

INF 1010

Estruturas de Dados Avançadas

Grafos



Percursos em grafos

em profundidade (*depth-first search*)

arestas que partem do vértice visitado por último

em largura (*breadth-first search*)

arestas que partem do vértice visitado primeiro

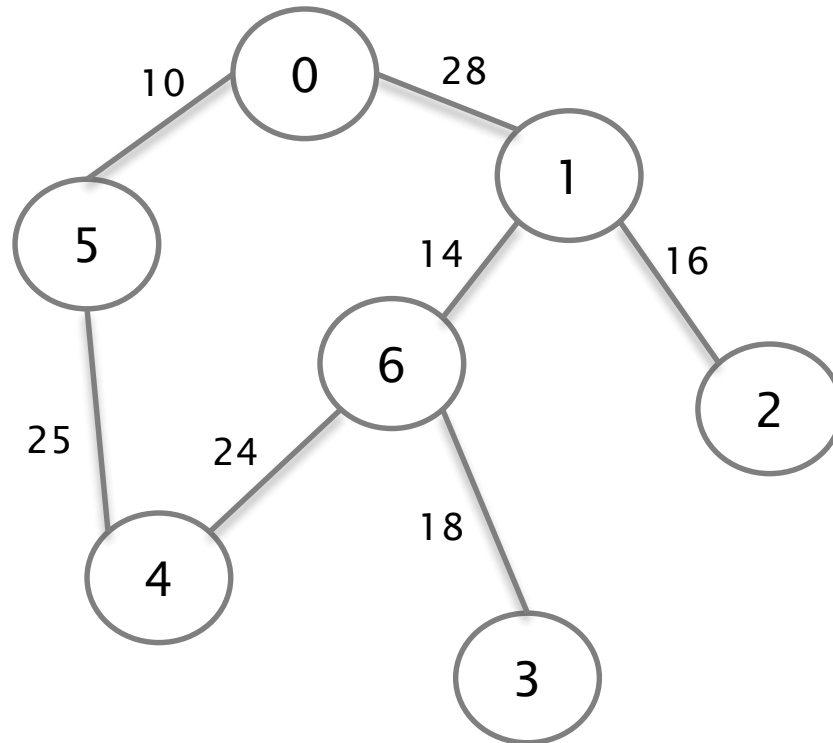
guloso (*greedy*)

arestas de menor custo, menor caminho



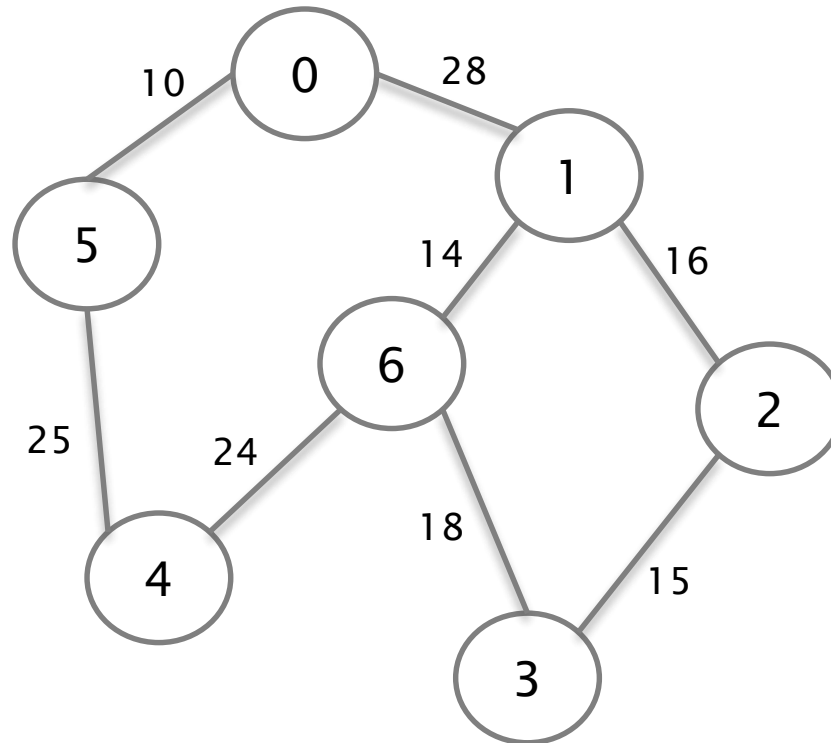
problemas comuns

- árvores geradoras (algoritmo de Kruskal, já visto)
- caminhos mais curtos (algoritmo de Dijkstra)



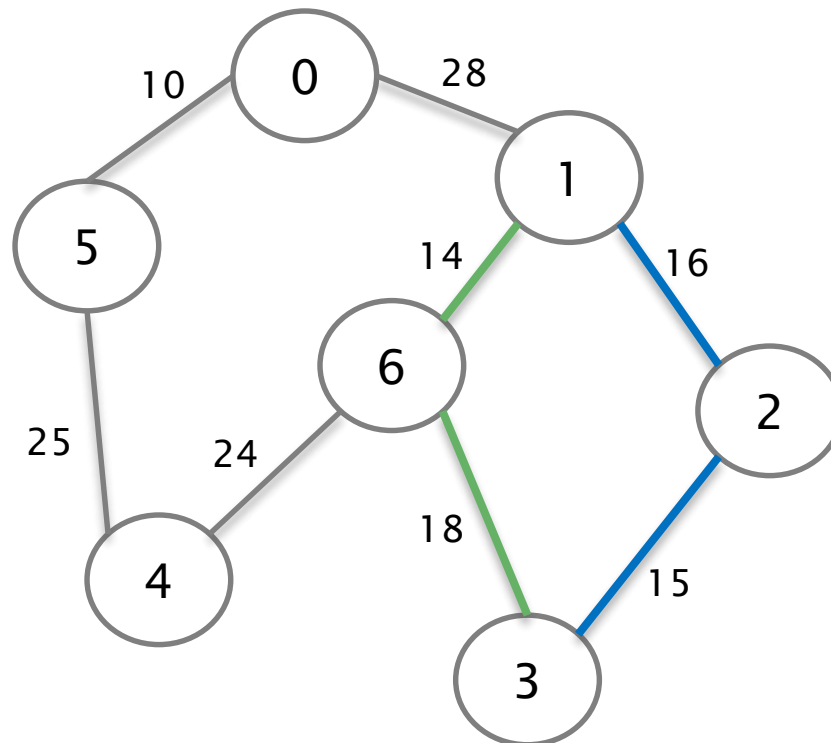
problemas comuns

- caminho mais curto
 - caminho entre nós i e j com menor peso total de arcos



problemas comuns

- caminho mais curto
 - caminho entre nós i e j com menor peso total de arcos



Algoritmo de Dijkstra

Algoritmo de Dijkstra

Entradas:

Um grafo ponderado $G = (V, E, p)$

Um vértice V do grafo

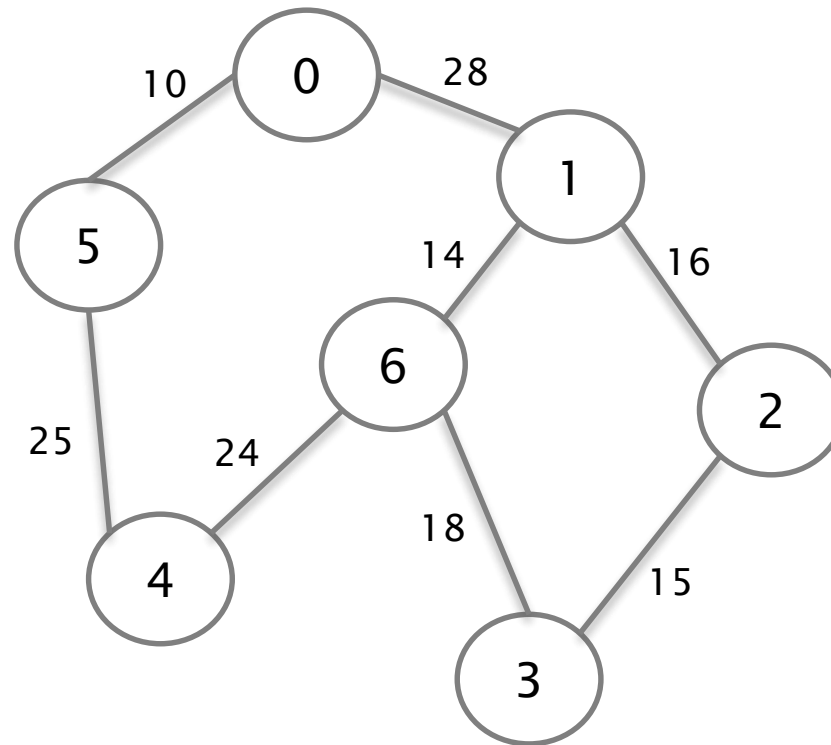
Saída:

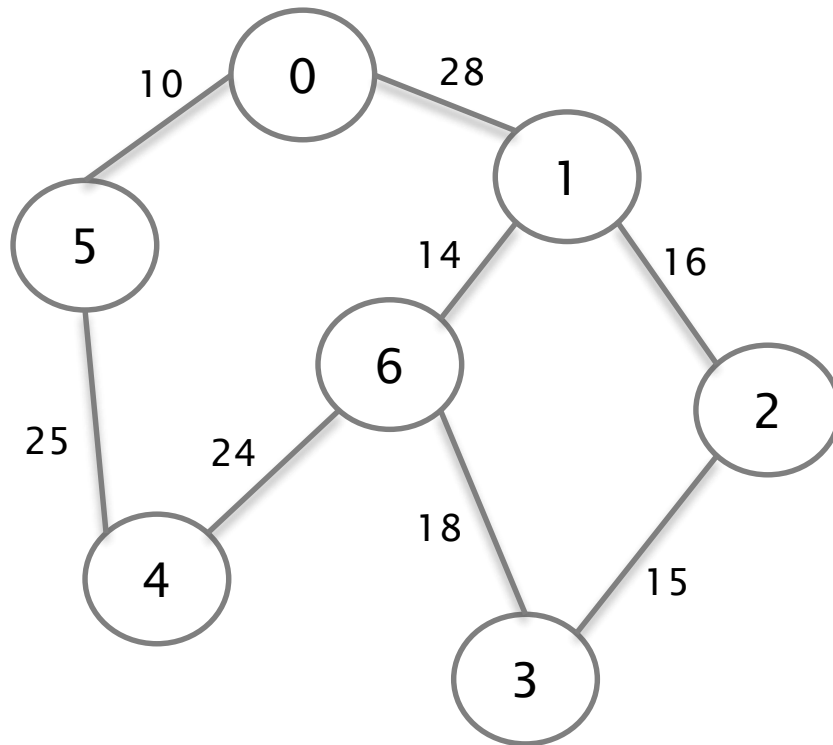
Menor caminho entre V e
cada um dos nós do grafo

- roteamento em redes
- deslocamento de caminhões em trânsito pesado
- desenho de chips
- roteamento de mensagens em telecomunicações
- ...



