoạt động và đặc điểm của nó.

So sánh với các thuật toán khác trong lý thuyết đồ thị, như Bellman-Ford, Floyd-Warshall.

* Ứng dụng của thuật toán Dijkstra:

Trong mạng lưới đường đi (network routing) và các vấn đề liên quan.

Trong hệ thống giao thông và quản lý đường đi.

Trong các ứng dụng mô phỏng và tối ưu hóa.

1. Phương pháp nghiên cứu

* Phân tích lý thuyết:

Tiến hành một phân tích sâu sắc về lý thuyết đồ thị và thuật toán Dijkstra.

So sánh và đối chiếu với các nghiên cứu, phát hiện và xu hướng hiện tại trong lĩnh vực này.

* Mô phỏng và thực nghiệm:

Tạo các mô hình mô phỏng để minh họa cách thuật toán Dijkstra hoạt động trong các tình huống cụ thể.

Áp dụng thuật toán vào các bài toán thực tế và đánh giá hiệu suất.

* Tối ưu hóa thuật toán:

Nghiên cứu về cách tối ưu hóa thuật toán Dijkstra cho các trường hợp đặc biệt hoặc dạng đồ thị cụ thể.

Đề xuất và thử nghiệm các biến thể hoặc cải tiến để cải thiện hiệu suất của thuật toán.

* Đánh giá và so sánh:

So sánh hiệu suất của thuật toán Dijkstra với các thuật toán khác trong các tình huống cụ thể.

Xem xét ưu nhược điểm của thuật toán trong các bối cảnh khác nhau và đề xuất cách cải thiện.

**CHƯƠNG II: THUẬT TOÁN DIJKSTRA**

1. Lịch sử

Thuật toán Dijkstra, mang tên của nhà khoa học máy tính người Hà Lan Edsger Dijkstra vào năm 1956 và ấn bản năm 1959, là một thuật toán giải quyết bài toán đường đi ngắn nhất từ một đỉnh đến các đỉnh còn lại của đồ thị có hướng không có cạnh mang trọng số không âm. Thuật toán thường được sử dụng trong định tuyến với một chương trình con trong các thuật toán đồ thị hay trong công nghệ Hệ thống định vị toàn cầu (GPS).

Edsger Dijkstra đã chia sẻ trong một cuộc phỏng vấn với Philip L. Frana, Communications of the ACM, 2001: “Đâu là con đường ngắn nhất để đi từ Rotterdam đến Groningen, hay nói chung: từ thành phố này đến thành phố nhất định. Đây là thuật toán tìm đường đi ngắn nhất, mà tôi đã thiết kế trong khoảng hai mươi phút. Vào một buổi sáng đi mua sắm ở Amsterdam cùng vị hôn thê trẻ tuổi với sự mệt mỏi, chúng tôi ngồi trên sân thượng để uống một tách cà phê và tôi chỉ nghĩ liệu tôi có thể làm điều này không, và sau đó tôi đã thiết kế thuật toán tìm đường đi ngắn nhất. Như tôi đã nói, đó là một phát minh trong hai mươi phút. Trên thực tế, nó đã được xuất bản vào năm 1959, ba năm sau đó. Các ấn phẩm vẫn có thể đọc được, quả thật như vậy, và khá đẹp. Một trong những lý do khiến nó rất hay là tôi đã thiết kế nó mà không cần bút chì và giấy. Sau này tôi đã học được rằng một trong những lợi thế của việc thiết kế mà không cần bút chì và giấy là bạn gần như buộc phải tránh đi mọi sự phức tạp có thể tránh được. Cuối cùng, thuật toán đó đã trở thành một trong những nền tảng giúp tôi nổi tiếng, với sự kinh ngạc lớn của tôi.”

Dijkstra đã nghĩ về bài toán đường đi ngắn nhất khi làm việc tại Trung tâm Toán học ở Amsterdam năm 1956 với tư cách là một lập trình viên để chứng minh khả năng của một máy tính mới có tên ARMAC. Mục tiêu của ông là chọn cả một bài toán và giải pháp (sẽ được tạo bởi máy tính) mà những người không thuần tính toán vẫn có thể hiểu được. Ông đã thiết kế thuật toán đường đi ngắn nhất và sau đó triển khai nó cho ARMAC bằng bản đồ giao thông được đơn giản hóa một chút của 64 thành phố ở Hà Lan[3]. Một năm sau, ông gặp một vấn đề khác từ các kỹ sư phần cứng làm việc trên máy tính tiếp theo của học viện: giảm thiểu lượng dây cần thiết để kết nối các chân trên bảng điều khiển phía sau của máy. Như một giải pháp, ông đã phát hiện lại thuật toán Prim (được biết đến trước đó với Jarník, và cũng được Prim khám phá lại). Dijkstra đã xuất bản thuật toán vào năm 1959, 2 năm sau Prim và 29 năm sau Jarník.

