

LECTURE 09

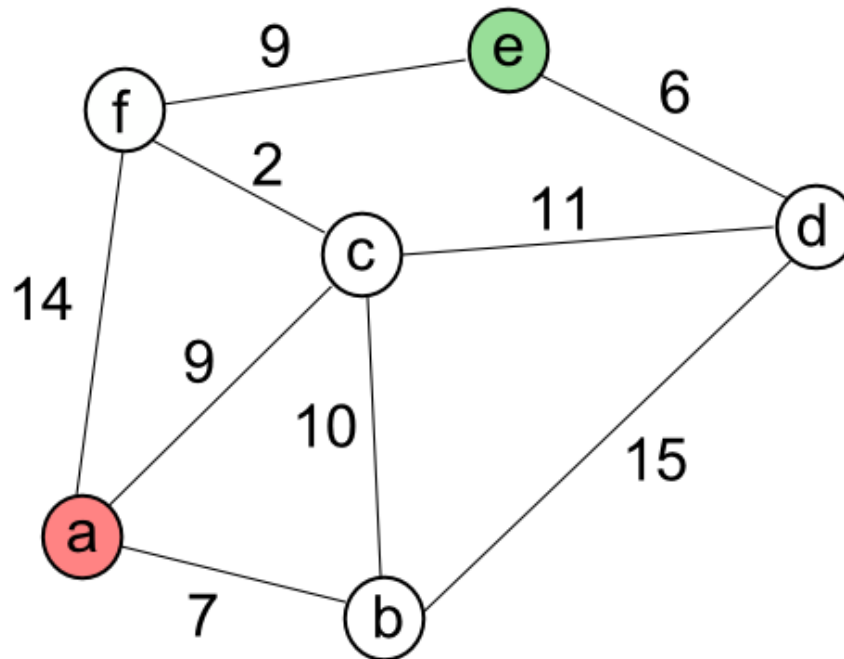
DIJKSTRA'S ALGORITHM

Phạm Nguyễn Sơn Tùng

Email: sontungtn@gmail.com

Định nghĩa thuật toán Dijkstra?

Thuật toán **Dijkstra** là thuật toán tìm đường đi có chi phí nhỏ nhất từ **một đỉnh** đến **tất cả các đỉnh** còn lại trong đồ thị có **trọng số** (trọng số **không âm**).



Độ phức tạp thuật toán Dijkstra?

Thuật toán **Dijkstra** bình thường sẽ có độ phức tạp là $O(V^2)$. Tuy nhiên ta có thể sử dụng kết hợp các cấu trúc dữ liệu khác để giảm độ phức tạp:

- Dùng Heap (priority queue) độ phức tạp sẽ là $O(E \log V)$.
- Dùng Set trong thư viện STL độ phức tạp là $O(E \log V)$.

Ý tưởng thuật toán

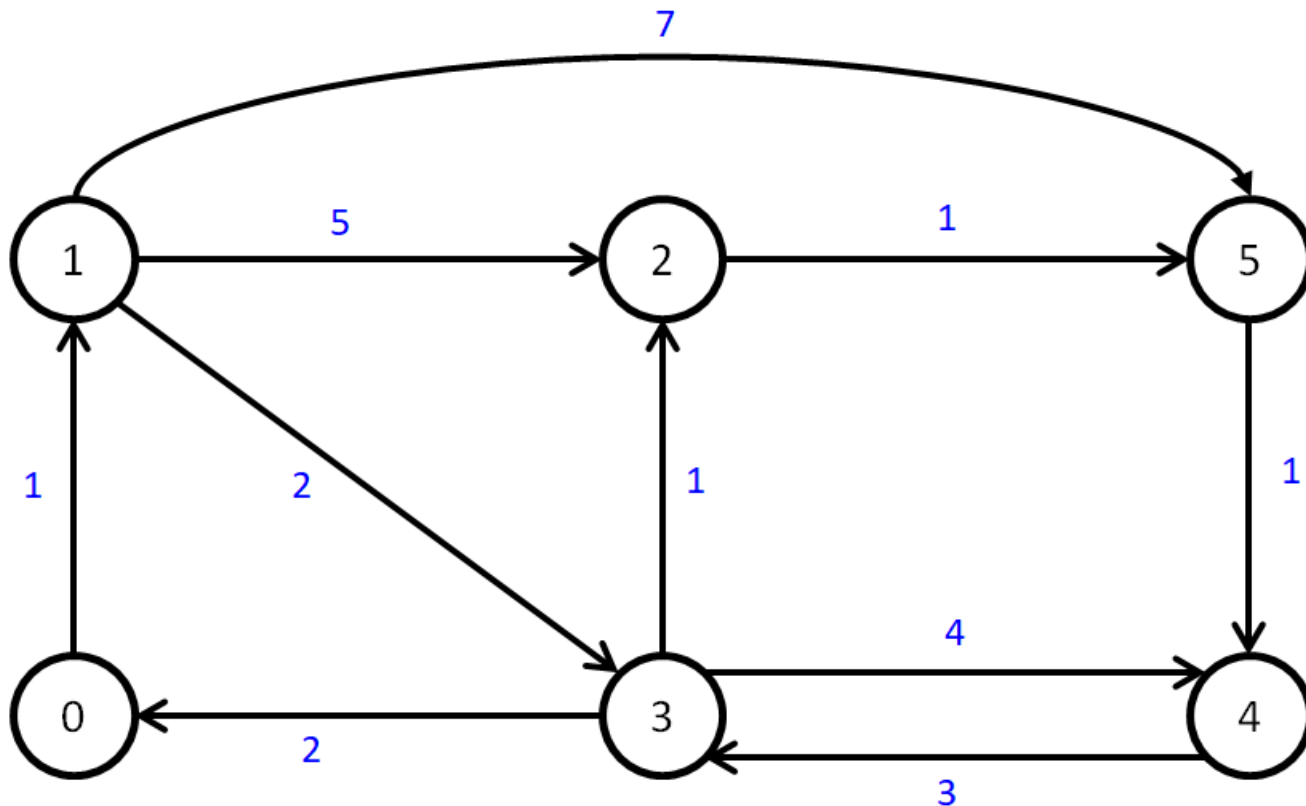
Xuất phát từ một đỉnh bất kỳ (đỉnh cần tìm đường đi ngắn nhất đến các đỉnh còn lại). Đi tới tất cả các đỉnh kề của đỉnh này, nếu chi phí đường đi đến đỉnh kề **nhỏ hơn** chi phí hiện tại của đỉnh đang xét thì ta cập nhật lại chi phí đường đi và lưu các đỉnh này lại.

