**Chương I. Tổng quan**

1. **Mô tả bài toán**

Bài toán tập trung vào việc xác định và tìm chu trình Euler hoặc nửa Euler trên đồ thị thông qua các công cụ trực quan được triển khai trên nền tảng web. Đồ thị được xây dựng và xử lý bằng ba phương pháp chính: tạo ngẫu nhiên, vẽ thủ công, và nhập từ file. Sau khi đồ thị được tạo, hệ thống sẽ xác định loại đồ thị (Euler, nửa Euler, hoặc không là Euler) và tìm chu trình hoặc đường đi Euler theo quy tắc ưu tiên bảng chữ cái.

1. **Mục tiêu cần đạt được**

* Xây dựng ứng dụng web hỗ trợ tạo, phân tích và xử lý đồ thị.
* Kiểm tra và phân loại đồ thị dựa trên các điều kiện lý thuyết Euler.
* Triển khai thuật toán để tìm chu trình Euler theo thứ tự ưu tiên bảng chữ cái.
* Đảm bảo ứng dụng hoạt động hiệu quả, chính xác và thân thiện với người dùng.

1. **Hướng giải quyết**

* Xây dựng giao diện: Thiết kế giao diện web trực quan với các chức năng hỗ trợ người dùng tạo đồ thị bằng ba phương pháp.
* Thuật toán xử lý: Sử dụng các thuật toán kiểm tra bậc đỉnh và tìm chu trình Euler, nửa Euler dựa trên điều kiện lý thuyết. Đảm bảo tính chính xác và hiệu quả khi tìm chu trình hoặc đường đi.
* Xuất kết quả: Hiển thị loại đồ thị (Euler, nửa Euler, không là Euler) và chu trình hoặc đường đi Euler theo quy tắc ưu tiên.

1. **Kế hoạch thực hiện**

* **Giai đoạn 1: Nghiên cứu và thiết kế hệ thống**
* Phân tích yêu cầu bài toán và thu thập tài liệu liên quan.
* Thiết kế giao diện và xác định cấu trúc dữ liệu phù hợp.
* **Giai đoạn 2: Phát triển ứng dụng**
* Xây dựng chức năng vẽ đồ thị: ngẫu nhiên, thủ công và từ file.
* Triển khai thuật toán kiểm tra và tìm chu trình hoặc đường đi Euler.
* **Giai đoạn 3: Kiểm thử và tối ưu hóa**
* Kiểm tra độ chính xác của ứng dụng với các trường hợp đồ thị khác nhau.
* Tối ưu hiệu năng xử lý và cải thiện trải nghiệm người dùng.
* **Giai đoạn 4: Hoàn thiện và báo cáo**
* Hoàn chỉnh ứng dụng, trình bày trên web.
* Soạn thảo báo cáo, tổng kết kết quả nghiên cứu và hướng phát triển.

**Chương II.** **Cơ sở lý thuyết**

1. **Định nghĩa đồ thị**

Đồ thị là một cấu trúc toán học rời rạc, bao gồm hai thành phần chính là các đỉnh và các cạnh, dùng để mô tả mối liên kết giữa các đỉnh. Các cạnh có thể có hướng hoặc không có hướng, và dựa trên đặc điểm này cũng như cách kết nối các đỉnh, đồ thị được phân thành nhiều loại khác nhau.A computer icons with a white background

Description automatically generated with medium confidence

Một **đơn đồ thị vô hướng** G=(V,E) là một đồ thị mà tập V (tập đỉnh) không rỗng và mỗi cạnh trong tập E (tập cạnh) là một cặp không có thứ tự của các đỉnh phân biệt.A diagram of a city

Description automatically generated

**Đa đồ thị vô hướng** cũng có tập đỉnh V và tập cạnh E, nhưng cho phép các cạnh bội (hay song song), tức là nhiều cạnh có thể cùng kết nối một cặp đỉnh. Điều này khiến đa đồ thị có tính linh hoạt hơn, nhưng không phải đa đồ thị nào cũng là đơn đồ thị.A diagram of a river

Description automatically generated

1. **Các thuật ngữ cơ bản**

Hai đỉnh uu và vv trong đồ thị G=(V,E) được gọi là liền kề nếu có cạnh nối chúng, ký hiệu (u,v) ∈ E. Cạnh này cũng được gọi là cạnh liên thuộc với hai đỉnh uu và v. Bậc của một đỉnh vv, ký hiệu deg(v) là số cạnh liên thuộc với đỉnh đó. Trong trường hợp một cạnh là khuyên (nối đỉnh với chính nó), bậc của đỉnh sẽ tăng thêm hai đơn vị. Các đỉnh có bậc bằng một được gọi là đỉnh treo, còn các đỉnh không có cạnh liên thuộc được gọi là đỉnh cô lập.

Đỉnh v gọi là đỉnh treo nếu deg(v) = 1 và gọi là đỉnh cô lập nếu deg(v) = 0.

**Ví dụ:**

A diagram of a diagram

Description automatically generated

Ta có deg(v1) = 3, deg(v2) = 4, deg(v3) = 5, deg(v4) = 1, deg(v5) = 3, deg(v6) = 2, deg(v7) = 2. Đỉnh v8 là đỉnh cô lập và v4 là đỉnh treo.

**Mệnh đề 1.8** Cho đồ thị G = (V, E). Khi đó

A black symbol with a white background

Description automatically generated

**Hệ quả 1.9** Số đỉnh bậc lẻ của một đồ thị là một số chẵn.

**Mệnh đề 1.10** Trong một đơn đồ thị, luôn tồn tại hai đỉnh có cùng bậc.

**Đường đi, chu trình, đồ thị liên thông**

Đường đi trong đồ thị vô hướng là một dãy liên tiếp các đỉnh, bắt đầu từ đỉnh u và kết thúc tại đỉnh v, trong đó mỗi cặp đỉnh liên tiếp trong dãy đều có cạnh nối. Nếu điểm bắt đầu và điểm kết thúc trùng nhau, đường đi này được gọi là chu trình. Một đường đi hoặc chu trình được gọi là đơn nếu không có cạnh nào bị lặp lại.

**Ví dụ:**

A black line with black dots and a triangle

Description automatically generated

* a, d, c, f, e là đường đi đơn độ dài 4.
* d, e, c, a không là đường đi vì (e,c) không phải là cạnh của đồ thị.
* Dãy b, c, f, e, b là chu trình độ dài 4.
* Đường đi a, b, e, d, a, b có độ dài 5 không phải là đường đi đơn vì cạnh (a,b) có mặt hai lần.

Đồ thị được gọi là liên thông nếu giữa mọi cặp đỉnh phân biệt đều tồn tại ít nhất một đường đi. Nếu không liên thông, đồ thị có thể chia thành các thành phần liên thông độc lập.

Đỉnh cắt và cạnh cắt là các yếu tố quan trọng trong việc xác định cấu trúc liên thông của đồ thị. Đỉnh cắt là đỉnh mà khi bị loại bỏ cùng với các cạnh liên thuộc sẽ làm tăng số thành phần liên thông. Tương tự, cạnh cắt là cạnh mà khi loại bỏ sẽ làm đồ thị bị chia thành nhiều thành phần hơn.

**Ví dụ:**

A diagram of a network

Description automatically generated

Đồ thị G là liên thông, nhưng đồ thị G’ không liên thông và có 3 thành phần liên thông.

**Ví dụ:**

A diagram of a network

Description automatically generated

Trong đồ thị trên, các đỉnh cắt là v, w, s và các cầu là (x,v), (w,s).