1. Thuật toán Dijkstra

Thuật toán Dijkstra là một thuật toán quan trọng trong lĩnh vực đồ thị và tìm đường đi. Thuật toán này được sử dụng để tìm đường đi ngắn nhất giữa hai đỉnh trong một đồ thị có trọng số dương. Thuật toán này đặc biệt hiệu quả khi áp dụng cho đồ thị có cấu trúc đơn đồ thị, không có chu trình âm.

* Dưới đây là mô tả cơ bản của thuật toán Dijkstra:

+ Khởi tạo: Gán một giá trị khoảng cách (distance) ban đầu cho tất cả các đỉnh trong đồ thị. Đối với đỉnh nguồn, giá trị khoảng cách là 0, còn đối với các đỉnh khác là vô cùng. Tạo một tập hợp các đỉnh chưa xử lý, thường được biểu diễn bằng hàng đợi ưu tiên (priority queue) dựa trên khoảng cách từ nguồn.

+ Lặp: Lặp qua tất cả các đỉnh chưa xử lý. Chọn đỉnh có khoảng cách nhỏ nhất từ nguồn. Đỉnh này được xác định là "đỉnh hiện tại." Duyệt qua tất cả các đỉnh kề của đỉnh hiện tại và cập nhật khoảng cách của chúng nếu có một đường đi ngắn hơn.

+ Đánh dấu đỉnh đã xử lý: Sau khi cập nhật khoảng cách của tất cả các đỉnh kề, đánh dấu đỉnh hiện tại là đã xử lý (đã có khoảng cách ngắn nhất từ nguồn).

+ Lặp lại: Lặp lại quá trình cho đến khi tất cả các đỉnh đều đã được xử lý hoặc hàng đợi ưu tiên trống.

Thuật toán Dijkstra đảm bảo tìm ra khoảng cách ngắn nhất từ đỉnh nguồn đến tất cả các đỉnh còn lại trong đồ thị. Tuy nhiên, nếu đồ thị chứa trọng số âm hoặc có chu trình âm, thuật toán này có thể không hoạt động đúng. Trong trường hợp đồ thị có trọng số âm, thuật toán Bellman-Ford thường được sử dụng.

1. Ứng dụng của thuật toán Dijkstra

Thuật toán Dijkstra có nhiều ứng dụng trong thực tế, chủ yếu là trong lĩnh vực mạng lưới, định tuyến, giao thông vận tải và quản lý tài nguyên. Dưới đây là một số ứng dụng phổ biến của thuật toán Dijkstra:

* Định tuyến trong mạng lưới: Trong mạng máy tính hoặc mạng di động, Dijkstra được sử dụng để xác định đường đi ngắn nhất giữa hai thiết bị hoặc điểm cuối.
* Quản lý giao thông: Trong hệ thống giao thông, Dijkstra có thể được áp dụng để tìm đường đi ngắn nhất giữa hai địa điểm trên bản đồ, giúp định tuyến giao thông hiệu quả.
* Quản lý tài nguyên trong mạng lưới điện: Trong mạng điện, thuật toán Dijkstra có thể được sử dụng để tìm đường đi ngắn nhất giữa các trạm biến áp, giúp tối ưu hóa quá trình truyền tải điện năng.
* Hệ thống đường sắt và giao thông công cộng: Trong lĩnh vực vận tải công cộng, Dijkstra giúp xác định đường đi ngắn nhất cho các tuyến xe buýt, đường sắt, hoặc các phương tiện giao thông khác để giảm thời gian và tối ưu hóa tài nguyên.
* Tìm đường đi trong ứng dụng bản đồ và điều hướng: Trong các ứng dụng điều hướng như Google Maps, thuật toán Dijkstra được sử dụng để tìm đường đi ngắn nhất giữa hai địa điểm.
* Mô phỏng đường đi của robot hoặc drone: Trong lĩnh vực robot và drone, thuật toán Dijkstra có thể được sử dụng để lập kế hoạch đường đi an toàn và hiệu quả để di chuyển từ điểm A đến điểm B tránh các chướng ngại vật.
* Mạng nước và xử lý nước thải: Trong lĩnh vực quản lý nước, thuật toán Dijkstra có thể được sử dụng để tối ưu hóa quá trình truyền tải nước trong một hệ thống đường ống hoặc kênh.

