## **Ví dụ**

Input:

7 ← số con đường

0 1 4

0 3 8

1 4 1

1 2 2

4 2 3

2 5 3

3 4 2

0 ← Vị trí bắt đầu

4 ← Số query

1

4

5

7

Đường đi ngắn nhất từ:

* 0 tới 1: 0 - 1 → Chi phí: **4**
* 0 tới 4: 0 - 1 - 4 → Chi phí: **5**
* 0 tới 5: 0 - 1 - 2 - 5 → Chi phí: **9**
* 0 tới 7: không có → **NO PATH**

## **Nhận xét:**

* Location được đánh số từ 0 .. 500 ⇒ Có 501 location.
* Mỗi location xem như là 1 node, mỗi road xem là 1 cạnh ⇒ Tạo thành đồ thị vô hướng.
* Tìm đường đi từ đỉnh S đến Q các đỉnh khác. Mà đồ thị có trọng số không âm (**1≤W≤100)** ⇒ Dùng thuật Dijkstra.

## **Giải thuật:**

B1: Đọc dữ liệu.

B2: Khởi tạo mảng dist gồm 501 phần tử với giá trị INF.

B3: Chạy thuật Dijkstra từ đỉnh S.

B4: Với mỗi đỉnh V (đỉnh kết thúc):

* Nếu dist[V] = INF thì xuất NO PATH
* Ngược lại, xuất dist[V].

## **Mã giả:**

class Node:

id, w

INF = 1e9 + 5

dijkstra(s):

pq = MinHeap() // lưu trữ dựa w của Node

pq.push(Node(s, 0))

dist[s] = 0

while pq not empty:

u, w = pq.pop()

for neighbor in graph[u]:

if w + neighbor.w < dist[v]:

dist[v] = w + neighbor.w

pq.push(Node(v, dist[v]))

main():

read(n) // số cạnh

graph = []

for i = 0 to n - 1:

read(u, v, w)

graph[u].append(Node(v, w))

graph[v].append(Node(u, w))

read(S)

dist = [] // có 501 phần tử INF

dijkstra(s) //O(ElogV)

read(Q)

for i = 0 to Q -1:

read(V)

if dist[V] = INF:

print(“NO PATH”)

else:

print(dist[V])

Độ phức tạp: O(ElogV)