**Mệnh đề:**

* Giữa mọi cặp đỉnh phân biệt của một đồ thị liên thông luôn có đường đi sơ cấp.
* Mọi đơn đồ thị n đỉnh (n >= 2) có tổng bậc của hai đỉnh tuỳ ý không nhỏ hơn n đều là đồ thị liên thông.
* Đơn đồ thị mà bậc của mỗi đỉnh của nó không nhỏ hơn một nửa số đỉnh là đồ thị liên thông.
* Nếu một đồ thị có đúng hai đỉnh bậc lẻ thì hai đỉnh này phải liên thông, tức là có một đường đi nối chúng.

1. **Biểu diễn đồ thị**

Có rất nhiều cách khác nhau để biểu diễn một đồ thị, mỗi cách đều có những ưu điểm và ứng dụng riêng phù hợp với từng bài toán hoặc nhu cầu cụ thể. Một số phương pháp phổ biến để biểu diễn đồ thị bao gồm:

1. **Ma trận kề**: Đây là một bảng hai chiều, trong đó mỗi ô biểu diễn mối liên hệ giữa hai đỉnh của đồ thị. Nếu có cạnh nối hai đỉnh, ô tương ứng sẽ có giá trị khác 0 (thường là 1 trong đồ thị không trọng số).
2. **Ma trận trọng số**: Một dạng mở rộng của ma trận kề, trong đó các ô chứa trọng số của cạnh nối hai đỉnh. Phương pháp này thường được sử dụng cho các bài toán liên quan đến đồ thị có trọng số.
3. **Danh sách cạnh**: Liệt kê tất cả các cạnh trong đồ thị, mỗi cạnh được biểu diễn bằng cặp đỉnh mà nó kết nối. Cách này thích hợp để lưu trữ đồ thị thưa, khi số lượng cạnh ít hơn nhiều so với số lượng đỉnh.
4. **Danh sách kề**: Mỗi đỉnh trong đồ thị được liên kết với một danh sách các đỉnh mà nó kết nối. Phương pháp này thường tiết kiệm bộ nhớ hơn so với ma trận kề, đặc biệt khi đồ thị có ít cạnh.

Trong phạm vi nghiên cứu này, để đơn giản hóa và phù hợp với mục đích trình bày, ta chỉ tập trung vào việc biểu diễn đồ thị bằng **ma trận kề**, một phương pháp trực quan và dễ hiểu, thường được sử dụng rộng rãi trong lý thuyết đồ thị.

* 1. **Khái niệm**

Xét đồ thị G = (X, U) (Có hướng hay vô hướng)

Giả sử tập X gồm n đỉnh và được sắp xếp theo thứ tự X = {x1, x2, …, xn}, tập U gồm n cạnh và được sắp thứ tự U = {u1, u2, …, un}

* 1. **Quy tắc**

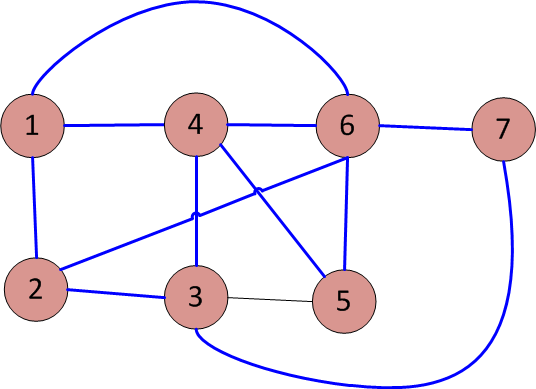
Ma trận kề của đồ thị G, ký hiệu B(G), là một ma trận nhị phân cấp n x n được định nghĩa như sau: B = (Bij) với:

* B = (Bij = 1) nếu có cạnh nối xi tới xj
* B = (Bij = 0) nếu không có cạnh nối xi tới xj

Nếu G là đồ thị vô hướng, ma trận liên thuộc của đồ thị G, ký hiệu A(G), là ma trận nhị phân cấp nxm được định nghĩa như sau: A = (Aij)

* A = (Aij) = 1 nếu có cạnh nối xi tới xj
* A = (Aij) = 0 nếu không có cạnh nối xi tới xj

**Ví dụ:**



Gọi A là ma trận kề biểu diễn đồ thị G.

Từ đồ thị G, ta thấy:

1 và 2 có cạnh nối => A12 = A21 = 1A 12 = A 21 = 1

1 và 4 có cạnh nối => A14 = A41 = 1A 14 = A 41 = 1 AA

1 và 6 có cạnh nối => A16 = A61 = 1A 16 = A 61 = 1 AASDA

2 và 3 có cạnh nối => A23 = A32 = 1A 23 = A 32 = 1 SAD

2 và 6 có cạnh nối => A26 = A62 = 1A 26 = A 62 = 1

3 và 4 có cạnh nối => A34 = A43 = 1A 34 = A 43 = 1 A

3 và 5 có cạnh nối => A35 = A53 = 1A 35 = A 53 = 1

3 và 7 có cạnh nối => A37 = A73 = 1A 37 = A 73 = 1

4 và 6 có cạnh nối => A46 = A64 = 1A 46 = A 64 = 1

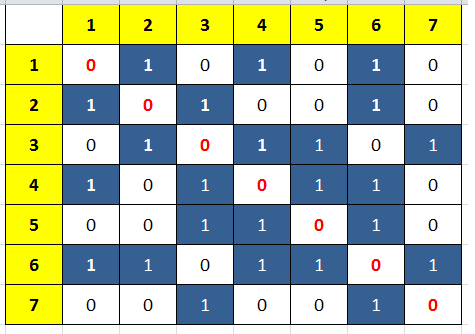
4 và 5 có cạnh nối => A45 = A54 = 1A 45 = A 54 = 1

5 và 6 có cạnh nối => A56 = A65 = 1A 56 = A 65 = 1 AA

6 và 7 có cạnh nối => A67 = A76 = 1A 67 = A 76 = 1

Còn lại các cặp đỉnh không có cạnh với nhau => Aij = Aji = 0

**Kết quả của ma trận kề:**



1. **Tìm kiếm trên đồ thị**

Trong phạm vi nghiên cứu này, chỉ có một thuật toán tìm kiếm được sử dụng, đó là thuật toán tìm kiếm theo chiều sâu (Depth First Search - DFS). Do đó, toàn bộ nội dung và phân tích trong bài sẽ tập trung vào việc làm rõ cơ chế hoạt động, các bước thực hiện, cũng như các ứng dụng cụ thể của thuật toán này để giải quyết bài toán được đặt ra

1. **Depth First Search – DFS là gì?**

* Tìm kiếm theo chiều sâu (Depth First Search - DFS) là một thuật toán quan trọng dùng để duyệt hoặc tìm kiếm trong các cấu trúc dữ liệu dạng đồ thị hoặc cây. Thuật toán khởi đầu từ một nút gốc (được chọn tùy ý trong trường hợp đồ thị) và tiếp tục duyệt xuống theo từng nhánh, đi càng sâu càng tốt trước khi quay lại để khám phá các nhánh chưa được duyệt.
* Kết quả của quá trình thực hiện DFS là một **cây bao trùm** (spanning tree), một dạng đồ thị đặc biệt không chứa chu trình, trong đó tất cả các đỉnh được kết nối với nhau thông qua các cạnh duy nhất. Cây bao trùm giúp xác định cấu trúc liên thông của đồ thị trong quá trình duyệt.
* Để triển khai DFS, cần sử dụng một cấu trúc dữ liệu ngăn xếp (stack). Ngăn xếp này sẽ lưu trữ các đỉnh tạm thời trong quá trình duyệt, với kích thước tối đa bằng tổng số đỉnh của đồ thị. Thuật toán hoạt động theo nguyên tắc LIFO (Last In, First Out), đảm bảo rằng các đỉnh mới được thêm vào ngăn xếp sẽ được xử lý trước khi quay lui để duyệt các đỉnh khác.