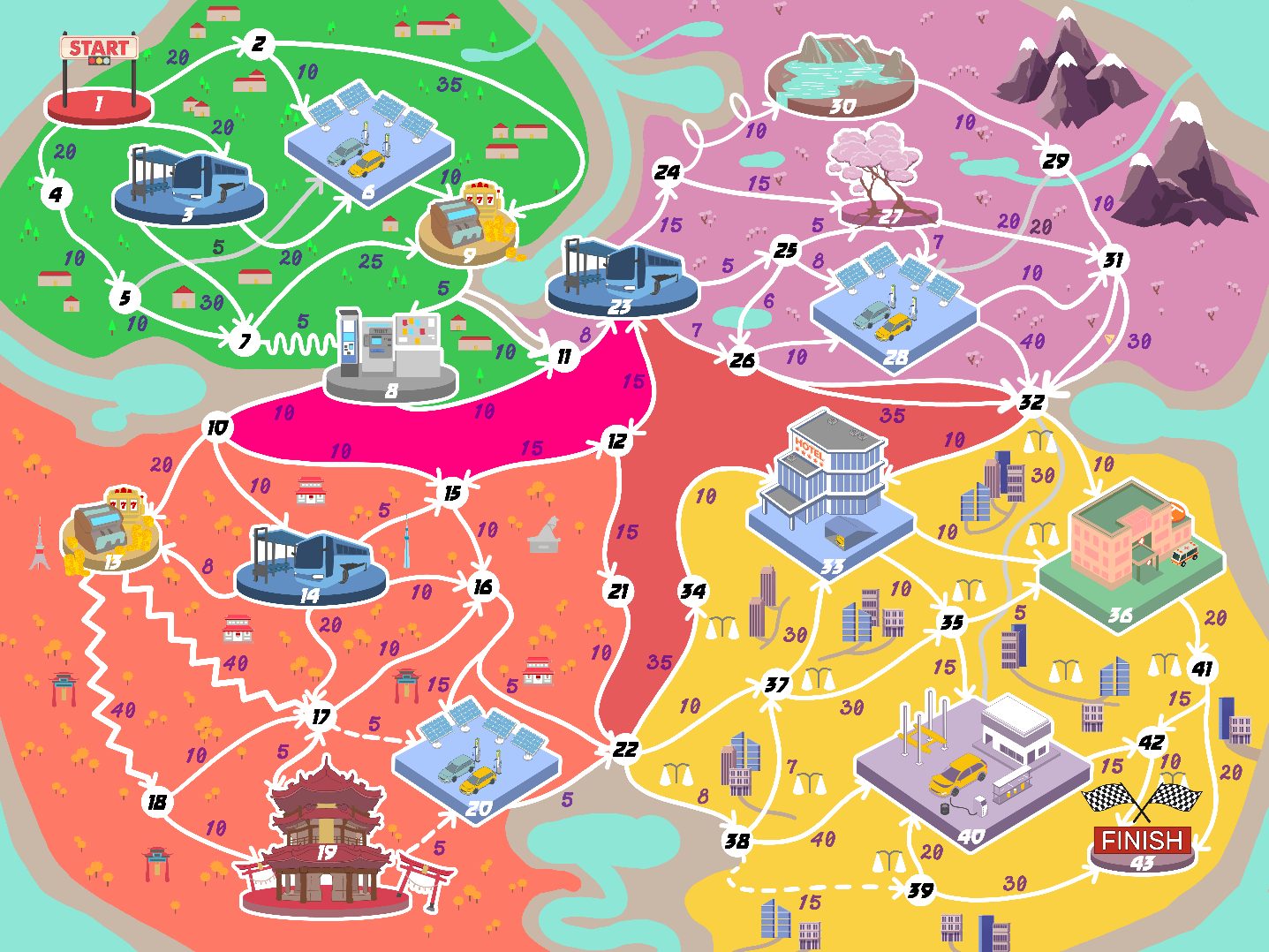
Những ứng dụng này chỉ là một số ví dụ cơ bản, và thuật toán Dijkstra có thể được áp dụng trong nhiều lĩnh vực khác nhau để giải quyết các vấn đề liên quan đến tìm đường đi ngắn nhất.

