ĐẠI HỌC
BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY
OF SCIENCE AND TECHNOLOGY

CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT

ĐỀ TÀI: Tìm kiếm đường đi trên bản đồ Bách Khoa

Họ tên sinh viên: Đặng Quang Vũ – 20223830

Mã lớp: 154832

Giảng viên: Trần Thị Thanh Hải

ONE LOVE. ONE FUTURE.

0. Nhiệm vụ

Nhiệm vụ	Mức độ hoàn thành
Trích xuất dữ liệu từ Google Maps	100%
Biểu diễn bản đồ tòa nhà dưới dạng đồ thị	90% (do số lượng tòa nhà và đường đi lớn)
Triển khai thuật toán Dijkstra	95% (thuật toán chạy tương đối tốt)
Lập trình	95% (thời gian biên dịch chưa nhanh – 1,3 giây)
Kiểm thử và đánh giá kết quả	95% (kết quả kiểm tra tương đối chính xác, tuy nhiên vẫn có một số trường hợp chưa chính xác)

1. Giới thiệu tổng quan về bài toán

▸ Yêu cầu bài toán

- Input: Các tòa nhà và khoảng cách giữa các tòa nhà.
- Output: Đường đi ngắn nhất giữa hai tòa nhà.

▸ Động lực thực hiện

Nhu cầu tìm đường trong khuôn viên Bách Khoa do Bách Khoa có diện tích lớn với nhiều tòa nhà.

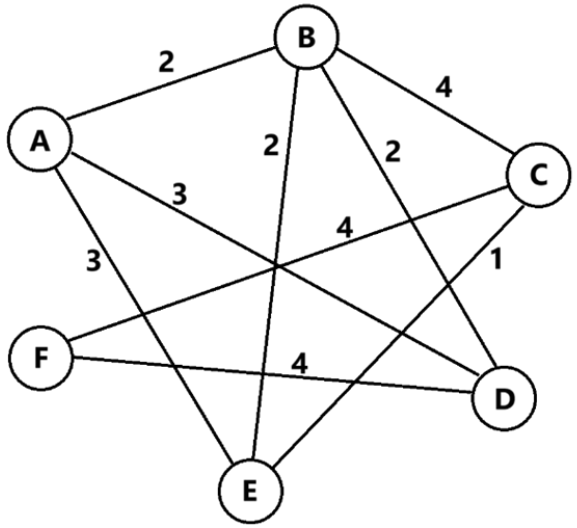
▸ Ứng dụng trong thực tế

- Tìm đường đi ngắn nhất trong mê cung.
- Phát triển hệ thống giao thông với nhiều nút giao phức tạp.

2. Phương pháp giải quyết

a. Dữ liệu

- Cấu trúc dữ liệu sử dụng trong bài toán là ***đồ thị***.
- Sử dụng đồ thị vô hướng có trọng số để biểu diễn khoảng cách giữa các tòa nhà.
- Lựa chọn đồ thị vô hướng vì việc di chuyển giữa hai tòa nhà là hai chiều (đi từ A → B và từ B → A cùng một đường).
- Lựa chọn đồ thị có trọng số vì khoảng cách giữa các tòa không giống nhau, trọng số sẽ biểu diễn khoảng cách giữa hai tòa nhà.