**Ví dụ:**

Hãy xem thuật toán Tìm kiếm theo chiều sâu hoạt động như thế nào với một ví dụ. Chúng ta dùng đồ thị vô hướng có 5 đỉnh.

A black background with white squares

Description automatically generated

Chúng ta bắt đầu từ đỉnh 0, thuật toán DFS bắt đầu bằng cách đứa nó vào danh sách Visited và đưa tất cả các cạnh liền kề đỉnh đang xét vào ngăn xếp.

A screenshot of a computer

Description automatically generated

Tiếp theo, chúng ta truy cập phần tử ở đầu ngăn xếp tức là 1 và đi đến các nút liền kề của nó. Vì 0 đã được truy cập, nên 2 là số được xét.

A screenshot of a video game

Description automatically generated

Đỉnh 2 có một đỉnh liền kề chưa được thăm là 4, vì vậy chúng ta thêm đỉnh đó vào vị trí đầu của ngăn xếp và duyệt nó.

A screenshot of a game

Description automatically generated

A screenshot of a video game

Description automatically generated

Sau khi chúng ta duyệt phần tử 3 cuối cùng, nó không có bất kỳ nút liền kề nào chưa được duyệt, vì vậy chúng tôi đã hoàn thành tìm kiếm theo chiều sâu trong đồ thị trên.

A screenshot of a video game

Description automatically generated

1. **Sử dụng DFS để kiểm tra tính liên thông của đồ thị**

**Ý tưởng:**

Thuật toán DFS duyệt qua tất cả các đỉnh và cạnh của đồ thị bằng cách bắt đầu từ một đỉnh ban đầu và tiếp tục đi sâu đến các đỉnh kề chưa được thăm. Nếu thuật toán có thể duyệt đến mọi đỉnh từ một đỉnh bất kỳ, đồ thị được coi là liên thông.

**Thực hiện:**

1. **Khởi tạo:**

* Tạo một tập hợp rỗng daTham để lưu các đỉnh đã được duyệt.
* Tạo một ngăn xếp stack để quản lý quá trình duyệt (tuân theo nguyên tắc LIFO).

1. **Xét duyệt**

* Vòng lặp với điều kiện ngăn xếp không rỗng thì sẽ:
  + Lấy đỉnh đầu tiên từ ngăn xếp.
  + Nếu đỉnh này chưa được thăm, thêm nó vào tập daTham.
  + Duyệt qua danh sách các cạnh của đồ thị, tìm tất cả các đỉnh kề với đỉnh vừa thêm mà chưa được thăm, và đẩy các đỉnh này vào ngăn xếp stack.

1. **Kết quả**

* Sau khi kết thúc quá trình duyệt, nếu số lượng đỉnh trong tập daTham bằng tổng số đỉnh của đồ thị, đồ thị là liên thông.
* Ngược lại, nếu có đỉnh nào không được thăm, đồ thị không liên thông.

**Thuật toán sẽ được thiết kế như sau:**

DFS-KiemTraLienThong(DoThi G):

Khởi tạo tập rỗng daTham

Khởi tạo ngăn xếp stack

Đẩy một đỉnh bất kỳ vào stack

Trong khi stack không rỗng:

dinh = stack.pop()

Nếu dinh chưa thuộc daTham:

Thêm dinh vào daTham

Với mỗi đỉnh kề dinhKe của dinhHienTai:

Nếu dinhKe chưa thuộc daTham:

Đẩy dinhKe vào stack

Nếu số lượng đỉnh trong daTham bằng tổng số đỉnh trong G:

Trả về “Liên thông”

Ngược lại:

Trả về “Không liên thông”

1. **Chu Trình Euler**
   1. **Tổng quan**

Những nguyên lý cơ bản của lý thuyết đồ thị chỉ mới được hình thành từthế kỷ XVIII, khởi nguồn từ công trình nghiên cứu của Leonhard Euler. Ông đã công bố một bài báo nổi tiếng giải quyết bài toán 7 cây cầu ở Königsberg, một vấn đề nổi bật trong lịch sử toán học.

Thành phố Königsberg, thuộc Đức (ngày nay là Kaliningrad, Nga), được chia thành bốn khu vực bởi các nhánh của sông Pregel, như hình minh họa. Năm 1736, thành phố có 7 cây cầu nối liền các khu vực này. Cư dân địa phương thắc mắc liệu có thể xuất phát từ một điểm trong thành phố, đi qua tất cả các cây cầu một lần duy nhất và quay lại điểm xuất phát hay không.

A diagram of a curved line

Description automatically generated with medium confidence

Euler đã thành công trong việc giải bài toán này bằng cách trừu tượng hóa bản đồ thành một đa đồ thị, trong đó các khu vực được biểu diễn bằng các đỉnh và các cây cầu là các cạnh nối. Bài toán thực tế này được tổng quát hóa thành một câu hỏi lý thuyết: Liệu có tồn tại một chu trình trong đa đồ thị đi qua tất cả các cạnh đúng một lần hay không?

Do **Leonhard Euler** là người đầu tiên đưa ra lời giải, loại chu trình đi qua tất cả các cạnh của đồ thị một lần duy nhất được đặt tên là chu trình Euler, như một sự tri ân dành cho nhà toán học vĩ đại trong lịch sử nhân loại.

* 1. **Khái niệm**

Chu trình đơn trong đồ thị G đi qua mỗi cạnh của đồ thị đúng một lần được gọi là chu trình Euler. Đường đi đơn trong G đi qua mỗi cạnh của nó đúng một lần được gọi là đường đi Euler. Đồ thị được gọi là đồ thị Euler nếu nó có chu trình Euler. Đồ thị có đường đi Euler được gọi là nửa Euler.

**Ví dụ**: Xét các đồ thị G1, G2, G3 sau:

A black and white image of a square with black dots and a square with black dots

Description automatically generated

Đồ thị G1 là đồ thị Euler vì nó có chu trình Euler a, e, c, d, e, b, a. Đồ thị G3 không có chu trình Euler nhưng chứa đường đi Euler a, c, d, e, b, d, a, b vì thế G3 là nửa Euler. G2 không có chu trình Euler cũng như đường đi Euler.

**Định lý:** Đồ thị (vô hướng) liên thông G là đồ thị Euler khi và chỉ khi mọi đỉnh của G đều có bậc chẵn.

**Bổ đề**: Nếu bậc của mỗi đỉnh của đồ thị G không nhỏ hơn 2 thì G chứa chu trình đơn.

**Hệ quả**: Đồ thị liên thông G là nửa Euler (mà không là Euler) khi và chỉ khi có đúng hai đỉnh bậc lẻ trong G.

* 1. **Giải thuật Fleury**

Thuật toán **Fleury** là một thuật toán mang tính lý thuyết nhưng lại khá đơn giản và được giới thiệu lần đầu tiên vào năm 1883. Thuật toán này áp dụng cho các đồ thị đã có sẵn thông tin về tính liên thông của tất cả các cạnh, đồng thời yêu cầu đồ thị phải có tối đa hai đỉnh bậc lẻ. Quá trình thực hiện thuật toán bắt đầu từ một đỉnh có bậc lẻ, hoặc nếu không có đỉnh bậc lẻ trong đồ thị, thuật toán sẽ lựa chọn ngẫu nhiên một đỉnh bất kỳ làm điểm xuất phát.