# **CHƯƠNG III: ỨNG DỤNG THUẬT TOÁN DIJKSTRA CHO MINIGAME TÌM ĐƯỜNG ĐI TRÊN ĐỒ THỊ**

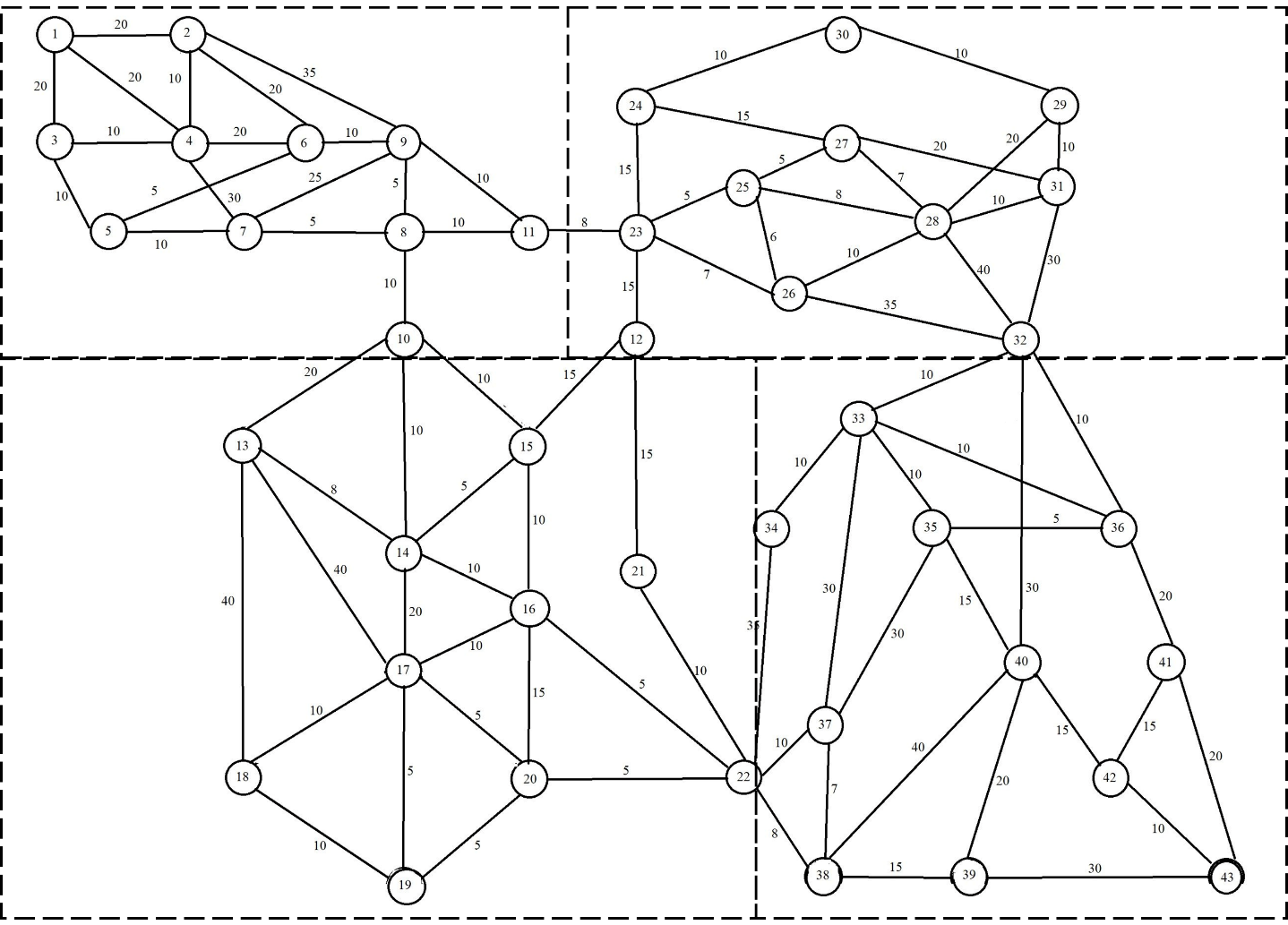
1. Khái quát chương trình

Chương trình sử dụng thuật toán Dijkstra làm cốt lỗi, từ đó phát triển lên thành một trò chơi với mục tiêu chính là sử dụng thuật toán Dijkstra để tìm đường đi ngắn nhất đồng thời khiến bản thân người dùng tự tìm đường đi ngắn nhất để hoàn thành chương trình. Cụ thể, trò chơi khiến người chơi trở thành tài xế của một chuyến xe khách với đích đến là điểm cuối cùng được đánh dấu trên đồ thị. Mục tiêu của người chơi là về đích với điểm số cao nhất có thể.

* Bản đồ của trò chơi:



* Bản đồ trò chơi (dưới dạng đồ thị):



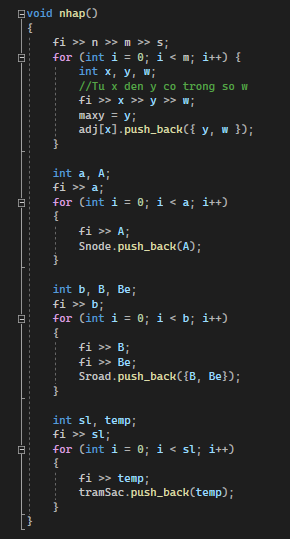
Bằng việc thúc đẩy người chơi phải tìm đường đi sao cho đạt điểm số cao nhất, chương trình giúp người chơi hiểu rõ hơn về bài toán tìm đường đi ngắn nhất hay tiết kiệm chi phí nhất, đồng thời sử dụng thuật toán Dijkstra xuyên suốt chương trình với mục đích giúp người dùng hiểu và nắm bắt được cách thuật toán này được triển khai.