Tiếp tục đem 1 đỉnh khác (từ tập đỉnh đã được lưu) ra xét và đi cho đến khi không còn đỉnh nào có thể đi. Mỗi bước đi nếu gặp chi phí mới nhỏ hơn chi phí hiện tại thì **cập nhật** lại.

Trong quá trình cập nhật chi phí, tiến hành **lưu đỉnh cha** của đỉnh kề. Kết quả lưu lại cuối cùng là đường đi với chi phí nhỏ nhất đi từ đỉnh xuất phát đến tất cả các đỉnh trong đồ thị.

Bài toán minh họa

Cho đồ thị có hướng như hình vẽ. Tìm **đường đi** từ đỉnh **0** đến tất cả các đỉnh còn lại.



Bước 0: Chuẩn bị dữ liệu

Từ dữ liệu đầu vào là ma trận kề, danh sách kề hoặc định dạng dữ liệu khác.

Adjacency Matrix

| | | | | | | |
|---|---|---|---|---|---|--|
| 6 | 0 | 4 | | | | |
| 0 | 1 | 0 | 0 | 0 | 0 | |
| 0 | 0 | 5 | 2 | 0 | 7 | |
| 0 | 0 | 0 | 0 | 0 | 1 | |
| 2 | 0 | 1 | 0 | 4 | 0 | |
| 0 | 0 | 0 | 3 | 0 | 0 | |
| 0 | 0 | 0 | 0 | 1 | 0 | |

Adjacency List

| | |
|---|-----|
| 6 | 10 |
| 0 | 1 1 |
| 1 | 2 5 |
| 1 | 3 2 |
| 1 | 5 7 |
| 2 | 5 1 |
| 3 | 0 2 |
| 3 | 2 1 |
| 3 | 4 4 |
| 4 | 3 3 |
| 5 | 4 1 |

Bước 0: Chuẩn bị dữ liệu

Chuyển ma trận đề vào **graph**.

| Đỉnh | 0 | 1 | 2 | 3 | 4 | 5 |
|------|--------|----------------------------|--------|----------------------------|--------|--------|
| Pair | (1, 1) | (2, 5) (3, 2) (5, 7) | (5, 1) | (0, 2) (2, 1) (4, 4) | (3, 3) | (4, 1) |

Mảng chứa chi phí đường đi **dist**.