Thuật toán hoạt động theo cách lựa chọn cạnh tiếp theo trong quá trình di chuyển, với điều kiện rằng việc loại bỏ cạnh đó không làm đồ thị bị rời rạc, trừ trường hợp không còn cạnh nào thỏa mãn điều kiện này. Trong trường hợp duy nhất này, thuật toán sẽ chọn cạnh còn lại tại đỉnh hiện tại để tiếp tục. Sau khi chọn được cạnh, thuật toán sẽ di chuyển đến đỉnh còn lại của cạnh và loại bỏ cạnh đó khỏi đồ thị. Quá trình này sẽ tiếp tục cho đến khi không còn cạnh nào trong đồ thị.

Khi thuật toán hoàn thành, kết quả sẽ phụ thuộc vào số lượng đỉnh bậc lẻ trong đồ thị:

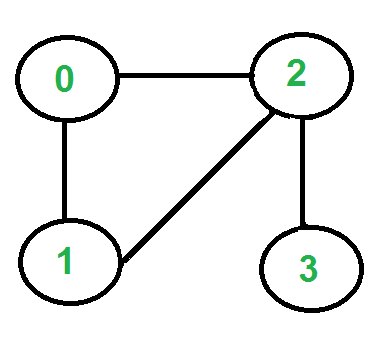
* Nếu đồ thị không có đỉnh bậc lẻ, dãy các cạnh được chọn sẽ tạo thành một **chu trình Euler** (tức là đồ thị có một chu trình đi qua tất cả các cạnh đúng một lần và quay lại điểm xuất phát).
* Nếu đồ thị có đúng hai đỉnh bậc lẻ, dãy các cạnh sẽ tạo thành một **đường đi Euler** (một đường đi đi qua tất cả các cạnh đúng một lần mà không quay lại điểm xuất phát).

Thuật toán Fleury đảm bảo rằng trong cả hai trường hợp trên, tất cả các cạnh của đồ thị sẽ được duyệt qua đúng một lần duy nhất, đồng thời quá trình lựa chọn các cạnh sẽ được thực hiện sao cho đồ thị không bị gián đoạn về tính liên thông, trừ khi không còn lựa chọn nào khác. Tuy nhiên, mặc dù thuật toán này mang lại kết quả chính xác, nhưng nó không được xem là tối ưu về mặt hiệu quả, đặc biệt khi số lượng cạnh trong đồ thị rất lớn.

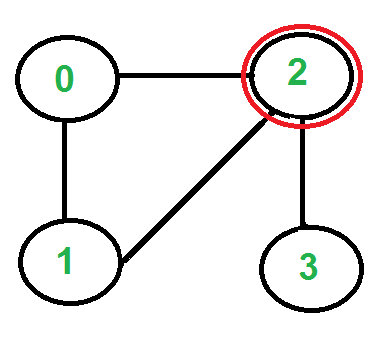
**Tóm lại:**

1. Đảm bảo đồ thị không có đỉnh bậc 0 hoặc bậc lẻ.
2. Nếu không có đỉnh bậc lẻ, thì bắt đầu duyệt ở bất kỳ đỉnh. Nếu có 2 đỉnh bậc lẻ, chọn một trong hai đỉnh để bắt đầu duyệt.
3. Duyệt qua từng cạnh một. Phải luôn luôn ưu tiên chọn cạnh không phải là cầu, nếu không còn lựa chọn khác thì chọn cạnh cầu
4. Dừng khi đã hết cạnh.

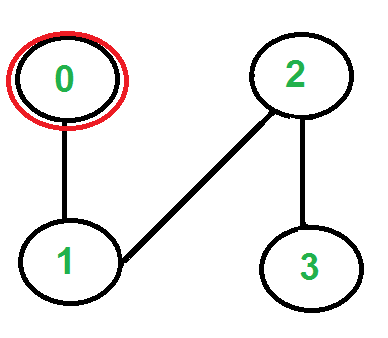
**Ví dụ:**



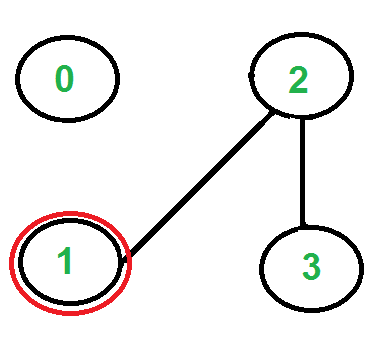
Ở đây có 2 đỉnh bậc lẻ là đỉnh “2” và đỉnh “3”, nên ta có thể bắt đầu ở một trong 2 đỉnh này. Ở ví dụ này sẽ chọn “2” để bắt đầu.



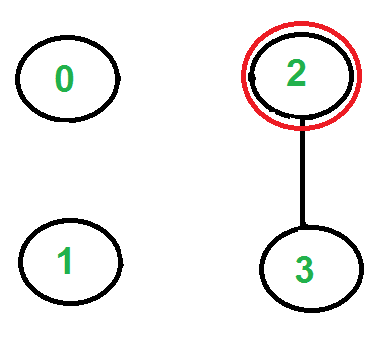
Từ đỉnh “2” ta có thể thấy có 3 cạnh đi ra từ nó. Đặc biệt là ở cạnh “2-3” chúng ta sẽ không chọn cạnh này vì cạnh này là cầu, một khi đi qua “3” thì sẽ không còn cạnh để quay lại. Nên ta có thể chọn 2 cạnh còn lại. Ở đây ta chọn “2-0”. Ta xoá cạnh này và di chuyển đến “0”. Chu trình hiện tại là: “2-0”.



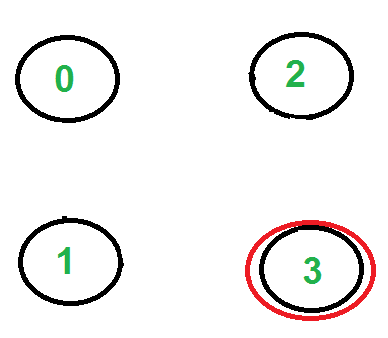
Ở đây chỉ còn 1 cạnh để chọn là “0-1” nên ta xoá cạnh này và di chuyển đến “1”. Chu trình hiện tại là: “0-2, 0-1”.



Ở đây cũng chỉ có 1 cạnh để chọn là “1-2” nên ta làm điều tương tự là xoá cạnh này và di chuyển đến “2”. Chu trình hiện tại là: “0-2, 0-1, 1-2”.



Tương tự như trên. Chọn cạnh “2-3”, xoá cạnh này, di chuyển đến “3”. Chu trình hiện tại là: “0-2, 0-1, 1-2, 2-3”.



Đến đây thì không còn cạnh nào để đi nữa nên thuật toán sẽ dừng!

Chu trình là: “0-2, 0-1, 1-2, 2-3”.