Ví dụ minh họa về đồ thị vô hướng có trọng số

2. Phương pháp giải quyết

a. Dữ liệu

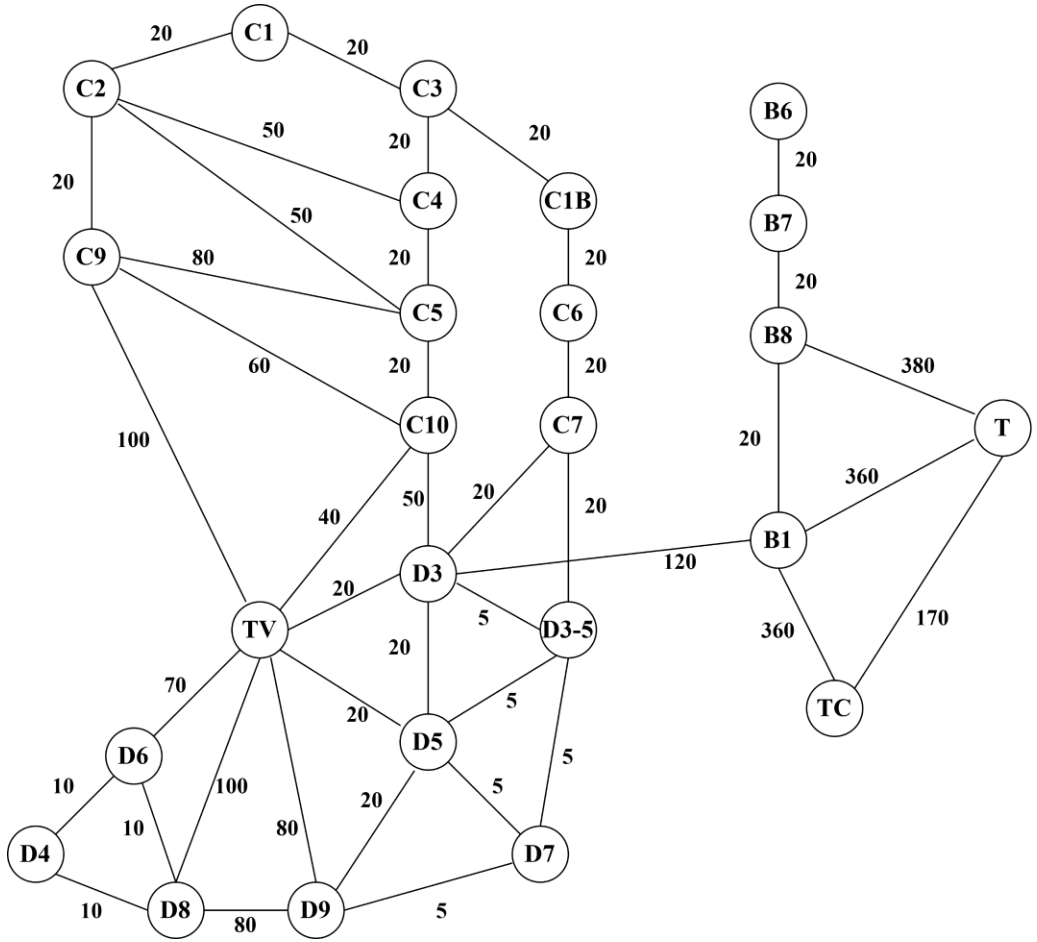
- Khai báo cấu trúc:
  + `to`: Tên tòa nhà, kiểu `string`.
  + `weight`: Khoảng cách giữa hai tòa nhà, tính theo mét, kiểu `int`.
- Do số lượng tòa nhà trong Bách Khoa tương đối lớn (khoảng 30 tòa nhà khác nhau) nên số lượng node cũng lớn. Dẫn theo đó số lượng đường đi cũng nhiều

Bản đồ các tòa nhà trong Bách Khoa

2. Phương pháp giải quyết

b. Giải thuật, phương thức giải quyết bài toán

- Giải thuật cho bài toán là giải thuật ***Dijkstra***.
- Nguyên lý của giải thuật:
  + Bắt đầu từ điểm xuất phát với khoảng cách là 0.
  + Tìm đỉnh có khoảng cách nhỏ nhất trong tập các đỉnh chưa được thăm.
  + Cập nhật khoảng cách ngắn nhất đến các đỉnh lân cận của nó.
  + Lặp lại cho đến khi đến đích hoặc tất cả các đỉnh được thăm.