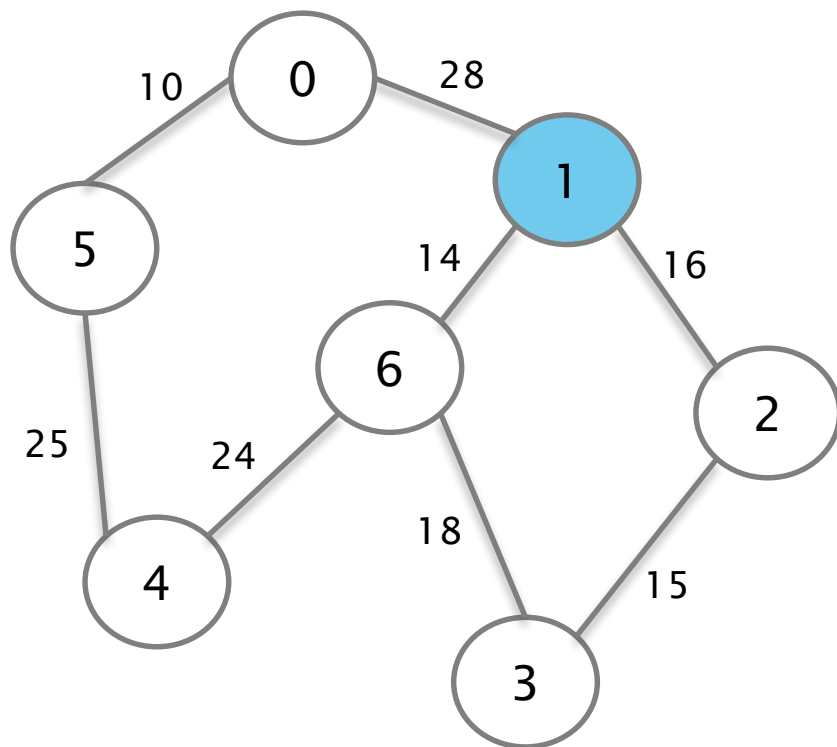
Dijkstra: caminhos mais curtos





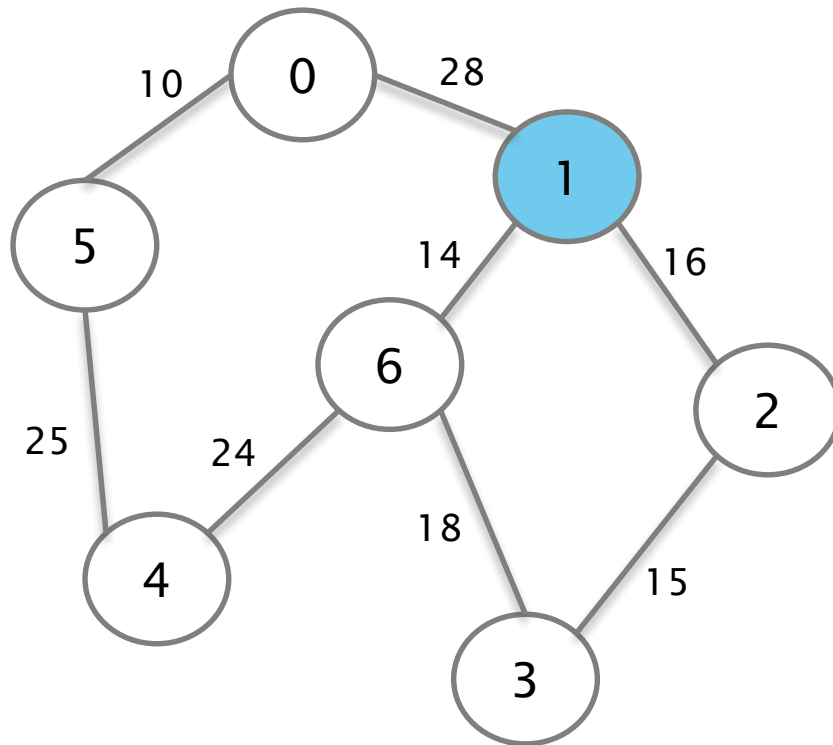
	dist
0	∞
1	0
2	∞
3	∞
4	∞
5	∞
6	∞





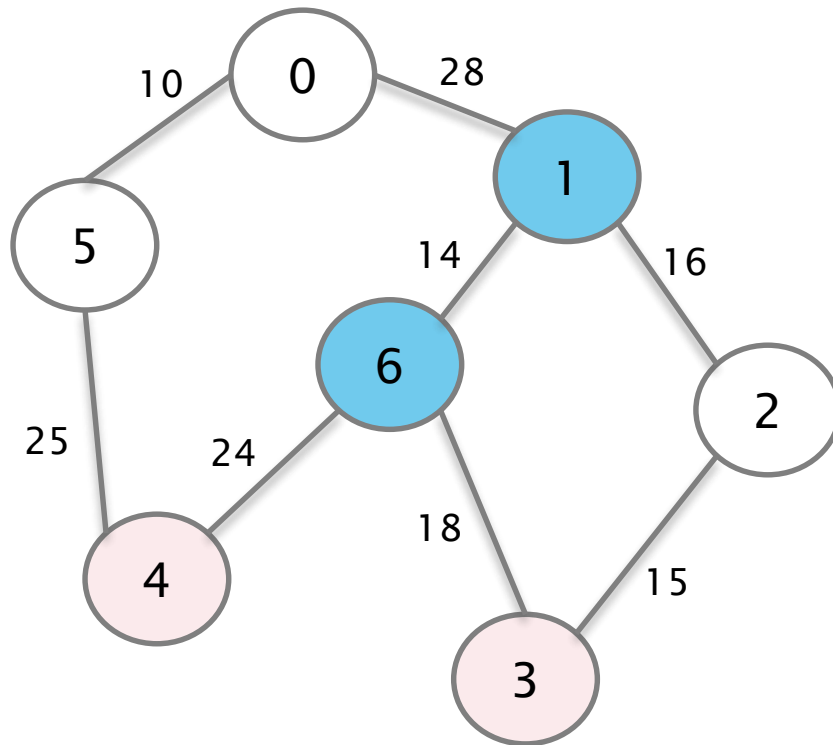
0	28
1	0
2	16
3	∞
4	∞
5	∞
6	14





0	28
1	0
2	16
3	∞
4	∞
5	∞
6	14

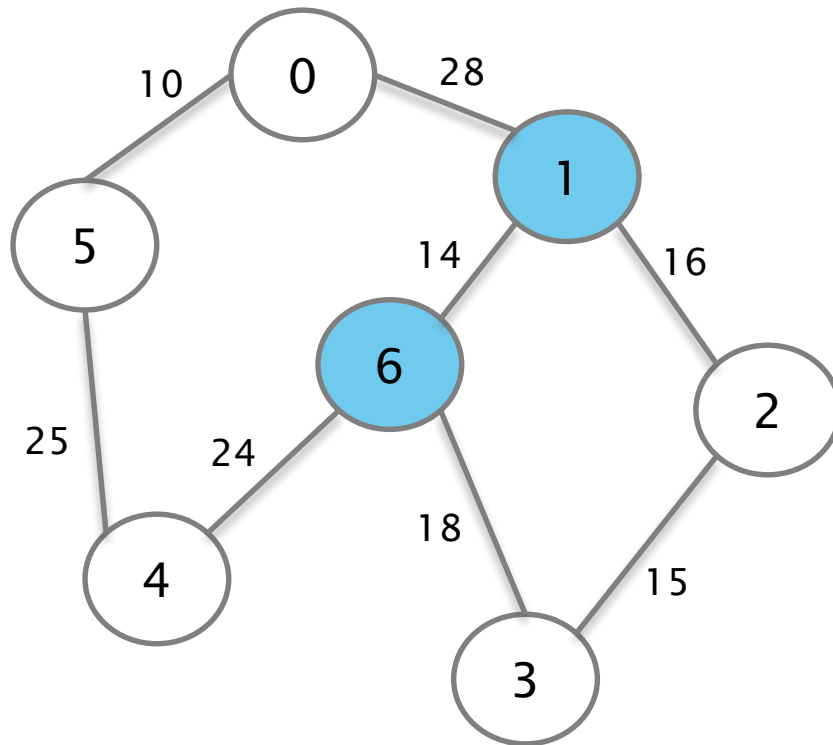




0	28
1	0
2	16
3	∞
4	∞
5	∞
6	14

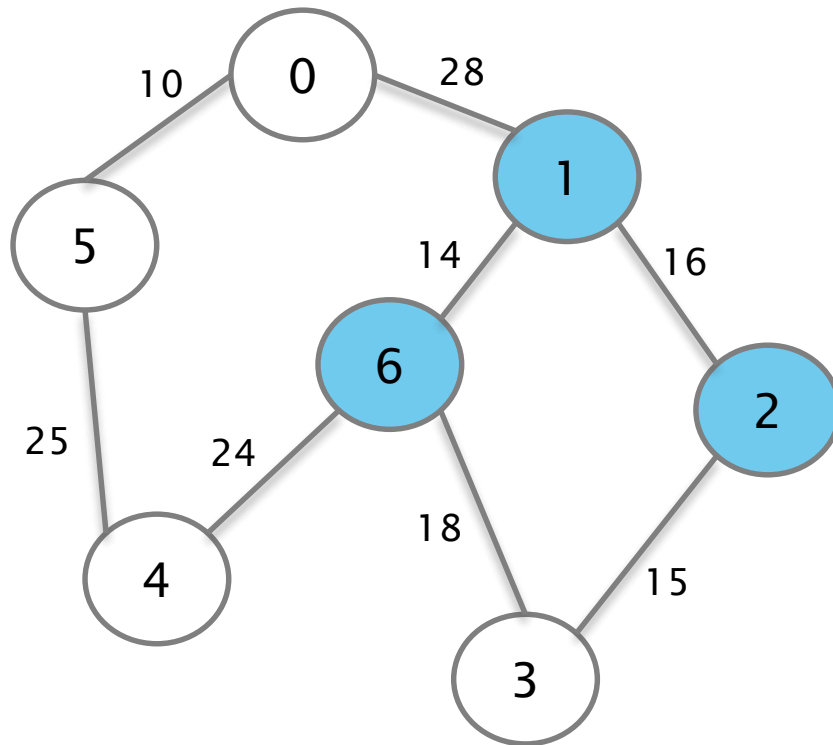
0	28
1	0
2	16
3	32
4	38
5	∞
6	14





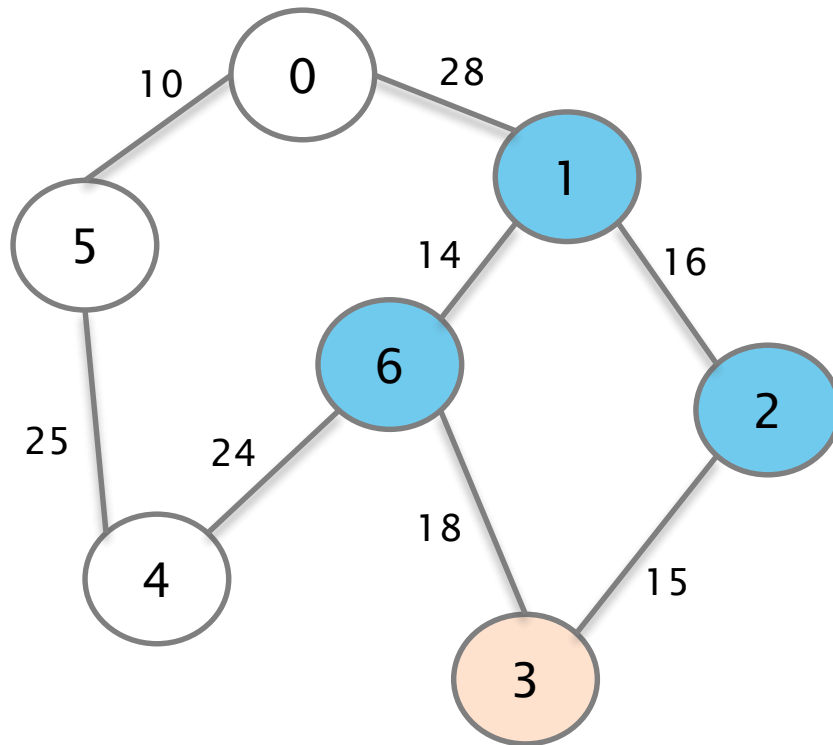
0	28
1	0
2	16
3	32
4	38
5	∞
6	14





0	28
1	0
2	16
3	32
4	38
5	∞
6	14

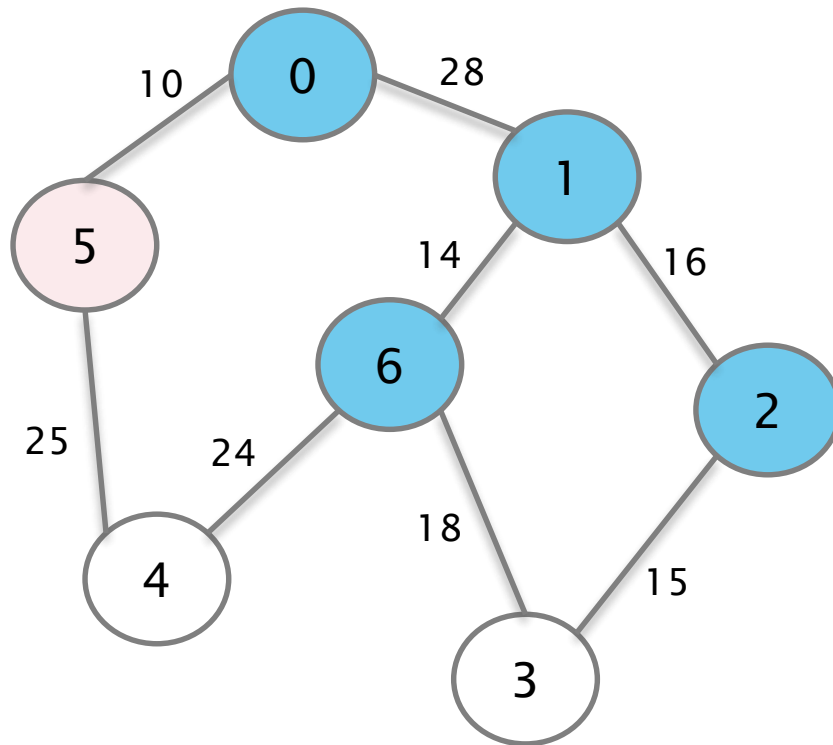




0	28
1	0
2	16
3	32
4	38
5	∞
6	14

0	28
1	0
2	16
3	31
4	38
5	∞
6	14

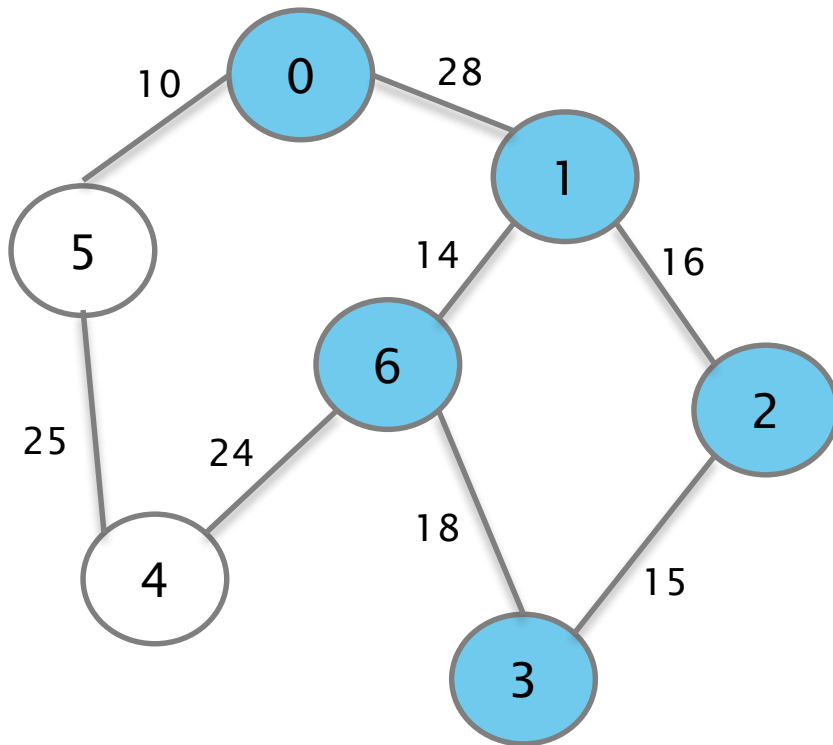




0	28
1	0
2	16
3	31
4	38
5	∞
6	14

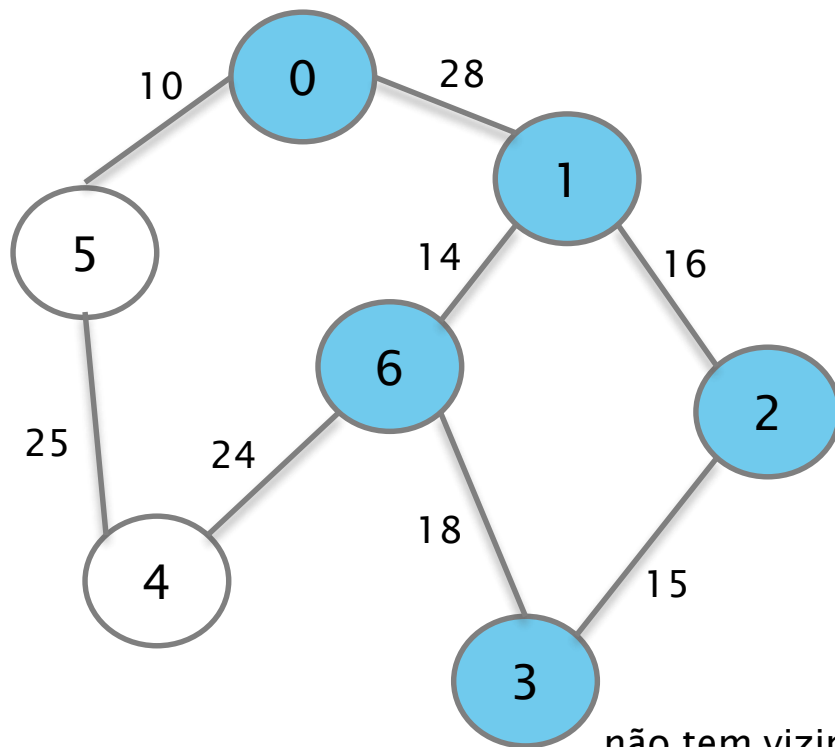
0	28
1	0
2	16
3	31
4	38
5	38
6	14





0	28
1	0
2	16
3	31
4	38
5	38
6	14

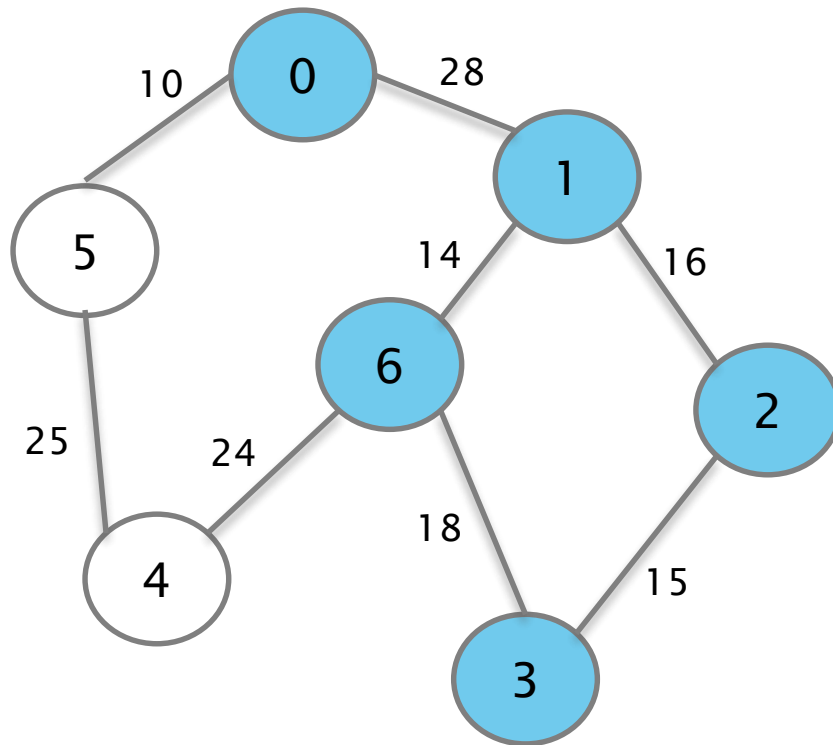




0	28
1	0
2	16
3	31