1. Mô tả chi tiết

Chương trình được chia ra làm nhiều phần, mỗi phần có các hàm hỗ trợ cũng như tính toán dọc theo quá trình chơi của người dùng. Trong đó, chương trình được chia làm 3 phần chính, phần thứ nhất thiết lập các tài nguyên có sẵn ban đầu – ngẫu nhiên giữa mỗi lần chơi , phần thứ hai do người chơi quản lý việc di chuyển trên đồ thị và kiểm soát tài nguyên khi đi qua các điểm, các đoạn đường đặc biệt, phần cuối cùng sẽ thực hiện việc tổng kết những thay đổi dựa trên tiến trình chơi từ đó đưa ra số điểm cụ thể và xếp hạng những người chơi. Các phần được cài đặt và mô tả chi tiết như sau:

* Phần thiết lập ban đầu:

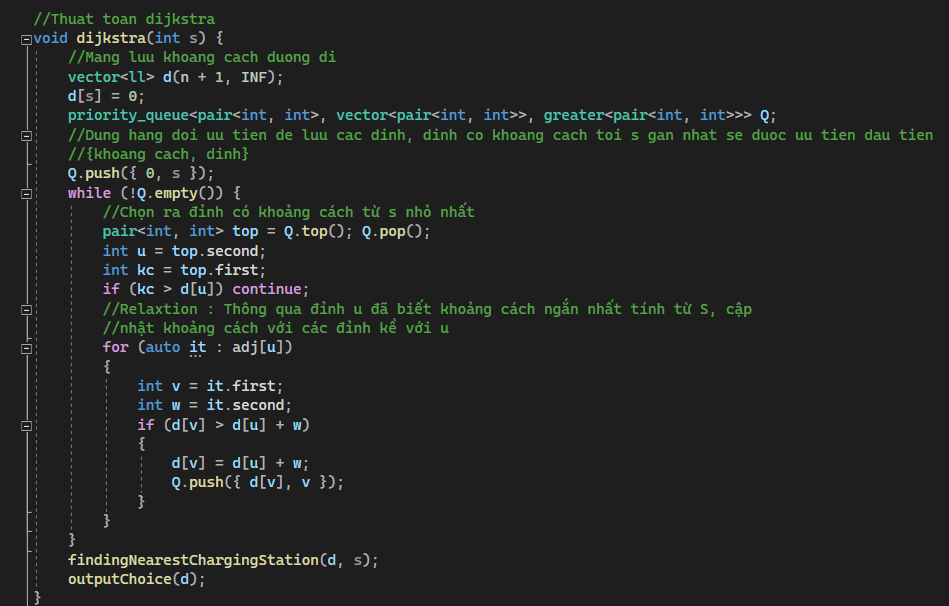
+ Hàm nhập đồ thị và các thông số:



Đồ thị nhập từ file lưu sẵn sau đó lưu vào một mảng vector pair có dạng

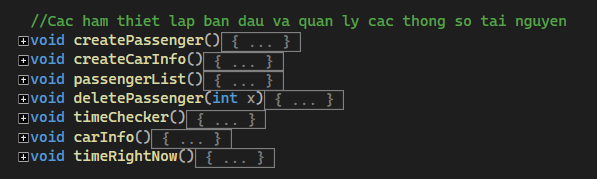
vector<pair<int, int>> adj[maxn];

+ Thuật toán Dijkstra:



Thuật toán Dijkstra sử dụng hàng đợi ưu tiên để lưu danh sách các đỉnh sao cho đỉnh có trộng số (khoảng cách đến đỉnh ban đầu) nhỏ nhất được ưu tiên cao nhất, sau đó tính toán và tích lũy khoảng cách từ đỉnh ban đầu (hay hiện tại) đến tất cả các đỉnh còn lại.

+ Các hàm thiết lập tài nguyên và cập nhật tài nguyên



Trong đó các thông số ban đầu được thiết lập nhự sau:

- Có 5/10 hành khách muốn đến biển

- Mỗi hành khách có ngẫu nhiên số tiền từ 150k-300k

- Tiền vé xe bus mỗi người là 50k

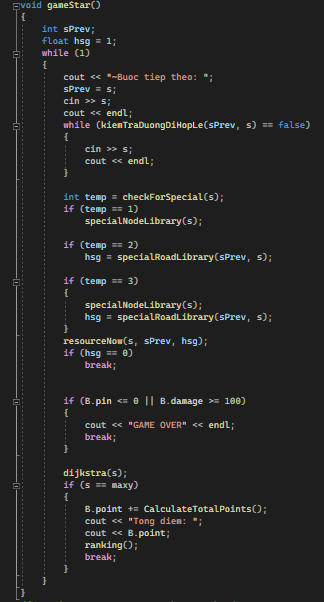
- Có ngẫu nhiên từ 1 -> 5 hành khách có thẻ thành viên