Đồ thị chứa các tòa nhà được xây dựng dựa trên dữ liệu của Google Maps

2. Phương pháp giải quyết

b. Giải thuật, phương thức giải quyết bài toán

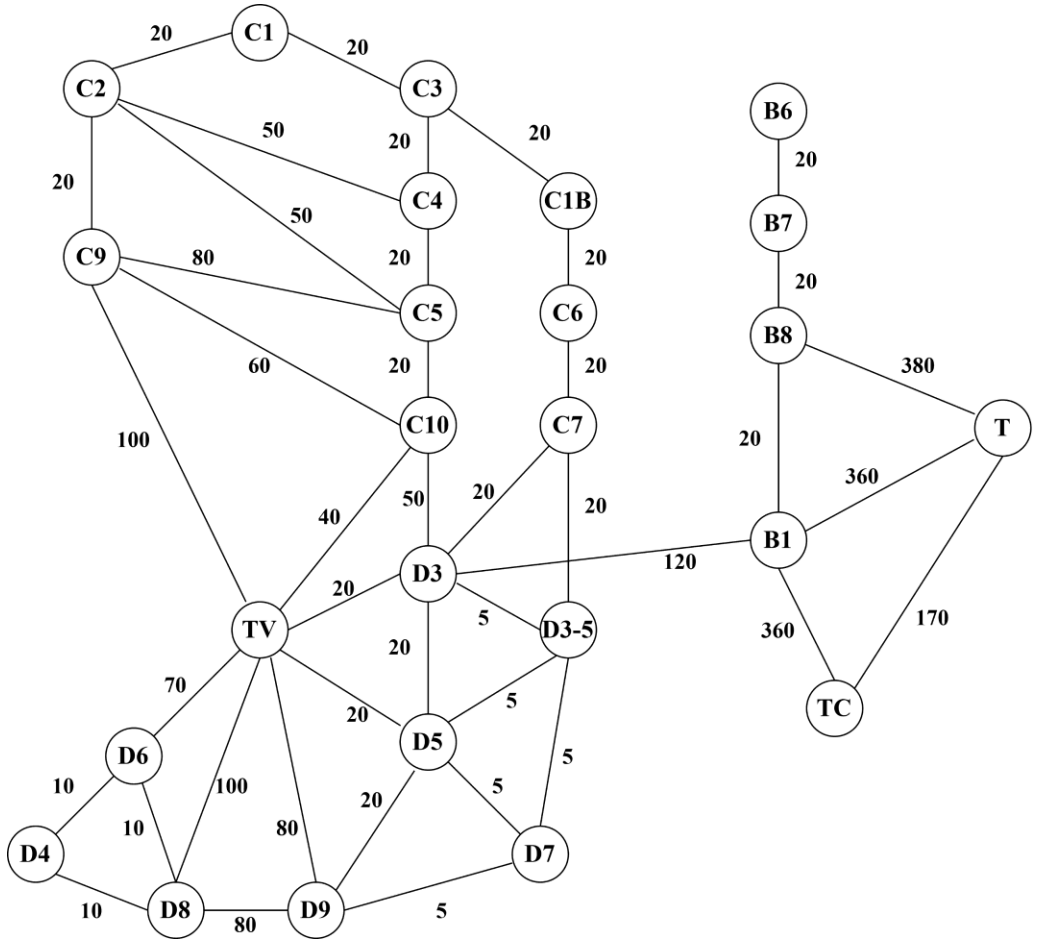
- Các bước thực hiện giải thuật:

  + Khởi tạo khoảng cách từ đỉnh đầu đến chính nó bằng 0, khoảng cách đến các đỉnh khác bằng ∞.

  + Lựa chọn đỉnh chưa thăm có khoảng cách nhỏ nhất (trọng số với đỉnh đầu nhỏ nhất).

  + Với mỗi đỉnh lân cận của đỉnh hiện tại, cập nhật nếu tìm được khoảng cách ngắn hơn.

  + Lặp lại cho đến khi đạt đích hoặc tất cả đỉnh được thăm.