4	38
5	38
6	14

não tem vizinhos não visitados!





0	28
1	0
2	16
3	31
4	38
5	38
6	14



Algoritmo de Dijkstra

Algoritmo de Dijkstra /* Menor caminho entre um nó de origem e um de destino */

1. Defina o nó de origem
2. Atribua a todos os nós um valor de distância ao nó de origem: valor zero ao nó de origem, e *infinito* para todos os outros nós.
3. Marque todos os demais nós como não visitados e o nó origem como corrente (A).
4. Considere a distância de todos os nós vizinhos não visitados ao nó corrente e calcule uma distância deles ao nó origem através do nó corrente.
 1. Por exemplo, se o nó atual A tiver distância 6 e houver uma aresta de peso 2 conectando-o com um outro nó B, a distância de B através de A será 8.
 2. Se essa distância for menor do que a distância registrada anteriormente (infinito, na primeira rodada; zero para o nó de origem), sobrescreva a distância de B.

Ao final, tem-se a menor distância entre o nó de origem e cada um dos nós do grafo.



Algoritmo de Dijkstra

Algoritmo de Dijkstra /* Menor caminho entre um nó de origem e um de destino */

1. Defina o nó de origem
2. Atribua a todos os nós um valor de distância ao nó de origem: valor zero ao nó de origem, e *infinito* para todos os outros nós.
3. Marque todos os demais nós como não visitados e o nó origem como corrente (A).
4. Considere a distância de todos os nós vizinhos não visitados ao nó corrente e calcule uma distância deles ao nó origem através do nó corrente.
 1. Por exemplo, se o nó atual A tiver distância 6 e houver uma aresta de peso 2 conectando-o com um outro nó B, a distância de B através de A será 8.
 2. Se essa distância for menor do que a distância registrada anteriormente (infinito, na primeira rodada; zero para o nó de origem), sobrescreva a distância de B.