- Điện còn lại trong xe là 100%

- Độ hao mòn của xe là 0%

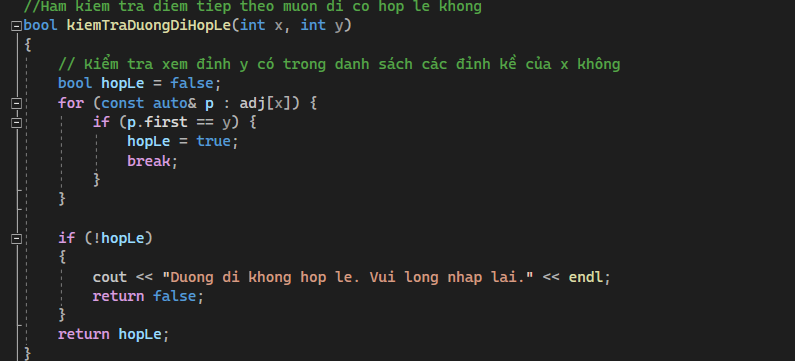
- Xe khởi hành lúc 6h

* Phần thao tác người dùng:

+ Hàm bắt đầu trò chơi

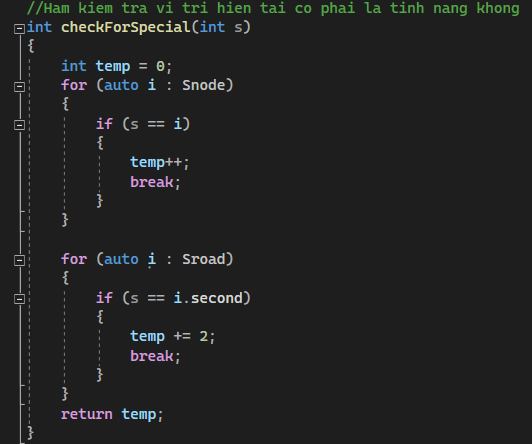
Hàm đảm nhiệm vai trò quản lý vị trí hiện tại và trước đó của người chơi, kiểm tra mỗi bước di chuyển của người chơi có hợp lệ không, có phải là điểm đặc biệt trên đồ thị không, có phải là đoạn đường đặc biệt trên đồ thị không. Lập lại cho tới khi người chơi về trị trí trong đồ thị hoặc tài nguyên người chơi đang có không còn đủ để tiếp tục.

+ Hàm kiểm tra đường đi hợp lệ:



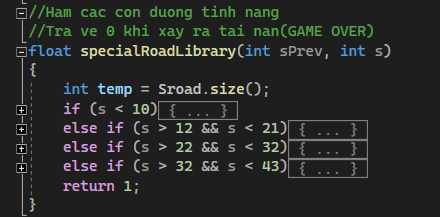
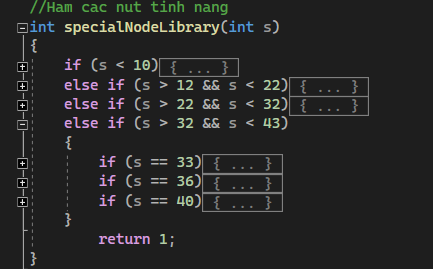
Hàm kiểm tra điểm đến tiếp theo của người chơi có hợp lệ không (có đường đi đến điểm đó trong đồ thị không), yêu cầu người chơi nhập lại nếu đijểm nhập không hợp lệ.

+ Hàm kiểm tra vị trí, đường đi đặc biệt



Hàm kiểm tra vị trí tiêp theo có phải là vị trí hay đường đi đặc biệt không, trả về 1 khi là điểm đặc biệt, trả về 2 khi là đường đi đặc biệt, trả về 3 khi vừa là điểm đặc biệt vừa là đường đi đặc biệt.

+ Hàm xử lý các nút đặc biệt và hàm xử lý các đường đi đặc biệt:



Hàm specialNodeLibrary sẽ xử lý các nút đặc biệt trên đồ thị, hàm specialRoadLibrary sẽ xử lý các đường đi đặc biệt trên đồ thị, trong đó hàm đường đi đặc biệt sẽ trả về 0 khi có sự kiện diễn ra khiến trò chơi không thể tiếp tục và trả về hệ số đường đi ảnh hưởng đến trọng số đường đi khi qua các đoạn đường đặc biệt.

Các điểm và đường đi đặc biệt được mô tả như sau:

+ Trạm đón khách: Mỗi trạm có 5 khách với 3 nhóm muốn đi núi, quốc lộ hay biển

người chơi được chọn nhận khách nào lên xe, xe chở tối đa 10 người cùng lúc

+ Trạm sạc: Sạc lại điện của xe, mất 1 phút cho mỗi 1% pin sạc lại, người chơi có thể chọn sạc bao nhiêu tùy thích

+ Trạm thu phí:

\* Trạm thu phí ở KV1 mất 50k mỗi người (không tính tài xế)

\* Trạm thu phí ở KV2 3 mất 75k mỗi người (Không tính tài xế)

+ Trạm tích điểm: Tích điểm cho thẻ thành viên

\* Thẻ thành viên chưa được tích điểm sẽ giảm 10% phí ở trạm thu phí

\* Thẻ thành viên đã được tích điểm sẽ giảm 25% phí ở trạm thu phí

+ Khu du lịch sẽ tăng tinh thần hành khách

\* Nếu số lượng nữ giới > nam giới: Tăng 20% tinh thần toàn bộ hành khách

\* Nếu số lượng nữ giới < nam giới: Tăng 10% tinh thần toàn bộ hành khách

+ Suối nước nóng sẽ tăng tinh thần hành khách 50%

+ Điểm và đường có quang cảnh đẹp sẽ tăng 20% tinh thần khách hàng

+ Bệnh viện sẽ tăng 30 điểm tinh thần cho khách hàng (50k mỗi người)