Đồ thị chứa các tòa nhà được xây dựng dựa trên dữ liệu của Google Maps

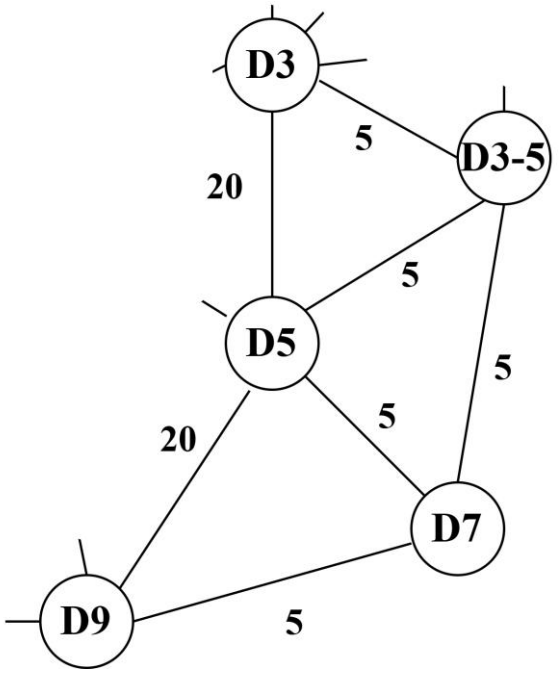
2. Phương pháp giải quyết

b. Giải thuật, phương thức giải quyết bài toán

- Minh họa cụ thể tìm đường đi ngắn nhất: Tìm đường đi ngắn nhất từ tòa D3 sang tòa D9.

  + Ta xây dựng được ma trận đỉnh kề như sau:

Đỉnh	D3	D3-5	D5	D7	D9
D3	0	5	20	∞	∞
D3-5	5	0	5	5	∞
D5	20	5	0	5	20
D7	∞	5	5	0	5
D9	∞	∞	20	5	0



Đồ thị minh họa

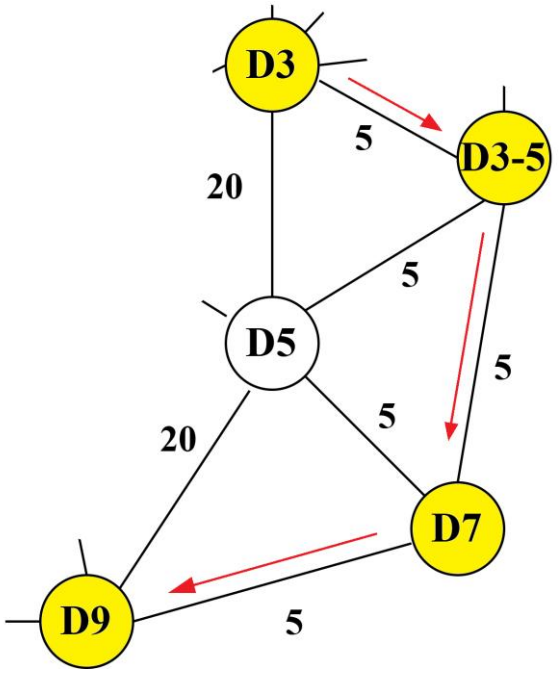
2. Phương pháp giải quyết

b. Giải thuật, phương thức giải quyết bài toán

- Minh họa cụ thể tìm đường đi ngắn nhất: Tìm đường đi ngắn nhất từ tòa D3 sang tòa D9.

  + Qua đó ta có bảng trạng thái khoảng cách:

Đỉnh đã xét	D3	D3-5	D5	D7	D9	Đỉnh tiếp theo
-	0	∞	∞	∞	∞	D3
D3	0	5	20	∞	∞	D3-5
D3; D3-5	0	5	10	10	∞	D5
D3; D3-5; D5	0	5	10	10	30	D7
D3; D3-5; D5; D7	0	5	10	10	15	D9
D3; D3-5; D5; D7; D9	0	5	10	10	15	-



Đồ thị minh họa

Từ đó khoảng cách nhỏ nhất khi đi từ tòa D3 sang D9 là 15 ***(D3 → D3-5 → D7 → D9)***.

2. Phương pháp giải quyết

c. Môi trường lập trình, các thư viện sử dụng

- Môi trường lập trình Dev-C++
- Các thư viện sử dụng

  + Thư viện `<iostream>`: Hỗ trợ nhập, xuất dữ liệu.

  + Thư viện `<vector>`: Cung cấp cấu trúc vector để lưu các cạnh của đồ thị.