5. Ao terminar de considerar todos os vizinhos do nó atual A, marque-o como visitado. Um nó visitado não será mais verificado; sua distância registrada agora é final e mínima.
6. Se todos os nós tiverem sido visitados, termine. Caso contrário, marque o nó não visitado com a menor distância (ao nó de origem) como o próximo "nó corrente", e repita a partir do passo 4.

Ao final, tem-se a menor distância entre o nó de origem e cada um dos nós do grafo.



Algoritmo de Dijkstra

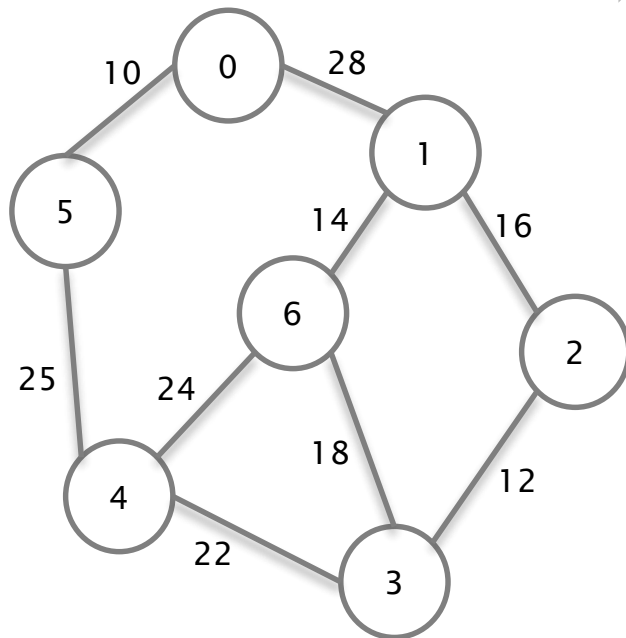
Complexidade temporal:

Depende da implementação do conjunto de vértices Q



Dijkstra – outro exemplo

Vértice inicial: 0



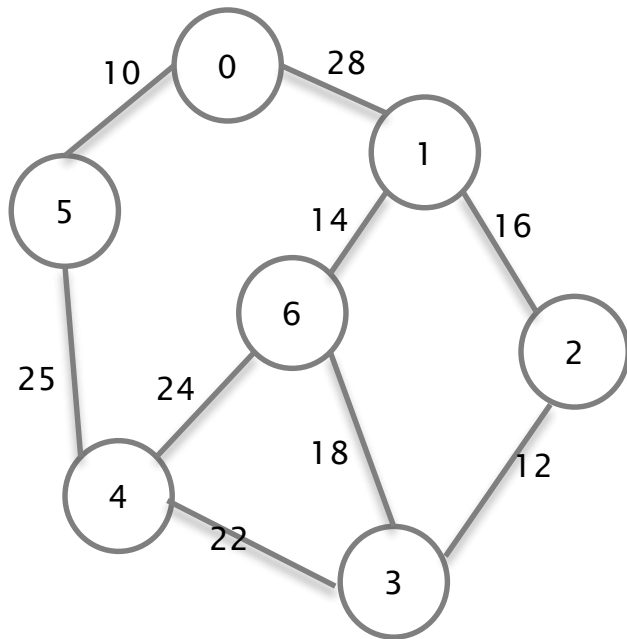
⇒ [0] ->

	dist	arcs	vertex	weight	link
[0] ->	0	→	1	28	→ 5 10 .
[1] ->	∞	→	0	28	→ 2 16 → 6 14 .
[2] ->	∞	→	1	16	→ 3 12 .
[3] ->	∞	→	2	12	→ 4 22 → 6 18 .
[4] ->	∞	→	3	22	→ 5 25 → 6 24 .
[5] ->	∞	→	0	10	→ 4 25 .
[6] ->	∞	→	1	14	→ 3 18 → 4 24 .