**Lưu đồ giải thuật**

**A diagram of a work flow

Description automatically generated**

1. **Môi trường sử dụng**

**Website tĩnh:**

* Là một loại website mà nội dung của nó được lưu trữ sẵn trên máy chủ dưới dạng các tệp được gửi trực tiếp đến trình duyệt của người dùng mà không cần xử lý hay thay đổi nội dung dựa trên hành vi người dùng. Các website này chủ yếu được phát triển bằng ngôn ngữ HTML để tạo cấu trúc cơ bản, cùng với CSS để định hình giao diện và phong cách trình bày. Ngoài ra, hình ảnh, âm thanh hoặc video thường được thêm vào để tăng tính hấp dẫn và làm phong phú nội dung.
* Một đặc điểm nổi bật của website tĩnh là nội dung hiển thị giống nhau cho tất cả khách truy cập, bất kể thời gian hay ngữ cảnh truy cập. Điều này tương tự như việc cung cấp một tài liệu in ấn cho nhiều khách hàng: nội dung không thay đổi trừ khi được cập nhật thủ công. Ví dụ điển hình của website tĩnh bao gồm các trang quảng bá đơn giản như website tài liệu quảng cáo, website năm trang hoặc website thông tin công ty, nơi thông tin được cố định và không yêu cầu sự tương tác phức tạp từ phía người dùng.

1. **Công cụ hỗ trợ**
   1. **Visual Studio Code (VS Code)**

Visual Studio Code (VS Code) là một trình soạn thảo mã nguồn được phát triển bởi Microsoft, hỗ trợ trên các hệ điều hành Windows, macOS, và Linux. Được phát hành lần đầu vào ngày 29 tháng 4 năm 2015 tại hội nghị Build 2015, Visual Studio Code nhanh chóng trở thành công cụ phổ biến cho lập trình viên nhờ giao diện thân thiện, khả năng tùy chỉnh mạnh mẽ, và hỗ trợ đa ngôn ngữ lập trình.

* **Đặc Điểm Chính**

1. **Hỗ trợ nhiều tính năng mạnh mẽ**:
   * **Debugging**: Tích hợp khả năng gỡ lỗi cho các ngôn ngữ như JavaScript, TypeScript, Python, C++, và nhiều ngôn ngữ khác.
   * **Syntax Highlighting**: Làm nổi bật cú pháp cho các ngôn ngữ như HTML, CSS, JavaScript, Python, Ruby, Rust, v.v.
   * **Tự động hoàn thành thông minh**: Sử dụng công nghệ IntelliSense để đưa ra các gợi ý mã phù hợp cho các ngôn ngữ phổ biến như TypeScript, CSS, HTML, và JSON.
   * **Snippets**: Cho phép sử dụng các đoạn mã mẫu để tăng tốc độ viết mã.
2. **Khả năng tùy chỉnh cao**:  
   VS Code cho phép người dùng tùy chỉnh giao diện và hành vi của phần mềm thông qua các file cấu hình JSON. Các tùy chỉnh này bao gồm việc thay đổi theme, gán phím tắt, và cài đặt tiện ích mở rộng phù hợp với nhu cầu lập trình.
3. **Hỗ trợ plugin và mở rộng**:  
   Kho tiện ích mở rộng của Visual Studio Code rất phong phú, cho phép bổ sung chức năng và hỗ trợ thêm các ngôn ngữ lập trình. Một điểm nổi bật là giao thức Language Server Protocol, giúp cải thiện phân tích mã nguồn thông qua các linter và công cụ tự động hóa.
4. **Quản lý phiên bản tích hợp**:  
   Một tính năng nổi bật khác là khả năng quản lý phiên bản (Version Control) được tích hợp sẵn, với hỗ trợ Git và các hệ thống khác như SVN và Perforce. Điều này cho phép lập trình viên dễ dàng theo dõi và quản lý các thay đổi trong dự án trực tiếp từ giao diện của VS Code.
   1. **GitHub**

GitHub là một dịch vụ cung cấp kho lưu trữ mã nguồn dựa trên hệ thống kiểm soát phiên bản Git, được phát triển với mục đích hỗ trợ cộng đồng lập trình viên trong việc quản lý, lưu trữ và chia sẻ mã nguồn. Ra mắt lần đầu vào tháng 4 năm 2008, GitHub nhanh chóng trở thành nền tảng phổ biến, cung cấp các giải pháp miễn phí cho các dự án mã nguồn mở và các tùy chọn trả phí cho doanh nghiệp hoặc cá nhân muốn bảo mật mã nguồn của mình.

**Tính năng nổi bật**

* Lưu trữ và quản lý mã nguồn: GitHub cho phép người dùng tạo và quản lý các kho lưu trữ (repository) sử dụng Git. Người dùng có thể tải mã lên, quản lý các thay đổi, và theo dõi lịch sử phát triển của dự án.
* Cộng đồng:
  + Thảo luận và cộng tác: Người dùng có thể thảo luận, xem xét mã nguồn, và đóng góp cho các dự án khác thông qua pull request.
  + Wiki và tài liệu: Hỗ trợ lưu trữ tài liệu và wiki trực tiếp trong các kho lưu trữ.
  + Theo dõi vấn đề: Giúp quản lý các yêu cầu tính năng hoặc lỗi trong dự án.
* Dịch vụ bổ sung:
  + Gist: Dịch vụ tương tự pastebin, dành cho việc chia sẻ các đoạn mã nhỏ hoặc ý tưởng.
  + GitHub Pages: Hỗ trợ tạo và lưu trữ các trang web tĩnh trực tiếp từ kho lưu trữ công khai.
* Tích hợp và mở rộng: GitHub hỗ trợ các công cụ kiểm tra tự động, linter, và hệ thống CI/CD thông qua các plugin. Ngoài ra, GitHub còn cung cấp API mạnh mẽ để các nhà phát triển tích hợp với các công cụ hoặc hệ thống khác.
  1. **diagrams.net**

**diagrams.net** (trước đây là **draw.io**) là một phần mềm vẽ sơ đồ đa nền tảng được phát triển bằng **HTML5** và **JavaScript**. Công cụ này hỗ trợ tạo ra nhiều loại sơ đồ như lưu đồ (flowcharts), khung giao diện (wireframes), sơ đồ UML, sơ đồ tổ chức, và sơ đồ mạng.

**Đặc điểm và Nền tảng**

* **Sẵn có trên nhiều nền tảng:**
  + **Web app**: diagrams.net hoạt động trực tuyến mà không yêu cầu đăng nhập hoặc đăng ký tài khoản, cho phép người dùng mở tệp từ ổ cứng và lưu tệp trực tiếp về thiết bị.
  + **Ứng dụng offline:** Cung cấp phiên bản dành cho các hệ điều hành **Linux**, **macOS**, và **Windows**, được xây dựng dựa trên nền tảng **Electron** để hỗ trợ làm việc ngoại tuyến.
* **Hỗ trợ lưu trữ và xuất tệp:**  
  diagrams.net hỗ trợ các định dạng lưu trữ và xuất tệp phổ biến như **PNG**, **JPEG**, **SVG**, và **PDF**, giúp người dùng dễ dàng chia sẻ và sử dụng tài liệu.
* **Tích hợp với các dịch vụ đám mây:**  
  Công cụ này tích hợp với các nền tảng lưu trữ đám mây hàng đầu như **Dropbox**, **OneDrive**, **Google Drive**, **GitHub**, và **GitLab**, cho phép người dùng lưu trữ và truy cập tệp từ bất kỳ thiết bị nào.
* **Hỗ trợ plugin nhúng:**  
  diagrams.net có thể được nhúng vào các nền tảng như **NextCloud**, **MediaWiki**, **Notion**, **Atlassian Confluence**, và **Jira**, nhờ các plugin chuyên biệt, giúp tối ưu hóa khả năng cộng tác và quản lý sơ đồ trong môi trường làm việc nhóm.