  + Thư viện `<unordered_map>`: Dùng để lưu đồ thị dưới dạng danh sách kề.

  + Thư viện `<queue>`: Dùng để triển khai hàng đợi ưu tiên trong thuật toán Dijkstra.

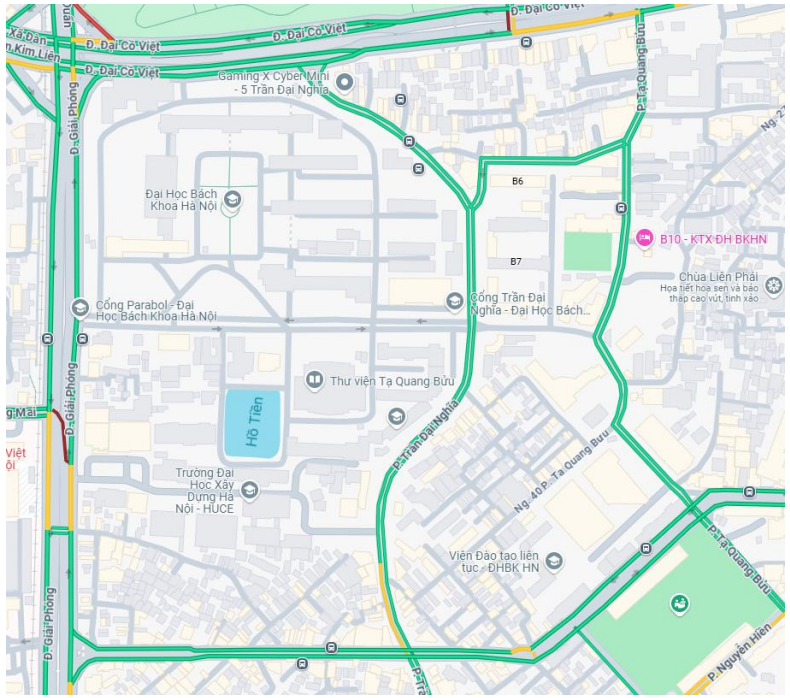
  + Thư viện `<climits>`: Chứa các giá trị cực đại (`INT_MAX`) để khởi tạo khoảng cách ban đầu trong Dijkstra.

  + Thư viện `<algorithm>`: Hỗ trợ hàm `reverse()` để đảo ngược đường đi tìm được.

3. Kết quả thực nghiệm

a. Xây dựng bộ test

Dựa trên bản đồ các tòa nhà và bản đồ trên Google Maps, ta có đồ thị chứa các tòa nhà với trọng số là khoảng cách gần đúng giữa các tòa, tính bằng mét.



Bản đồ Bách Khoa trên Google Maps

Bản đồ các tòa nhà trong Bách Khoa

3. Kết quả thực nghiệm

a. Xây dựng bộ test

No.	Input	Output
1	Nhap toa nha hien tai: D3 Nhap toa nha muon den: D9	Duong di ngan nhat: D3 -> D3-5 -> D7 -> D9 Do dai duong di ngan nhat khoang: 15 met.
2	Nhap toa nha hien tai: C2 Nhap toa nha muon den: T	Duong di ngan nhat: C2 -> C5 -> C10 -> D3 -> B1 -> T Do dai duong di ngan nhat khoang: 600 met.
3	Nhap toa nha hien tai: C7 Nhap toa nha muon den: D4	Duong di ngan nhat: C7 -> D3 -> TV -> D6 -> D4 Do dai duong di ngan nhat khoang: 120 met.
4	Nhap toa nha hien tai: TV Nhap toa nha muon den: TC	Duong di ngan nhat: TV -> D3 -> B1 -> TC Do dai duong di ngan nhat khoang: 500 met.
5	Nhap toa nha hien tai: D3 Nhap toa nha muon den: TC	Duong di ngan nhat: D3 -> B1 -> TC Do dai duong di ngan nhat khoang: 480 met.
6	Nhap toa nha hien tai: B7 Nhap toa nha muon den: TC	Duong di ngan nhat: B7 -> B8 -> B1 -> TC Do dai duong di ngan nhat khoang: 400 met.