Dijkstra - exemplo

Modifica distâncias dos vértices 1 e 5



[0]->

[1]->

[2]->

[3]->

[4]->

[5]->

[6]->

dist	arcs	vertex	weight	link
0		1	28	→ 5 10 .
28		0	28	→ 2 16 → 6 14 .
∞		1	16	→ 3 12 .
∞		2	12	→ 4 22 → 6 18 .
∞		3	22	→ 5 25 → 6 24 .
10		0	10	→ 4 25 .
∞		1	14	→ 3 18 → 4 24 .



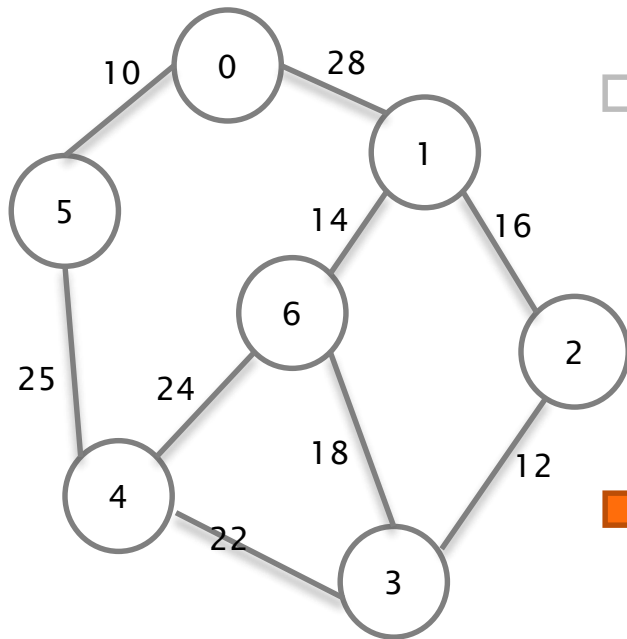
Dijkstra - exemplo

Marca o vértice 0 como visitado

Selecione o vértice 5

(vértice não visitado de menor distância à origem)

Ignore vértice 0 (já visitado) e modifique distância do vértice 4



[0]->

[1]->

[2]->

[3]->

[4]->

[5]->

[6]->

dist	arcs	vertex	weight	link
0		1	28	→ 5 10 .
28		0	28	→ 2 16 → 6 14 .
∞		1	16	→ 3 12 .
∞		2	12	→ 4 22 → 6 18 .
35		3	22	→ 5 25 → 6 24 .
10		0	10	→ 4 25 .
∞		1	14	→ 3 18 → 4 24 .



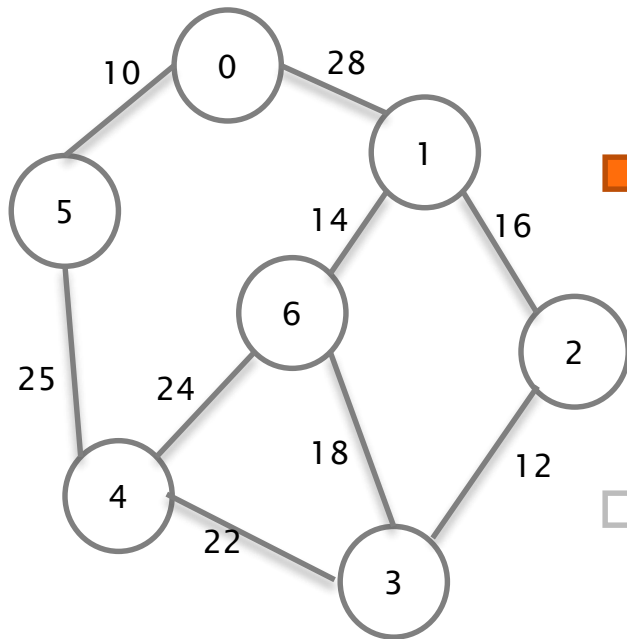
Dijkstra - exemplo

Marca o vértice 5 como visitado

Selecione o vértice 1

(vértice não visitado de menor distância à origem)

Ignore vértice 0 e modifique distâncias dos vértices 2 e 6



	dist	arcs	vertex	weight	link
[0]->	0		1	28	→ 5 10 .
[1]->	28		0	28	→ 2 16 → 6 14 .
[2]->	44		1	16	→ 3 12 .
[3]->	∞		2	12	→ 4 22 → 6 18 .
[4]->	35		3	22	→ 5 25 → 6 24 .
[5]->	10		0	10	→ 4 25 .
[6]->	42		1	14	→ 3 18 → 4 24 .



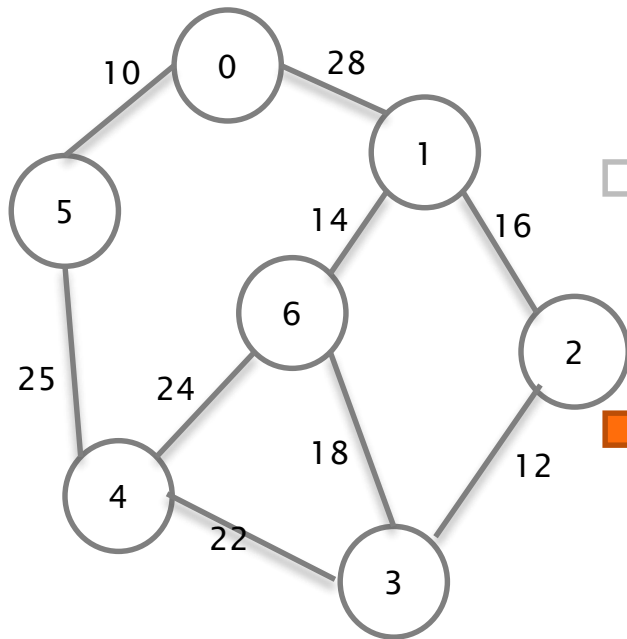
Dijkstra - exemplo

Marca o vértice 1 como visitado

Selecione o vértice 4

(vértice não visitado de menor distância à origem)

Modifica distância do vértice 3; ignora vértice 5;
mantém a distância do vértice 6



[0]->
[1]->
[2]->
[3]->
[4]->
[5]->
[6]->

dist	arcs	vertex	weight	link
0		1	28	5 10 .
28		0	28	2 16 6 14 .
44		1	16	3 12 .
57		2	12	4 22 6 18 .
35		3	22	5 25 6 24 .
10		0	10	4 25 .
42		1	14	3 18 4 24 .