**Lợi thế so với các đối thủ**

Theo đánh giá từ các chuyên trang công nghệ như **TechRadar** và **PCMag**, diagrams.net được xem như một giải pháp thay thế hiệu quả và tiết kiệm chi phí so với các công cụ phổ biến như **Lucidchart**, **Microsoft Visio**, và **SmartDraw**. Điều này nhờ vào:

* Giao diện thân thiện, dễ sử dụng.
* Tính năng linh hoạt, hỗ trợ đa dạng loại sơ đồ.
* Không yêu cầu tài khoản để sử dụng, tăng tính riêng tư.

1. **Ngôn ngữ sử dụng**
2. **HTML**

**HTML** (viết tắt của **Hyper Text Markup Language**, hay **Ngôn ngữ Đánh dấu Siêu văn bản**) là một ngôn ngữ đánh dấu được thiết kế để tạo cấu trúc và trình bày nội dung cho các trang web trên World Wide Web. HTML là nền tảng cơ bản để xây dựng các tài liệu web và thường được kết hợp với các công nghệ khác như **CSS** (Cascading Style Sheets) để định dạng giao diện và **JavaScript** để thêm tính năng tương tác.

**Cấu Trúc và Chức Năng của HTML**

1. **Cấu trúc ngữ nghĩa:**  
   HTML mô tả cấu trúc của một trang web thông qua các **phần tử HTML**, được định nghĩa bằng các thẻ (tags) như:
   * <h1>: Tiêu đề chính.
   * <p>: Đoạn văn bản.
   * <a>: Liên kết.
   * <img>: Hình ảnh.

Các phần tử này giúp trình bày ngữ nghĩa của tài liệu, như tiêu đề, danh sách, trích dẫn, hoặc liên kết, và có thể bao gồm các phần tử khác như phần tử con.

1. **Tính năng đa phương tiện:**  
   HTML cho phép nhúng hình ảnh, video, âm thanh, và các đối tượng tương tác như biểu mẫu. Các trình duyệt web sử dụng mã HTML để hiển thị nội dung này thành các trang web đa phương tiện mà người dùng có thể tương tác.
2. **Tương tác với trình duyệt:**  
   Trình duyệt nhận tài liệu HTML từ máy chủ web hoặc từ kho lưu trữ cục bộ, sau đó render tài liệu thành giao diện trang web. Các thẻ HTML không hiển thị trực tiếp trên trang nhưng được trình duyệt sử dụng để xác định cấu trúc và nội dung.
3. **Kết hợp với CSS và JavaScript:**
   * **CSS** (Cascading Style Sheets): Quy định giao diện và bố cục của nội dung HTML. Kể từ năm 1997, **W3C** (World Wide Web Consortium) khuyến khích sử dụng CSS để thay thế HTML trong việc định dạng trực quan.
   * **JavaScript**: Được nhúng vào HTML để kiểm soát hành vi và thêm tính năng tương tác cho các trang web.
4. **CSS**

Trong lĩnh vực tin học, Cascading Style Sheets (CSS), hay còn gọi là ngôn ngữ định kiểu theo tầng, được sử dụng để mô tả cách trình bày của các tài liệu web được viết bằng HTML và XHTML. CSS là một công cụ mạnh mẽ cho phép lập trình viên tách biệt phần cấu trúc nội dung khỏi phần định dạng, mang lại sự linh hoạt và hiệu quả trong thiết kế web.

**Ứng dụng và phạm vi**

Ngoài việc được áp dụng phổ biến cho HTML và XHTML, CSS còn hỗ trợ các định dạng khác như XML, SVG (Scalable Vector Graphics), và XUL (XML User Interface Language). Điều này giúp CSS trở thành một chuẩn định dạng toàn diện cho các ngôn ngữ đánh dấu khác nhau.

**Ưu Điểm của CSS**

1. **Tính linh hoạt và tái sử dụng:**CSS cho phép bạn áp dụng một kiểu dáng duy nhất cho nhiều trang web, giúp giảm thiểu công việc lặp lại và dễ dàng thực hiện các thay đổi toàn cục khi cần.
2. **Cải thiện khả năng bảo trì:**Bằng cách tách biệt nội dung và định dạng, CSS giúp việc chỉnh sửa hoặc nâng cấp giao diện trang web trở nên đơn giản hơn mà không làm ảnh hưởng đến nội dung chính.
3. **Hỗ trợ đa nền tảng và thiết bị:**CSS cung cấp các thuộc tính giúp tối ưu hóa hiển thị trên nhiều loại thiết bị, từ màn hình máy tính, điện thoại di động, đến máy in.

**Kết Hợp với HTML và Các Công Nghệ Khác**

* **HTML và CSS:** HTML cung cấp cấu trúc cơ bản cho trang web, trong khi CSS định dạng giao diện, giúp trang web trở nên bắt mắt và dễ sử dụng hơn.
* **SVG và XUL:** CSS cung cấp khả năng định dạng các đối tượng đồ họa và giao diện người dùng dựa trên các ngôn ngữ đánh dấu này.

1. **JavaScript**

**JavaScript** là một ngôn ngữ lập trình dựa trên các ý niệm nguyên mẫu, được sử dụng rộng rãi cả trong phát triển web phía người dùng (client-side) lẫn phía máy chủ (server-side) thông qua Node.js. Ngôn ngữ này ban đầu được phát triển bởi **Brendan Eich** tại hãng Netscape Communications, với tên gọi đầu tiên là Mocha, sau đó đổi thành LiveScript, và cuối cùng được đặt tên là JavaScript nhằm tận dụng sự phổ biến của ngôn ngữ Java vào thời điểm đó.

Dù cú pháp của JavaScript tương tự với C, nó lại có nhiều điểm chung với Self hơn là Java. Các tệp mã nguồn JavaScript thường được lưu với phần mở rộng .js.

**Ứng Dụng Của JavaScript**

**1. Trên Trình Duyệt**

JavaScript thường được sử dụng để tạo các trang web động và tăng cường trải nghiệm người dùng. Một số ứng dụng điển hình bao gồm:

* **Tương tác với DOM (Document Object Model):** Tạo hiệu ứng hình ảnh, thay đổi nội dung trang web, kiểm tra dữ liệu nhập vào của người dùng, và cập nhật dữ liệu tự động mà không cần tải lại trang.
* **Công nghệ kết hợp:** JavaScript kết hợp với HTML và CSS để xây dựng các công nghệ nổi bật như:
  + **DHTML (Dynamic HTML):** Tăng cường khả năng tương tác động.
  + **Ajax:** Gửi và nhận dữ liệu từ máy chủ mà không cần tải lại trang.
  + **SPA (Single Page Applications):** Xây dựng ứng dụng web tải nhanh và mượt mà.

**2. Bên Ngoài Trình Duyệt**

JavaScript không chỉ giới hạn trong trình duyệt mà còn được sử dụng trong nhiều môi trường khác:

* **Node.js (2009):** Mang JavaScript từ trình duyệt đến môi trường máy chủ, cho phép xử lý backend nhanh chóng và hiệu quả.
* **Adobe Acrobat:** Hỗ trợ JavaScript để xử lý và tương tác trong tệp PDF.
* **Hệ điều hành:** Dashboard trên macOS 10.4 và Active Scripting của Microsoft đều sử dụng JavaScript để hỗ trợ các tính năng kịch bản động.