3. Kết quả thực nghiệm

b. Kết quả bộ test

- ***Test case 1:*** Đường đi ngắn nhất từ D3 đến D9

```
Nhap toa nha hien tai: D3
Nhap toa nha muon den: D9
Duong di ngan nhat: D3 -> D3-5 -> D7 -> D9
Do dai duong di ngan nhat khoang: 15 met.
```

- ***Test case 2:*** Đường đi ngắn nhất từ C2 đến T

```
Nhap toa nha hien tai: C2
Nhap toa nha muon den: T
Duong di ngan nhat: C2 -> C5 -> C10 -> D3 -> B1 -> T
Do dai duong di ngan nhat khoang: 600 met.
```

- ***Test case 3:*** Đường đi ngắn nhất từ C7 đến D4

```
Nhap toa nha hien tai: C7
Nhap toa nha muon den: D4
Duong di ngan nhat: C7 -> D3 -> TV -> D6 -> D4
Do dai duong di ngan nhat khoang: 120 met.
```

3. Kết quả thực nghiệm

b. Kết quả bộ test

- ***Test case 4:*** Đường đi ngắn nhất từ TV đến TC

```
Nhap toa nha hien tai: TV
Nhap toa nha muon den: TC
Duong di ngan nhat: TV -> D3 -> B1 -> TC
Do dai duong di ngan nhat khoang: 500 met.
```

- ***Test case 5:*** Đường đi ngắn nhất từ D3 đến TC

```
Nhap toa nha hien tai: D3
Nhap toa nha muon den: TC
Duong di ngan nhat: D3 -> B1 -> TC
Do dai duong di ngan nhat khoang: 480 met.
```

- ***Test case 6:*** Đường đi ngắn nhất từ B7 đến TC

```
Nhap toa nha hien tai: B7
Nhap toa nha muon den: TC
Duong di ngan nhat: B7 -> B8 -> B1 -> TC
Do dai duong di ngan nhat khoang: 400 met.
```

4. Kết luận

a. Đánh giá về mức độ hoàn thành chung

- Đã xây dựng được hệ thống tìm đường cơ bản hoạt động chính xác, sử dụng thuật toán Dijkstra để xác định đường đi ngắn nhất giữa các tòa nhà.
- Kết quả trả về có độ chính xác tương đối tốt, đáp ứng hầu hết các yêu cầu tìm đường trong các test case.
- Với số lượng và kích thước các tòa nhà vừa và nhỏ, hệ thống đáp ứng tốt hơn với độ chính xác tương đối cao.
- Độ chính xác được duy trì kể cả khi thử nghiệm với trường hợp các tòa xa nhau, giữa các tòa có nhiều đường đi phức tạp.
- Với quy mô dữ liệu bản đồ vừa và nhỏ, hệ thống cho thời gian xử lý nhanh, với độ trễ thấp.

4. Kết luận

b. Các khó khăn / vướng mắc gặp phải khi triển khai

- Số lượng các tòa nhà và đường đi lớn, với trên dưới 30 tòa nhà lớn nhỏ khác nhau, dẫn đến lượng dữ liệu cho đồ thị cũng lớn theo. → Tối thiểu hóa số đường đi bằng cách loại bỏ các đường đi có độ dài tương tự nhau.
- Khoảng cách, độ dài đường đi giữa các tòa nhà sẽ không chính xác hoàn toàn, đặc biệt với các tòa nhà lớn như C1, C2, C7, D3… → Lấy xấp xỉ khoảng cách giữa các tòa nhà.

c. Hạn chế còn tồn tại và hướng phát triển

- *Hạn chế:* Hệ thống chỉ có đầu vào và đầu ra là dạng văn bản, ngươời dùng chưa hình dung trực quan đường đi và bản đồ như thế nào.
- *Hướng phát triển:* Tích hợp hoàn toàn dữ liệu chuẩn của Google Maps API để hiển thị bản đồ, tích hợp giải thuật vào các ứng dụng để người dùng có cái nhìn trực quan hơn…

HUST

THANK YOU !