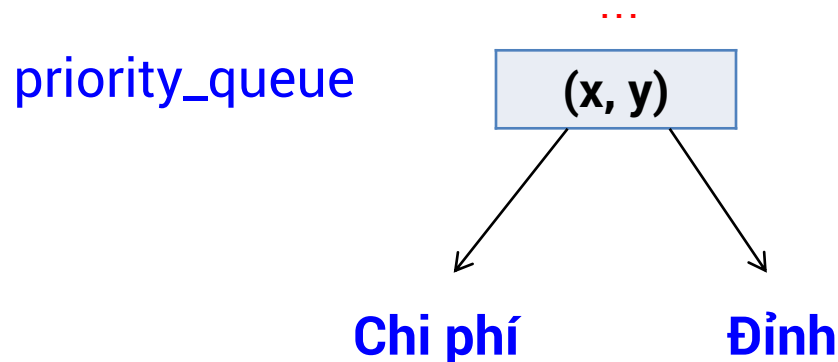
| Đỉnh | 0 | 1 | 2 | 3 | 4 | 5 |
|---------|----------|----------|----------|----------|----------|----------|
| Chi phí | ∞ | ∞ | ∞ | ∞ | ∞ | ∞ |

Bước 0: Chuẩn bị dữ liệu

Mảng lưu vết đường đi.

| | 0 | 1 | 2 | 3 | 4 | 5 |
|------|----|----|----|----|----|----|
| path | -1 | -1 | -1 | -1 | -1 | -1 |

Hàng đợi ưu tiên lưu cặp giá trị.



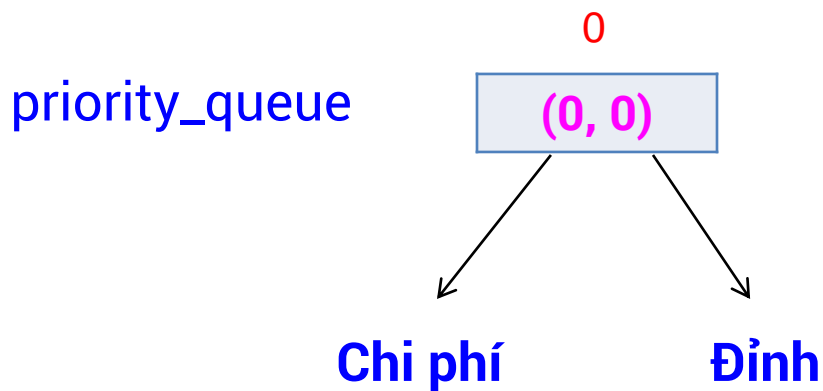
Bước 1: Chạy thuật toán lần 1

Lấy đỉnh bắt đầu đi là **đỉnh 0**. Gán chi phí cho đỉnh 0 là 0.

dist

| Đỉnh | 0 | 1 | 2 | 3 | 4 | 5 |
|---------|---|----------|----------|----------|----------|----------|
| Chi phí | 0 | ∞ | ∞ | ∞ | ∞ | ∞ |

➤ Bỏ đỉnh 0 vào hàng đợi theo quy tắc:



Bước 2: Chạy thuật toán lần 2

Lấy cặp **giá trị 0, đỉnh 0** ra khỏi hàng đợi và xét xem đỉnh 0 có kết nối với đỉnh nào trong đồ thị.

graph

| Đỉnh | 0 | 1 | 2 | 3 | 4 | 5 |
|------|--------|----------------------------|--------|----------------------------|--------|--------|
| Pair | (1, 1) | (2, 5) (3, 2) (5, 7) | (5, 1) | (0, 2) (2, 1) (4, 4) | (3, 3) | (4, 1) |

Chi phí trong bảng dist thay đổi (đang đứng ở đỉnh 0, chi phí 0).

- (1, 1): $0 + 1 < \infty \rightarrow$ Cập nhật đỉnh 1 của mảng dist.

dist

| Đỉnh | 0 | 1 | 2 | 3 | 4 | 5 |
|---------|---|------------------------|----------|----------|----------|----------|
| Chi phí | 0 | $\infty \rightarrow 1$ | ∞ | ∞ | ∞ | ∞ |

Bước 2: Chạy thuật toán lần 2

dist

| Đỉnh | 0 | 1 | 2 | 3 | 4 | 5 |
|---------|---|---|----------|----------|----------|----------|
| Chi phí | 0 | 1 | ∞ | ∞ | ∞ | ∞ |

Lưu cặp giá trị (1, 1) vào hàng đợi ưu tiên.

priority_queue

| |
|--------|
| 0 |
| (1, 1) |

Lưu giá trị đỉnh cha của đỉnh 1 lại.

path

| 0 | 1 | 2 | 3 | 4 | 5 |
|----|---|----|----|----|----|
| -1 | 0 | -1 | -1 | -1 | -1 |

Bước 3: Chạy thuật toán lần 3

Lấy cặp **giá trị 1, đỉnh 1** ra khỏi hàng đợi và xét xem đỉnh 1 có kết nối với đỉnh nào trong đồ thị.

graph

| Đỉnh | 0 | 1 | 2 | 3 | 4 | 5 |
|------|--------|----------------------------|--------|----------------------------|--------|--------|
| Pair | (1, 1) | (2, 5) (3, 2) (5, 7) | (5, 1) | (0, 2) (2, 1) (4, 4) | (3, 3) | (4, 1) |

Chi phí trong bảng dist thay đổi (đang đứng đỉnh 1, chi phí 1).