1. **Thư viện sử dụng**
   1. **jQuery**

**jQuery** là một thư viện JavaScript mã nguồn mở, được thiết kế để đơn giản hóa các thao tác với **HTML DOM** (Document Object Model), xử lý sự kiện, tạo hoạt ảnh CSS, và thực hiện các yêu cầu **Ajax**. Được phát hành theo giấy phép **MIT License**, jQuery miễn phí và có thể sử dụng tự do trong các dự án cá nhân lẫn thương mại.

* **Tính Phổ Biến và Ứng Dụng**
* Tính đến tháng 5 năm 2019, jQuery đã được triển khai trên **73% trong số 10 triệu trang web phổ biến nhất**, cho thấy sự thống trị của nó trong hệ sinh thái web.
* Phân tích từ các công cụ theo dõi web chỉ ra rằng jQuery là thư viện JavaScript được sử dụng rộng rãi nhất trên thế giới, với tỷ lệ sử dụng cao hơn ít nhất 3 đến 4 lần so với bất kỳ thư viện JavaScript nào khác.
* **Lợi Ích Chính của jQuery**

1. **Đơn giản hóa thao tác DOM:**  
   jQuery cung cấp một cú pháp ngắn gọn và thân thiện, giúp lập trình viên dễ dàng chọn, sửa đổi và thao tác các phần tử HTML.
2. **Xử lý sự kiện linh hoạt:**  
   Thư viện này giúp quản lý sự kiện trên trang web một cách hiệu quả và đồng nhất trên các trình duyệt khác nhau.
3. **Hỗ trợ Ajax:**  
   jQuery cho phép gửi và nhận dữ liệu từ máy chủ mà không cần tải lại trang, giúp xây dựng các ứng dụng web động và mượt mà hơn.
4. **Tạo hoạt ảnh CSS:**  
   Với jQuery, lập trình viên có thể dễ dàng thêm các hiệu ứng động vào trang web, từ các chuyển đổi cơ bản đến các hiệu ứng phức tạp hơn.
5. **Tương thích trình duyệt:**  
   jQuery giúp giải quyết vấn đề không tương thích giữa các trình duyệt, mang lại trải nghiệm thống nhất cho người dùng.
   1. **Bootstrap**

**Bootstrap** là một framework CSS miễn phí và mã nguồn mở chuyên dụng cho phát triển web front-end, giúp đơn giản hóa quá trình tạo và thiết kế trang web. Framework này cung cấp một tập hợp các mẫu thiết kế kiểu chữ, biểu mẫu, nút, thanh điều hướng, và các thành phần giao diện người dùng khác bằng ngôn ngữ **HTML**, **CSS**, và **JavaScript**.

* **Đặc Điểm Chính Của Bootstrap:**

1. **Tính linh hoạt và dễ sử dụng:**  
   Bootstrap cung cấp các định nghĩa kiểu mẫu có sẵn cho toàn bộ các thành phần HTML như văn bản, bảng biểu, biểu mẫu, nút nhấn, thanh điều hướng, v.v. Điều này giúp các nhà phát triển tiết kiệm thời gian và công sức trong việc thiết kế và định dạng trang web.
2. **Hỗ trợ responsive design:**  
   Bootstrap hỗ trợ **CSS Flexbox** và **Grid System**, cho phép tạo các bố cục linh hoạt và responsive. Người dùng có thể dễ dàng tạo các thiết kế phù hợp với nhiều kích thước màn hình từ **nhỏ (dưới 576px)** đến **lớn hơn 1200px**. Điều này bao gồm định nghĩa các vùng chứa với chiều rộng cố định hoặc chiều rộng linh hoạt, giúp nội dung tự động thay đổi kích thước theo từng thiết bị.
3. **Tích hợp JavaScript không phụ thuộc:**  
   Bootstrap đi kèm với một số thành phần JavaScript có sẵn, bao gồm hộp thoại, tooltip, modal, thanh tiến trình, trình đơn thả xuống điều hướng, và băng chuyền. Các thành phần này không yêu cầu thư viện như jQuery, mà chỉ cần gọi các plugin của Bootstrap trực tiếp từ các tệp JavaScript được thêm vào trang web.
4. **Tùy chỉnh và mở rộng dễ dàng:**  
   Bootstrap cung cấp phiên bản biên dịch sẵn với tệp **CSS** và **JavaScript** để dễ dàng thêm vào bất kỳ dự án nào. Tuy nhiên, framework này cũng hỗ trợ người dùng triển khai các tùy chỉnh sâu hơn thông qua **Sass** (công cụ tiền xử lý CSS) để thay đổi kiểu dáng, áp dụng chủ đề, và loại bỏ các thành phần không cần thiết.
5. **Giao diện người dùng thống nhất:**  
   Bootstrap giúp duy trì giao diện người dùng thống nhất trên mọi trình duyệt và thiết bị. Điều này đảm bảo rằng người dùng sẽ có trải nghiệm nhất quán, từ các thiết bị di động đến máy tính để bàn.

* **Các Thành Phần Nổi Bật Của Bootstrap**

1. **Container System:**  
   Bootstrap cung cấp 5 loại container với các kích thước cố định khác nhau tùy thuộc vào kích thước của màn hình. Từ **nhỏ hơn 576px**, **576–768px**, **768–992px**, **992–1200px**, đến **lớn hơn 1200px**.
2. **Grid System:**  
   Với hệ thống **Grid System** của Bootstrap, người dùng có thể tạo các cột và hàng linh hoạt bằng cách sử dụng các lớp CSS. Các lớp này giúp sắp xếp và căn chỉnh nội dung một cách dễ dàng, bao gồm **dòng lưới (row)** và **cột (col)**.
3. **Mẫu thiết kế (Templates):**  
   Bootstrap cung cấp các mẫu thiết kế sẵn như các thanh điều hướng, nút nhấn, mẫu form, hộp thoại, và các kiểu biểu mẫu tương tác. Người dùng có thể tùy chỉnh hoặc sử dụng trực tiếp các mẫu này vào dự án của mình.

* **Ứng Dụng Thực Tiễn**
* **Tạo trang web thông tin:** Các trang web đơn giản như **blog**, **portfolio cá nhân**, hoặc **trang quảng bá công ty** thường sử dụng Bootstrap để tạo giao diện đẹp mắt, dễ điều hướng.
* **Ứng dụng doanh nghiệp:** Bootstrap cũng được sử dụng để xây dựng các ứng dụng web doanh nghiệp phức tạp hơn, chẳng hạn như **quản lý nội dung (CMS)**, **quản lý tác vụ** (task management), và **quản lý quan hệ khách hàng (CRM)**.
* **Ứng dụng di động:** Bootstrap có khả năng tạo thiết kế **responsive**, hỗ trợ tối ưu hóa cho thiết bị di động, giúp các trang web hiển thị tốt trên thiết bị nhỏ như điện thoại thông minh, máy tính bảng.
  1. **p5**

**p5.js** là một thư viện JavaScript thân thiện, được thiết kế để giúp người dùng học lập trình và tạo nghệ thuật trực tuyến. Thư viện này miễn phí và mã nguồn mở, được phát triển bởi một cộng đồng toàn diện và hỗ trợ, với mục tiêu mang lại một công cụ dễ tiếp cận cho mọi đối tượng, từ nghệ sĩ, nhà thiết kế, người mới bắt đầu cho đến các giáo viên.

