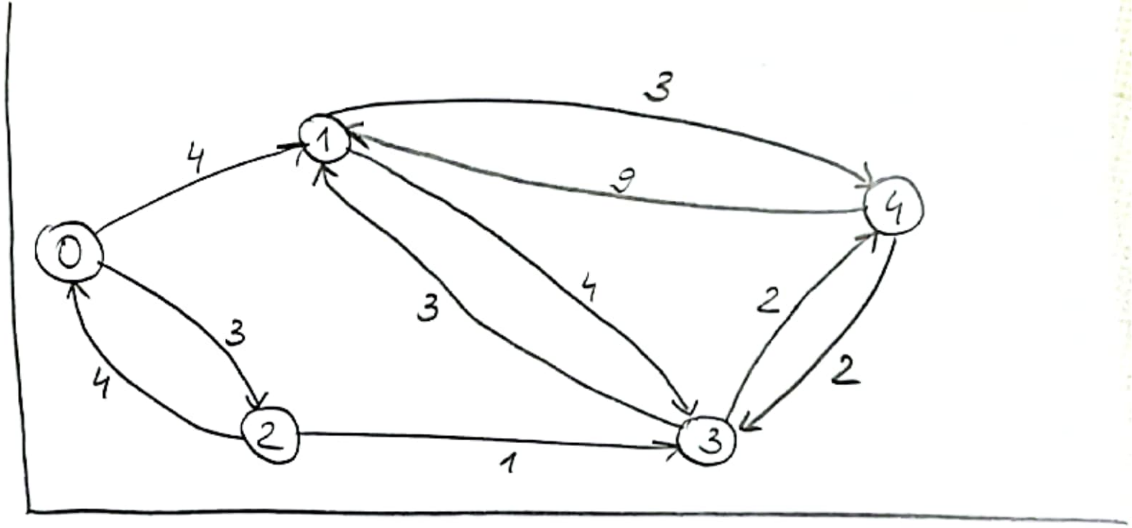
Dijkstra's algorithm =

$\boxed{s=0}$, $\boxed{t=4}$



Both s and t are in the graph
There are no negative cost edges } =>
-> the algorithm can be applied

	priority_queue	next	dist	cost	vertex	neighbour	found
init	← (0, 4) ←	{ 0: None, 1: None, 2: None, 3: None, 4: None }	{ 0: ∞, 1: ∞, 2: ∞, 3: ∞, 4: 0 }	—	—	—	False
it1	← ← ← (3, 1) ←	{ 0: None, 1: 4, 2: None, 3: None, 4: None }	{ 0: ∞, 1: 3, 2: ∞, 3: ∞, 4: 0 }	0	4	1	
it2	← (2, 3) (3, 1) ←	{ 0: None, 1: 4, 2: None, 3: 4, 4: None }	{ 0: ∞, 1: 3, 2: ∞, 3: 2, 4: 0 }			3	

	priority-queue	next	dist	cost	vertex	neighbour	found
i21 i22 i23	← | (3,1) | ← ← | (3, 1) | (3,2) | ←	{0: None, 1: 4, 2: 3, 3: 4, 4: None}	{0: ∞, 1: 3, 2: 3, 3: 2, 4: 0}	2	3	1 2 4	
i31 i32 i33	← | (3,2) | ← ← | (3,2) | (7,0) | ←	{0: 1, 1: 4, 2: 3, 3: 4, 4: None}	{0: 7, 1: 3, 2: 3, 3: 2, 4: 0}	3	1	0 3 4	
i41	← | (7,0) | ← ← | (6,0) | (7,0) | ←	{0: 2, 1: 4, 2: 3, 3: 4, 4: None}	{0: 6, 1: 3, 2: 3, 3: 2, 4: 0}	3	2	0	

	priority_queue	next	dist	cost	vertex	neighbour	found
	← (7,0) ←						
i51				6	0 (= start)	2	True

vertex == 0 == s ⇒ the algorithm stops. We reached the start vertex and found the minimum cost walk from s to t.

dist[0] = 6 ≠ ∞ ⇒ the vertex s is accessible from the vertex 0
Using the "next" dictionary we build the walk from s to t.

s = 0, next[0] = 2, next[2] = 3, next[3] = 4, next[4] = None

⇒ We get the walk:

$$0 \xrightarrow{3} 2 \xrightarrow{1} 3 \xrightarrow{2} 4, \quad \text{cost} = 6$$