- (2, 5): $1 + 5 < \infty \rightarrow$ Cập nhật đỉnh 2 của mảng dist.
- (3, 2): $1 + 2 < \infty \rightarrow$ Cập nhật đỉnh 3 của mảng dist.
- (5, 7): $1 + 7 < \infty \rightarrow$ Cập nhật đỉnh 5 của mảng dist.

dist

| Đỉnh | 0 | 1 | 2 | 3 | 4 | 5 |
|---------|---|---|------------------------|------------------------|----------|------------------------|
| Chi phí | 0 | 1 | $\infty \rightarrow 6$ | $\infty \rightarrow 3$ | ∞ | $\infty \rightarrow 8$ |

Bước 3: Chạy thuật toán lần 3

dist

| Đỉnh | 0 | 1 | 2 | 3 | 4 | 5 |
|---------|---|---|---|---|----------|---|
| Chi phí | 0 | 1 | 6 | 3 | ∞ | 8 |

Lưu cặp giá trị (6, 2), (3, 3), (8, 5) vào hàng đợi ưu tiên, thứ tự trong hàng đợi ưu tiên thay đổi.

priority_queue

| 0 | 1 | 2 |
|--------|--------|--------|
| (3, 3) | (6, 2) | (8, 5) |

Lưu giá trị đỉnh cha của đỉnh 3 lại.

path

| 0 | 1 | 2 | 3 | 4 | 5 |
|----|---|---|---|----|---|
| -1 | 0 | 1 | 1 | -1 | 1 |

Bước 4: Chạy thuật toán lần 4

Lấy cặp **giá trị 3, đỉnh 3** ra khỏi hàng đợi và xét xem đỉnh 3 có kết nối với đỉnh nào trong đồ thị.

graph

| Đỉnh | 0 | 1 | 2 | 3 | 4 | 5 |
|------|--------|----------------------------|--------|----------------------------|--------|--------|
| Pair | (1, 1) | (2, 5) (3, 2) (5, 7) | (5, 1) | (0, 2) (2, 1) (4, 4) | (3, 3) | (4, 1) |

Chi phí trong bảng dist thay đổi (đang đứng đỉnh 3, chi phí 3).

- (0, 2): $3 + 2 > 0 \rightarrow$ **KHÔNG** cập nhật mảng dist.
- (2, 1): $3 + 1 < 6 \rightarrow$ Cập nhật đỉnh 2 của mảng dist.
- (4, 4): $3 + 4 < \infty \rightarrow$ Cập nhật đỉnh 4 của mảng dist.

dist

| Đỉnh | 0 | 1 | 2 | 3 | 4 | 5 |
|---------|---|---|-------------------|---|------------------------|---|
| Chi phí | 0 | 1 | 6 \rightarrow 4 | 3 | $\infty \rightarrow$ 7 | 8 |

Bước 4: Chạy thuật toán lần 4

dist

| Đỉnh | 0 | 1 | 2 | 3 | 4 | 5 |
|---------|---|---|---|---|---|---|
| Chi phí | 0 | 1 | 4 | 3 | 7 | 8 |

Lưu cặp giá trị (4, 2), (7, 4) vào hàng đợi ưu tiên, thứ tự trong hàng đợi ưu tiên thay đổi.

priority_queue

| 0 | 1 | 2 | 3 |
|--------|--------|--------|--------|
| (4, 2) | (6, 2) | (7, 4) | (8, 5) |

Lưu giá trị đỉnh cha của đỉnh 2, 4 lại.

path

| 0 | 1 | 2 | 3 | 4 | 5 |
|----|---|---|---|---|---|
| -1 | 0 | 3 | 1 | 3 | 1 |

Bước 5: Chạy thuật toán lần 5

Lấy cặp **giá trị 4, đỉnh 2** ra khỏi hàng đợi và xét xem đỉnh 2 có kết nối với đỉnh nào trong đồ thị.

graph

| Đỉnh | 0 | 1 | 2 | 3 | 4 | 5 |
|------|--------|----------------------------|--------|----------------------------|--------|--------|
| Pair | (1, 1) | (2, 5) (3, 2) (5, 7) | (5, 1) | (0, 2) (2, 1) (4, 4) | (3, 3) | (4, 1) |

Chi phí trong bảng dist thay đổi (đang đứng đỉnh 2, chi phí 4).

