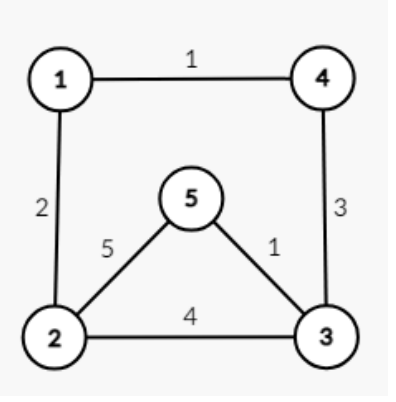
Dijkstra (ডায়াক্সট্রা)

input (use CSA academy graph editor)

```
5 6
1 2 2
2 5 5
2 3 4
1 4 1
4 3 3
3 5 1
```



Graph representation using adj list:

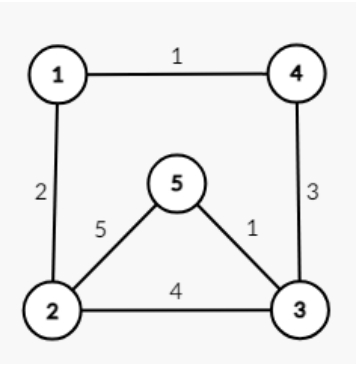
1 ⟶ (1,4), (2,2) // weight, node pair

2 ⟶ (2,1), (5,5), (4,3)

3 ⟶ (4,2), (1,5), (3,4)

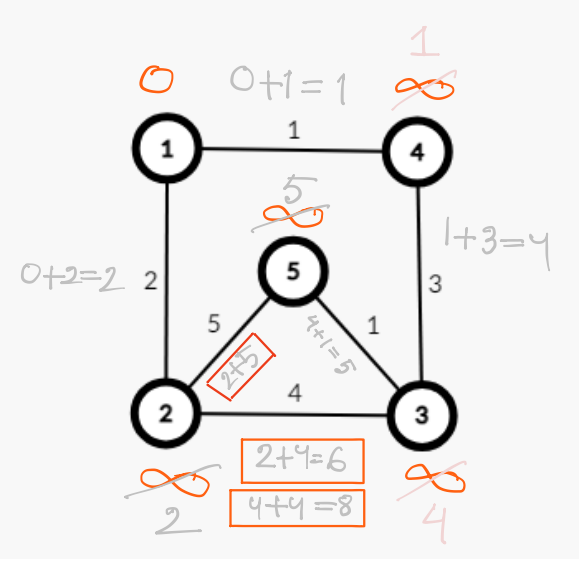
4 ⟶ (1,1), (3,3)

5 ⟶ (5,2), (1,3)




```
if(dis[u] + w < dis[v]) // relax the path
```

very very easy

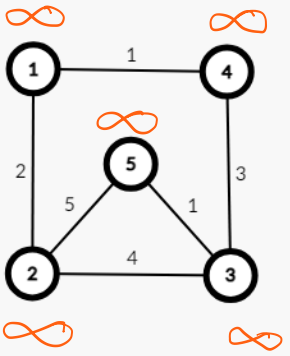


* লাল বক্স দ্বারা dis[u] + w > dis[v] বুঝানো হয়েছে। অর্থাৎ এরচে shortest path available আছে।

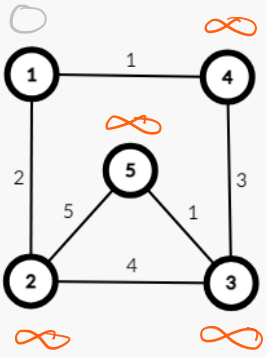
Select Node	1	2	3	4	5
	∞	∞	∞	∞	∞
নিজ হাতে সেটা zero		∞	∞	∞	∞
1	Explored 0	2	∞	1	∞
4	0	2	4	1 Explored	∞
2	0	2 Explored	4	1	7
3	0	2	4 Explored	1	5
5	0	2	4	1	5 Explored

Let's implement the code

```cpp
for (int i = 1; i <= n; i++)
    {
        dis[i] = INF;
    }
```




```cpp
dis[src] = 0;
```



Now,

(i) select করো

(ii) Pop করে দাও

(iii) যদি Relax করতে পারে, Queue তে পুশ করো।

Repeat until !pq.empty()

```cpp
priority_queue<pi, vector<pi>, greater<pi>> pq;
    pq.push({dis[src], src});
```

নিজ হাতে source কে push করা

```cpp
while (!pq.empty())
    {
        pi parent = pq.top();
        pq.pop();
        int parentNode = parent.second;
        int parentCost = parent.first;
        for (pi child : adj[parentNode])
        {
            int childCost = child.first;
            int childNode = child.second;
            if (dis[parentNode] + childCost < dis[childNode])
            {
                dis[childNode] = dis[parentNode] + childCost;
                pq.push({dis[childNode], childNode});
                par[childNode] = parentNode;
            }
        }
    }
```

(i) Select করা

(ii) Pop করে দাও

(iii) যদি Relax করতে পারো, queue তে push করা

Repeat until !pq.empty()