+ Trạm sửa xe sẽ giảm 50% hao mòn xe

+ Hotel cho phép giảm 1 lượng khách hàng tùy ý (Giảm bất kỳ ai)

\* -50 điểm cho mỗi khách hàng bị trả tại khách sạn

+ Đường kẹt xe sẽ kẹt xe vào lúc 10h-12h sáng và 3h-5h chiều (Tăng x2 thời gian cần để qua đoạn đường này)

+ Cao tốc:

\* kẹt cao tốc lúc 10h-12h sáng và 3h-5h chiều (Tăng x2 thời gian cần để qua đoạn đường này)

\* không kẹt cao tốc các khung giờ còn lại (Giảm x2.5 thời gian cần để qua đoạn đường này)

+ Đường có thể xảy ra tai nạn

\* Có 25% tỉ lệ xảy ra tai nạn nếu số lượng nam > số lượng nữ làm bị thương ngẫu nhiên số hành khách trên xe

\* Có 30% tỉ lệ xảy ra tai nạn nếu số lượng nam = số lượng nữ làm bị thương ngẫu nhiên số hành khách trên xe

\* Có 35% tỉ lệ xảy ra tia nạn nếu số lượng nam < số lượng nữ làm bị thương ngẫu nhiên số hành khách trên xe

\* Nếu xảy ra tai nạn tăng (số lượng hành khách x 5)% hao mòn xe

+ Đường đèo:

\* Có 20% tỉ lệ xảy ra tai nạn nếu số lượng nam > số lượng nữ làm bị thương ngẫu nhiên số hành khách trên xe

\* Có 25% tỉ lệ xảy ra tai nạn nếu số lượng nam = số lượng nữ làm bị thương ngẫu nhiên số hành khách trên xe

\* Có 30% tỉ lệ xảy ra tia nạn nếu số lượng nam < số lượng nữ làm bị thương ngẫu nhiên số hành khách trên xe

\* Nếu xảy ra tai nạn tăng (số lượng hành khách x 8)% hao mòn xe

\* Nếu không xảy ra tai nạn tăng 10% hao mòn xe

\*Giảm x2 thời gian cần để qua đèo

+ Đường hầm:

\* Đường hầm nguy hiểm(Chọn đi nhanh hay chậm):

` Nếu đi nhanh: Có 50% khả năng xảy ra tai nạn

` Nếu đi chậm: Có 25% xảy ả tai nạn

\* Đường hầm không nguy hiểm

` Nếu đi nhanh: Có 5% khả năng xảy ra tai nạn

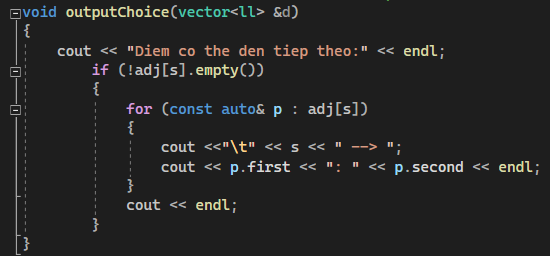
` Nếu đi chậm: có 1% khả năng xảy ra tai nạn