- (5, 1): $4 + 1 < 8 \rightarrow$ Cập nhật đỉnh 5 của mảng dist.

dist

| Đỉnh | 0 | 1 | 2 | 3 | 4 | 5 |
|---------|---|---|---|---|---|-------------------|
| Chi phí | 0 | 1 | 4 | 3 | 7 | 8 \rightarrow 5 |

Bước 5: Chạy thuật toán lần 5

| | | | | | | | |
|------|---------|---|---|---|---|---|---|
| dist | Đỉnh | 0 | 1 | 2 | 3 | 4 | 5 |
| | Chi phí | 0 | 1 | 4 | 3 | 7 | 5 |

Lưu cặp giá trị (5, 5) vào hàng đợi ưu tiên, thứ tự trong hàng đợi ưu tiên thay đổi.

| | | | | |
|----------------|--------|--------|--------|--------|
| | 0 | 1 | 2 | 3 |
| priority_queue | (5, 5) | (6, 2) | (7, 4) | (8, 5) |

Lưu giá trị đỉnh cha của đỉnh 5 lại.

| | | | | | | |
|------|----|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 |
| path | -1 | 0 | 3 | 1 | 3 | 2 |

Bước 6: Chạy thuật toán lần 6

Lấy cặp **giá trị 5, đỉnh 5** ra khỏi hàng đợi và xét xem đỉnh 5 có kết nối với đỉnh nào trong đồ thị.

graph

| Đỉnh | 0 | 1 | 2 | 3 | 4 | 5 |
|------|--------|----------------------------|--------|----------------------------|--------|--------|
| Pair | (1, 1) | (2, 5) (3, 2) (5, 7) | (5, 1) | (0, 2) (2, 1) (4, 4) | (3, 3) | (4, 1) |

Chi phí trong bảng dist thay đổi (đang đứng đỉnh 5, chi phí 5).

- (4, 1): $5 + 1 < 7 \rightarrow$ Cập nhật đỉnh 4 của mảng dist.

dist

| Đỉnh | 0 | 1 | 2 | 3 | 4 | 5 |
|---------|---|---|---|---|-------------------|---|
| Chi phí | 0 | 1 | 4 | 3 | 7 \rightarrow 6 | 5 |

Bước 6: Chạy thuật toán lần 6

dist

| Đỉnh | 0 | 1 | 2 | 3 | 4 | 5 |
|---------|---|---|---|---|---|---|
| Chi phí | 0 | 1 | 4 | 3 | 6 | 5 |

Lưu cặp giá trị (6, 4) vào hàng đợi ưu tiên, thứ tự trong hàng đợi ưu tiên thay đổi.

priority_queue

| 0 | 1 | 2 | 3 |
|--------|--------|--------|--------|
| (6, 2) | (6, 4) | (7, 4) | (8, 5) |

Lưu giá trị đỉnh cha của đỉnh 4 lại.

path

| 0 | 1 | 2 | 3 | 4 | 5 |
|----|---|---|---|---|---|
| -1 | 0 | 3 | 1 | 5 | 2 |

Bước 7: Chạy thuật toán lần 7

Lấy cặp **giá trị 6, đỉnh 2** ra khỏi hàng đợi và xét xem đỉnh 2 có kết nối với đỉnh nào trong đồ thị.

graph

| Đỉnh | 0 | 1 | 2 | 3 | 4 | 5 |
|------|--------|----------------------------|--------|----------------------------|--------|--------|
| Pair | (1, 1) | (2, 5) (3, 2) (5, 7) | (5, 1) | (0, 2) (2, 1) (4, 4) | (3, 3) | (4, 1) |

Chi phí trong bảng dist thay đổi (đang đứng đỉnh 2, chi phí 6).

- (5, 1): $6 + 1 > 5 \rightarrow$ **KHÔNG** cập nhật mảng dist.

dist

| Đỉnh | 0 | 1 | 2 | 3 | 4 | 5 |
|---------|---|---|---|---|---|---|
| Chi phí | 0 | 1 | 4 | 3 | 6 | 5 |

priority_queue

| | | |
|--------|--------|--------|
| 0 | 1 | 2 |
| (6, 4) | (7, 4) | (8, 5) |

Bước 8: Chạy thuật toán lần 8

Lấy cặp **giá trị 6, đỉnh 4** ra khỏi hàng đợi và xét xem đỉnh 4 có kết nối với đỉnh nào trong đồ thị.

graph

| Đỉnh | 0 | 1 | 2 | 3 | 4 | 5 |
|------|--------|----------------------------|--------|----------------------------|--------|--------|
| Pair | (1, 1) | (2, 5) (3, 2) (5, 7) | (5, 1) | (0, 2) (2, 1) (4, 4) | (3, 3) | (4, 1) |

Chi phí trong bảng dist thay đổi (đang đứng đỉnh 4, chi phí 6).

