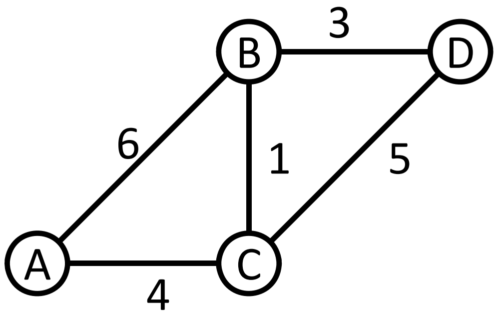
Dijkstra algorithm in time O(N^2), small example



Weight matrix

	A	B	C	D
A	0	6	4	0
B	6	0	1	3
C	4	1	0	5
D	0	3	5	0

Progress Table - inititial state

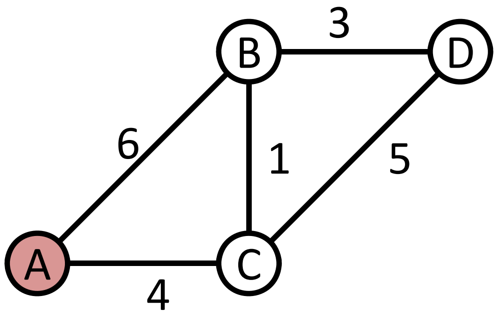
node	dist	pred	closed
A	0	null	false
B	inf	null	false
C	inf	null	false
D	inf	null	false

Dijkstra repeatedly updates the table

```
Init:   for each node X: dist[X] = infinity;  pred[X] = null;  dist[start] = 0; closed[X] = false;

for i in [1..N–1]
        1.  Find the open node X with smallest distance from start.             // O(N)
            minDist =  infinity
            for Y in nodes:            // Apply linear (slowest!) search,
                if   not(closed[Y])  and   dist[Y] < minDist:
                    X = Y; minDist = dist[Y]
       2.  // Recalculate the distances of all non-closed neighbours of X.     // O(dg(X)) ⊂ O(N)
             for YY in neighbours X:
                if dist(YY) > dist(X) + edgeWeight(YY, X)
                       dist(YY) =  dist(X) + edgeWeight(YY, X);   pred(YY) = X
       3.  closed[X] = true                                                       // O(1)
```

Dijkstra algorithm in time O(N²), small example



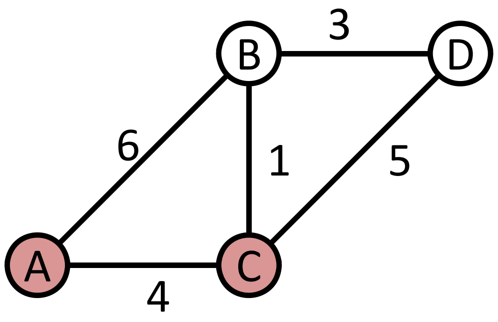
Weight matrix

	A	B	C	D
A	0	6	4	0
B	6	0	1	3
C	4	1	0	5
D	0	3	5	0

Progress Table

node	dist	pred	closed
A	0	null	true
B	6	A	false
C	4	A	false
D	inf	null	false

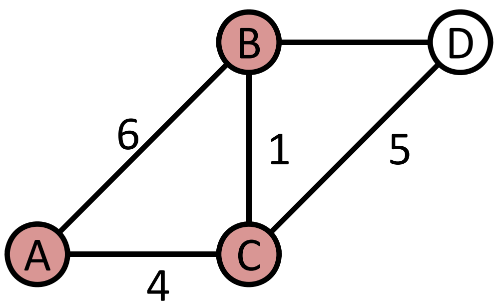
A is the closest, neighbours updated



	A	B	C	D
A	0	6	4	0
B	6	0	1	3
C	4	1	0	5
D	0	3	5	0

node	dist	pred	closed
A	0	null	true
B	5	C	false
C	4	A	true
D	9	C	false

C is the closest, neighbours updated



	A	B	C	D
A	0	6	4	0
B	6	0	1	3
C	4	1	0	5
D	0	3	5	0

node	dist	pred	closed
A	0	null	true
B	5	C	true
C	4	A	true
D	8	B	false

B is the closest, neighbours updated

closed

updated values in each step