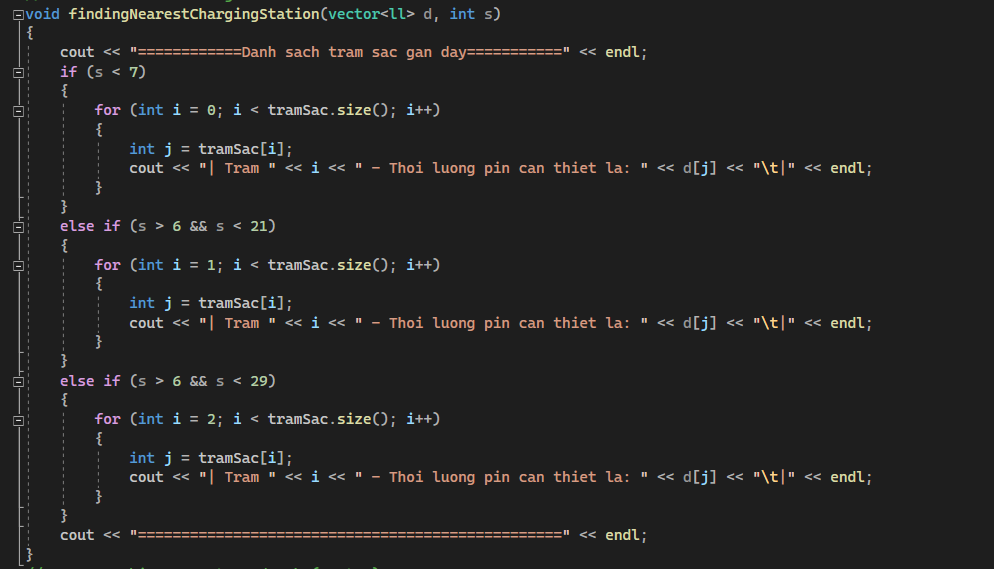
\* Nếu xảy ra tai nạn, lập tức thua

\* Nếu không xảy ra tai nạn, giảm x3 thời gian cần để qua hầm

+ Đường cát: Hao mòn xe tăng 10%, tinh thần hành khách tăng 20%

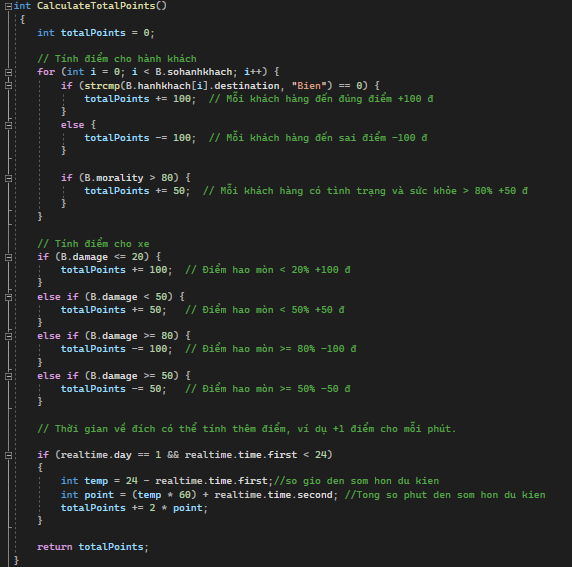
+ Hàm in ra màn hình danh sách điểm có thể đến tiếp theo trên đồ thị:



+ Ham tính toán thời lượng pin cần thiết để đến các trạm sạc gần nhất có thể đến:  


* Phần kết chương trình:

+ Hàm tính toán các thông số tài nguyên và đưa ra điểm tổng



Trong đó các thống số sẽ được tính điểm như sau:

- Điểm hành khách:

+ Mỗi khách hàng đến đúng điểm: +100đ

+ Mỗi khách hàng đến sai điểm: -100đ

+ Mỗi khách hàng có tình trạng và sức khỏe >80%: +50đ

- Điểm xe:

+ Độ hao mòn <50%: +50đ

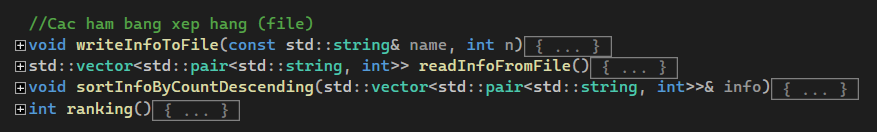
+ Độ hao mòn <=20%: +100đ

+ Độ hao mòn >=50%: -50đ

+ Độ hao mòn >= 80%: -100đ

+ Thời gian về đích: +60đ mỗi phút đến sớm hơn 24h ngày thứ nhất

+ Nhóm các hàm lưu lại và in ra bảng xếp hạng những lần chơi kèm theo điểm



# PHẦN KẾT LUẬN

Qua bài tiểu luận trên, chúng ta đã phần nào hiểu được cách thuật toán Dijkstra hoạt động cũng như ứng dụng của nó nói riêng và các giải thuật lý thuyết đồ thị khác nói chung. Có thể nói các thuật toán đồ thị này không chỉ là bước đệm cho các lập trình viên mới mà còn là các công cụ mạnh mẽ cho các chuyên gia trong các lĩnh vực khác nhau giải quyết những bài toán khó về quản lý tài nguyên và tìm đường đi. Hơn hết, chương trình minh họa cho ứng dụng của Dijkstra ở trên chính là điển hình cho một trong nhiều ứng dụng của lý thuyết đồ thị. Nhóm chúng tôi xin được gửi lời cảm chân thành đến tất cả những ai đã hỗ trợ và tiếp sức cho nhóm xây dựng bài tiểu luận trên!

Tài Liệu Tham Khảo

1. GitHub – mburst/Dijkstra algorithm:

(<https://github.com/mburst/dijkstras-algorithm>)

1. Youtube – 28tech/#21 [Lý thuyết đồ thị | Toán rời rạc]. Thuật Toán Dijkstra | Thuật Toán Tìm Đường Đi Ngắn Nhất:

(<https://www.youtube.com/watch?v=JqOPBodZmLk&t=443s>)

1. Wikipedia – Nhiều tác giả/Thuật toán Dijkstra:

(https://vi.wikipedia.org/w/index.php?title=Thuật\_toán\_Dijkstra&action=history)