- (3, 3): $6 + 3 > 3 \rightarrow$ **KHÔNG** cập nhật mảng dist.

dist

| Đỉnh | 0 | 1 | 2 | 3 | 4 | 5 |
|---------|---|---|---|---|---|---|
| Chi phí | 0 | 1 | 4 | 3 | 6 | 5 |

priority_queue

| | |
|--------|--------|
| 0 | 1 |
| (7, 4) | (8, 5) |

Bước 9: Chạy thuật toán lần 9

Lấy cặp **giá trị 7, đỉnh 4** ra khỏi hàng đợi và xét xem đỉnh 4 có kết nối với đỉnh nào trong đồ thị.

graph

| Đỉnh | 0 | 1 | 2 | 3 | 4 | 5 |
|------|--------|----------------------------|--------|----------------------------|--------|--------|
| Pair | (1, 1) | (2, 5) (3, 2) (5, 7) | (5, 1) | (0, 2) (2, 1) (4, 4) | (3, 3) | (4, 1) |

Chi phí trong bảng dist thay đổi (đang đứng đỉnh 4, chi phí 7).

- (3, 3): $7 + 3 > 3 \rightarrow$ **KHÔNG** cập nhật mảng dist.

dist

| Đỉnh | 0 | 1 | 2 | 3 | 4 | 5 |
|---------|---|---|---|---|---|---|
| Chi phí | 0 | 1 | 4 | 3 | 6 | 5 |

0

priority_queue

(8, 5)

Bước 10: Chạy thuật toán lần 10

Lấy cặp **giá trị 8, đỉnh 5** ra khỏi hàng đợi và xét xem đỉnh 4 có kết nối với đỉnh nào trong đồ thị.

graph

| Đỉnh | 0 | 1 | 2 | 3 | 4 | 5 |
|------|--------|----------------------------|--------|----------------------------|--------|--------|
| Pair | (1, 1) | (2, 5) (3, 2) (5, 7) | (5, 1) | (0, 2) (2, 1) (4, 4) | (3, 3) | (4, 1) |

Chi phí trong bảng dist thay đổi (đang đứng đỉnh 5, chi phí 8).

- (4, 1): $8 + 1 > 6 \rightarrow$ **KHÔNG** cập nhật mảng dist.

dist

| Đỉnh | 0 | 1 | 2 | 3 | 4 | 5 |
|---------|---|---|---|---|---|---|
| Chi phí | 0 | 1 | 4 | 3 | 6 | 5 |

...

priority_queue





...

\rightarrow dừng thuật toán.

Kết quả chạy Dijkstra

Tìm đường đi ngắn nhất từ 0 đến 2.

path

| | | | | | | |
|--|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 |
| |  |  |  |  | | |
| | -1 | 0 | 3 | 1 | 5 | 2 |