* **Mục Đích và Đặc Điểm**

1. **Giới thiệu về lập trình và nghệ thuật:**  
   p5.js được phát triển để giúp người học dễ dàng khám phá các khái niệm lập trình thông qua việc tạo ra các tác phẩm nghệ thuật trực quan, giúp kết nối lập trình và nghệ thuật một cách trực quan và dễ hiểu.
2. **Cộng đồng hỗ trợ và bao gồm:**  
   p5.js không chỉ là công cụ lập trình mà còn là một phần của một cộng đồng lớn, chào đón tất cả mọi người, không phân biệt nền tảng hay kinh nghiệm. Thư viện này thúc đẩy việc học tập và sáng tạo, mở rộng khả năng cho các nghệ sĩ, nhà thiết kế và lập trình viên.
3. **Tính dễ sử dụng:**  
   p5.js có cú pháp đơn giản và dễ làm quen, đặc biệt phù hợp cho người mới bắt đầu. Nó cung cấp một nền tảng tuyệt vời để các nhà sáng tạo khám phá và thử nghiệm các ý tưởng nghệ thuật bằng cách lập trình.

* **Ứng Dụng của p5.js**
* **Tạo nghệ thuật số:** p5.js cho phép người dùng tạo ra các tác phẩm nghệ thuật động, bao gồm vẽ đồ họa 2D, hiệu ứng hình ảnh và âm thanh tương tác.
* **Giáo dục lập trình:** Với mục tiêu là công cụ giáo dục, p5.js được sử dụng rộng rãi trong các lớp học lập trình, giúp học sinh và sinh viên hiểu được cách lập trình thông qua việc tạo ra những sản phẩm nghệ thuật thú vị.
* **Tương tác và hoạt hình:** p5.js cũng hỗ trợ việc tạo ra các tương tác và hoạt ảnh, cho phép người dùng tạo ra các trải nghiệm phong phú và thú vị trên web.
* **Miễn Phí**

p5.js là một dự án mã nguồn mở, với tất cả tài nguyên và công cụ đều có sẵn miễn phí. Điều này khuyến khích cộng đồng đóng góp và cải thiện thư viện, tạo ra một công cụ mạnh mẽ và linh hoạt hơn.

**Chương 3. Kết quả**

**1. Giới thiệu chương trình**

Đây là giao diện chính hay còn gọi là trang chủ.

A screenshot of a computer

Description automatically generated

Chương trình bao gồm ba chức năng chính, mỗi chức năng mang đến một cách thức vẽ khác nhau:

1. **Vẽ bằng dữ liệu tải lên của tệp** **(txt)**
2. **Vẽ thủ công**: nơi người dùng có thể tự tay thiết kế và tạo ra các hình vẽ theo
3. **Vẽ ngẫu nhiên**: cho phép chương trình tự động tạo ra các hình vẽ ngẫu nhiên mà không cần sự can thiệp của người dùng.
4. **Vẽ bằng dữ liệu tải lên của tệp (txt)**

**A screenshot of a computer

Description automatically generated**

Chức năng này cho phép người dùng tải lên một tệp văn bản (.txt) chứa dữ liệu đồ thị và vẽ đồ thị tương ứng. Cách sử dụng như sau:

**Tải lên tệp:**

* Người dùng bấm vào nút **Tải lên** nằm ở giữa màn hình để chọn tệp **.txt** từ thiết bị của mình.

A screenshot of a computer

Description automatically generated

* Định dạng của tệp phải tuân thủ quy tắc sau để đảm bảo dữ liệu được xử lý đúng:
  + **Dòng đầu tiên**: Chứa **số đỉnh** của đồ thị (ví dụ: 4, nghĩa là đồ thị có 4 đỉnh).
  + **Các dòng tiếp theo**: Chứa các giá trị nhị phân đại diện cho ma trận kề của đồ thị, mỗi dòng tương ứng với các cạnh kết nối của đỉnh với các đỉnh còn lại.

Ví dụ về tệp có định dạng hợp lệ:

4

1 1 1 0

1 0 1 1

1 1 1 0

0 1 0 0

**Chọn lại tệp:**

* Nếu muốn chọn lại tệp khác, người dùng có thể bấm vào nút **Tải lại** dưới màn hình. Nút này sẽ xóa tệp hiện tại và cho phép người dùng chọn lại một tệp mới.

A screenshot of a computer

Description automatically generated

**Về trang chủ:**

* Để trở về trang chủ, người dùng chỉ cần bấm vào nút **Trang chủ** ở góc trên bên trái của màn hình.

A yellow and white rectangle

Description automatically generated

**Kết quả cuối cùng:**

A screenshot of a cell phone

Description automatically generated

1. **Vẽ thủ công**

A screenshot of a computer

Description automatically generated

Đây là giao diện chính của chức năng này. Chức năng này cho phép người dùng tự tay vẽ một đồ thị bằng cách nhập dữ liệu kết nối giữa các đỉnh. Dưới đây là cách thực hiện từng bước:

**Nhập số đỉnh:**

* Người dùng bắt đầu bằng cách nhập vào số lượng đỉnh mong muốn trong ô nhập liệu tương ứng.
* Đây là bước cần thiết để xác định quy mô đồ thị.

A screenshot of a computer

Description automatically generated

**Tạo ô nhập liệu**

* Sau khi nhập số đỉnh, nhấn nút "Tạo".
* Chương trình sẽ hiển thị các ô nhập liệu tương ứng với số đỉnh đã nhập, cho phép người dùng thiết lập các kết nối giữa các đỉnh.

A screenshot of a computer

Description automatically generated

**Nhập đỉnh kết nối**

* Với các ô nhập liệu đã được tạo, người dùng điền thông tin về các đỉnh mà mình muốn kết nối.
* Ví dụ: Nếu muốn kết nối đỉnh A với đỉnh B, nhập các giá trị tương ứng vào ô nhập liệu.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Thêm đỉnh kết nối (Tuỳ chọn)**

* Nếu muốn thêm nhiều kết nối hơn, người dùng có thể nhấn nút "Thêm đỉnh" để tạo ra thêm các ô nhập liệu.
* Tính năng này hữu ích trong trường hợp cần mở rộng hoặc chỉnh sửa các kết nối đã nhập.

A screenshot of a quiz

Description automatically generated

A screenshot of a test

Description automatically generated

**Vẽ đồ thị**

* Khi đã hoàn tất việc nhập liệu và kết nối các đỉnh, người dùng nhấn nút "Vẽ".
* Hệ thống sẽ xử lý dữ liệu và hiển thị đồ thị tương ứng với các kết nối đã được chỉ định.

A screenshot of a quiz

Description automatically generated

**Kết quả cuối cùng:**

A screenshot of a cell phone

Description automatically generated

1. **Vẽ ngẫu nhiên**

**Giao diện chính**

**A screenshot of a computer

Description automatically generated**

Chức năng này được thiết kế để hỗ trợ người dùng tạo ra các đồ thị ngẫu nhiên một cách nhanh chóng và tiện lợi. Với thao tác đơn giản, người dùng chỉ cần nhập số đỉnh mong muốn và nhấn vào nút "Vẽ", hệ thống sẽ tự động vẽ ra một đồ thị ngẫu nhiên, bao gồm các đỉnh và cạnh được kết nối ngẫu nhiên theo quy luật được lập trình sẵn.

A screenshot of a computer

Description automatically generated

**Kết quả**  
A screenshot of a cell phone

Description automatically generated