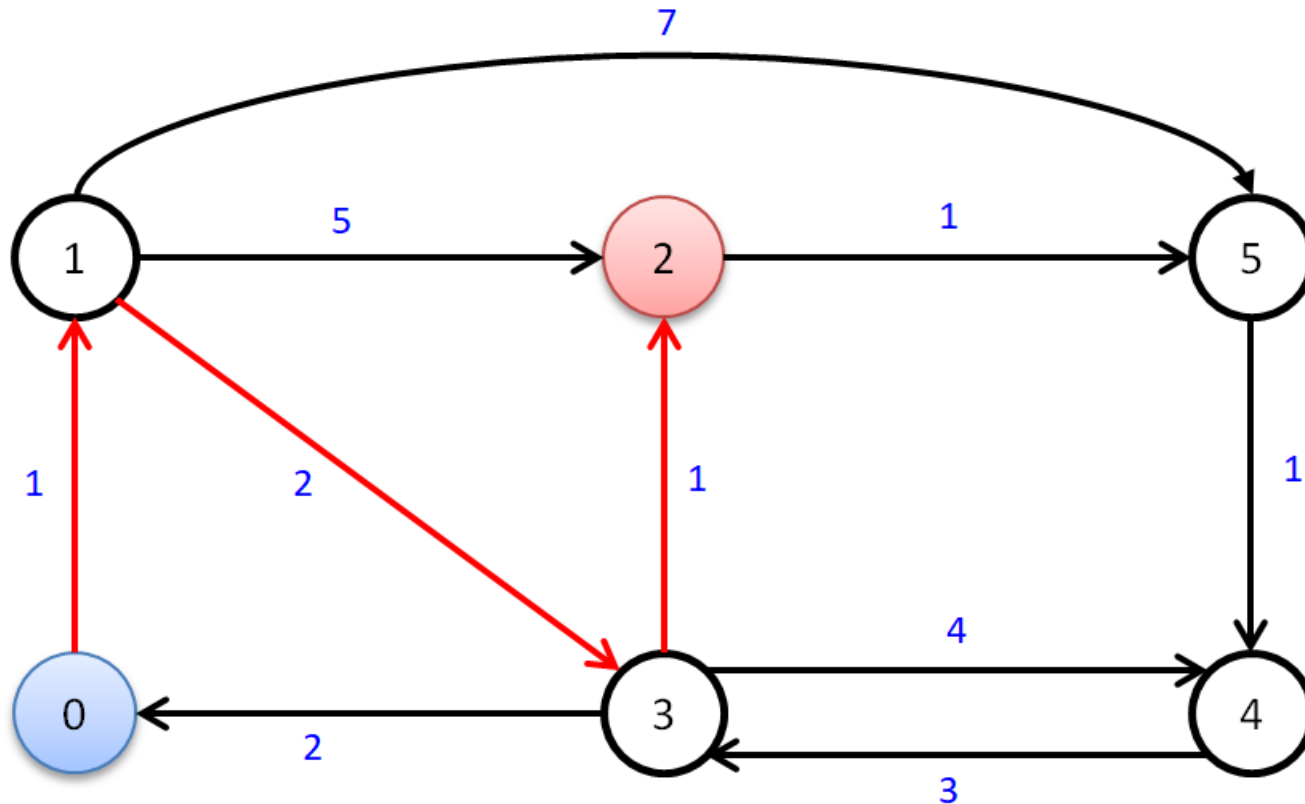
dist

| | | | | | | |
|---------|---|---|---|---|---|---|
| Đỉnh | 0 | 1 | 2 | 3 | 4 | 5 |
| Chi phí | 0 | 1 | 4 | 3 | 6 | 5 |



0 → 1 → 3 → 2
Chi phí: 4







Kết quả chạy Dijkstra



Kết quả chạy Dijkstra

Tìm đường đi ngắn nhất từ 0 đến 4.

path

| | | | | | | |
|--|---|---|---|---|---|---|
| |  |  |  |  |  |  |
| | 0 | 1 | 2 | 3 | 4 | 5 |
| | -1 | 0 | 3 | 1 | 5 | 2 |

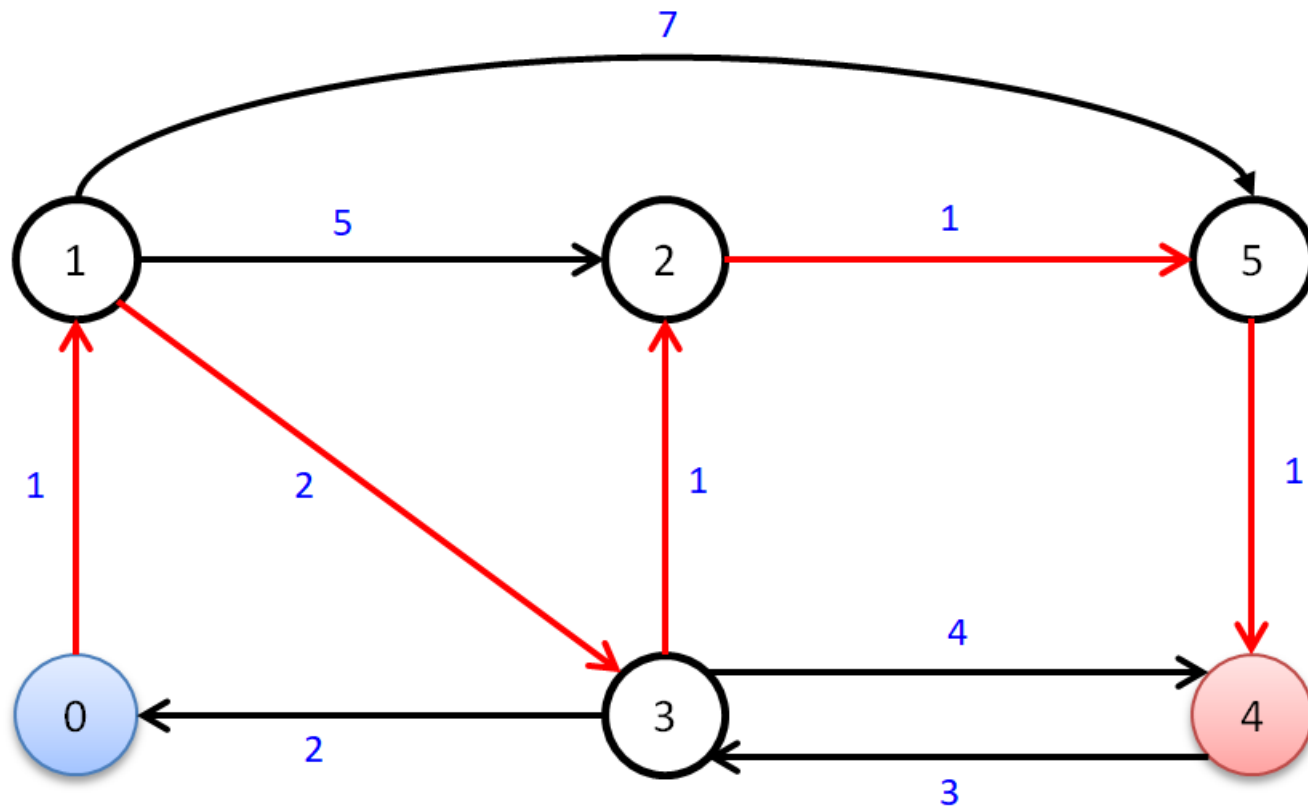
dist

| | | | | | | |
|---------|---|---|---|---|---|---|
| Đỉnh | 0 | 1 | 2 | 3 | 4 | 5 |
| Chi phí | 0 | 1 | 4 | 3 | 6 | 5 |



0 → 1 → 3 → 2 → 5 → 4
Chi phí: 6

Kết quả chạy Dijkstra



Source Code Dijkstra

Khai báo thư viện và các biến toàn cục:

```
#include <algorithm>
#include <iostream>
#include <string>
#include <vector>
#include <queue>
#include <functional>
using namespace std;
#define MAX 100
const int INF = 1e9;
vector<vector<pair<int, int> > > graph;
vector<int> dist(MAX, INF);
int path[MAX];
```

Source Code Dijkstra

Thuật toán Dijkstra (part 1)

```
void Dijkstra(int source, vector<vector<pair<int, int> > > &v, vector<int>
&distance)
{
    //Tạo một hàng đợi ưu tiên phần tử có độ ưu tiên cao nhất là phần tử có
    giá trị nhỏ nhất (greater)
    priority_queue<pair<int, int>, vector<pair<int, int> >,
                    greater<pair<int, int> > > pq;

    //first: chỉ phí, source đỉnh.
    pq.push(pair<int, int>(0, source));

    //Gán đỉnh bắt đầu chỉ phí là 0
    distance[source] = 0;
```

Source Code Dijkstra

Thuật toán Dijkstra (part 2)

```
while (!pq.empty())
{
    pair<int, int> top = pq.top();
    pq.pop();
    int node = top.second;
    int d = top.first;
    for (int i = 0; i < v[node].size(); ++i)
    {
        pair<int, int> neighbor = v[node][i];
        if (d + neighbor.second < distance[neighbor.first])
        {
            distance[neighbor.first] = d + neighbor.second;
            pq.push(pair<int, int>(distance[neighbor.first], neighbor.first));
            path[neighbor.first] = node;
        }
    }
}
```

Source Code Dijkstra

Hàm main để chạy chương trình

```
int main()
{
    freopen("INPUT.INP", "rt", stdin);
    int n, s, t;
    dist = vector<int>(MAX, INF);
    cin >> n >> s >> t;
    graph = vector<vector<pair<int, int> > >(MAX + 5, vector<pair<int, int> >());
    int d = 0;
    for (int i = 0; i<n; i++)
        for (int j = 0; j<n; j++)
        {
            cin >> d;
            if(d>0)
                graph[i].push_back(pair<int, int>(j, d));
        }
    Dijkstra(s, graph, dist);
    int ans = dist[t];
    cout << ans;
    return 0;
}
```

Bonus: Cấu trúc pair (1)

pair là một cấu trúc gồm 2 thuộc tính, 2 thuộc tính này có thể cùng hoặc khác kiểu dữ liệu với nhau.



first
second

Bonus: Cấu trúc pair (2)

Thư viện:

```
<Thư viện bất kỳ nào đó>  
using namespace std;
```

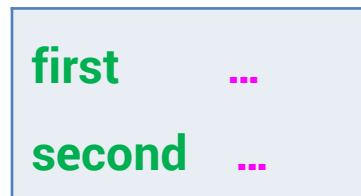
Khai báo dạng mặc định:

```
pair<data_type1, data_type2> variable_name;
```

Ví dụ:

```
pair<int, string> p;
```

p



Bonus: Cấu trúc pair (3)

Khai báo dạng có tham số đầu vào:

```
pair<data_type1, data_type2> variable_name(value1, value2);
```

Ví dụ:

```
pair<int, string> p(10, "abc");
```

p

| | |
|--------|-------|
| first | 10 |
| second | "abc" |

Bonus: Cấu trúc pair (4)

first, second: Truy cập và gán giá trị cho pair.

```
pair<int, string> p;  
p.first = 2;  
p.second = "abc";  
cout<<p.first<<"- "<<p.second;
```



Kết quả

2-abc

Bonus: Cấu trúc pair (5)

Kết hợp với vector: Tạo thành mảng cặp giá trị.

```
vector<pair<int, string> > v;  
pair<int, string> p;  
p = make_pair(2, "abc");  
v.push_back(p);  
p = make_pair(3, "def");  
v.push_back(p);
```



| 0 | 1 |
|----------|----------|
| 2, "abc" | 3, "def" |

Bonus: Cấu trúc pair (6)

Kết hợp với mảng tĩnh: Tạo thành mảng cặp giá trị.

```
typedef pair<int, int> Student;
```

```
int main()  
{  
    Student a[100];  
    return 0;  
}
```

Hỏi đáp