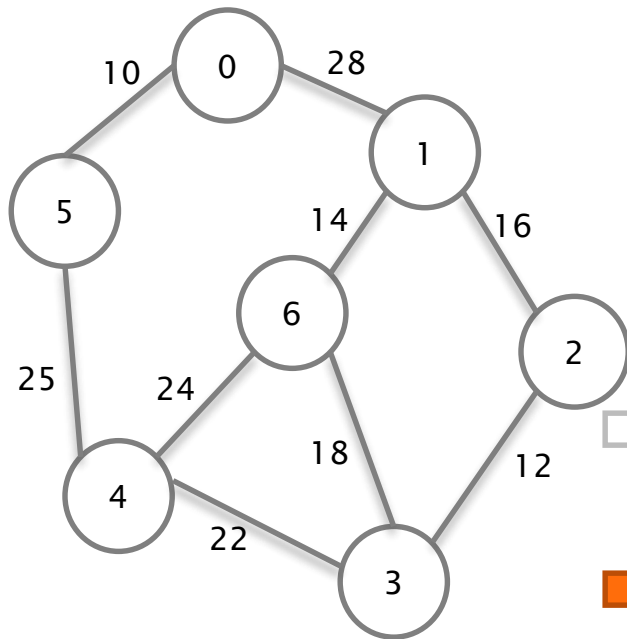
Dijkstra - exemplo

Marca o vértice 4 como visitado

Selecione o vértice 6

(vértice não visitado de menor distância à origem)

Ignore vértice 1; mantenha distância do vértice 3; ignore vértice 4



	dist	arcs	vertex	weight	link
[0]->	0		1	28	→ 5 10 .
[1]->	28		0	28	→ 2 16 → 6 14 .
[2]->	44		1	16	→ 3 12 .
[3]->	57		2	12	→ 4 22 → 6 18 .
[4]->	35		3	22	→ 5 25 → 6 24 .
[5]->	10		0	10	→ 4 25 .
[6]->	42		1	14	→ 3 18 → 4 24 .



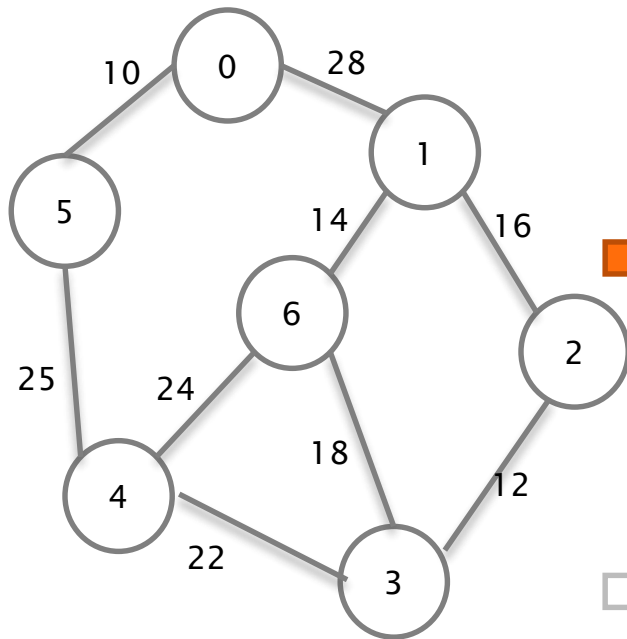
Dijkstra - exemplo

Marca o vértice 6 como visitado

Selecione o vértice 2

(vértice não visitado de menor distância à origem)

ignora vértice 1; modifica distância do vértice 3



[0]->

[1]->

[2]->

[3]->

[4]->

[5]->

[6]->

dist	arcs	vertex	weight	link
0		1	28	5 10 .
28		0	28	2 16 6 14 .
44		1	16	3 12 .
56		2	12	4 22 6 18 .
35		3	22	5 25 6 24 .
10		0	10	4 25 .
42		1	14	3 18 4 24 .



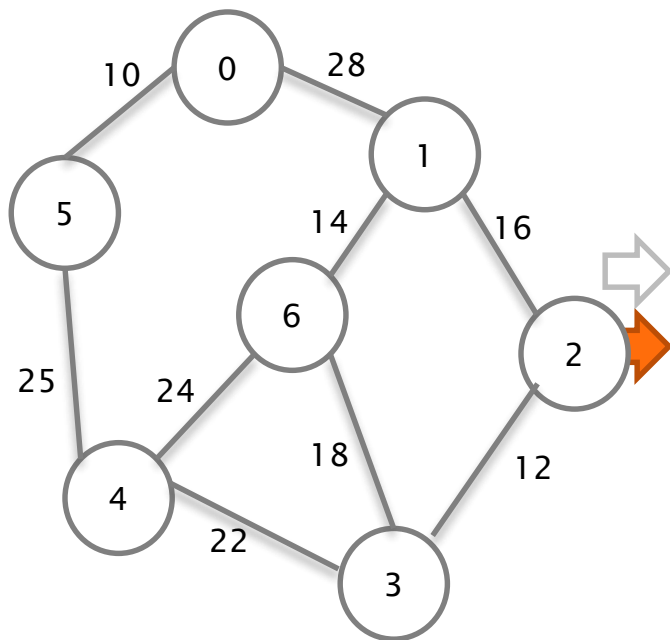
Dijkstra - exemplo

Marca o vértice 2 como visitado

seleciona o vértice 3

(vértice não visitado de menor distância à origem)

Ignora vértices 2, 4 e 6



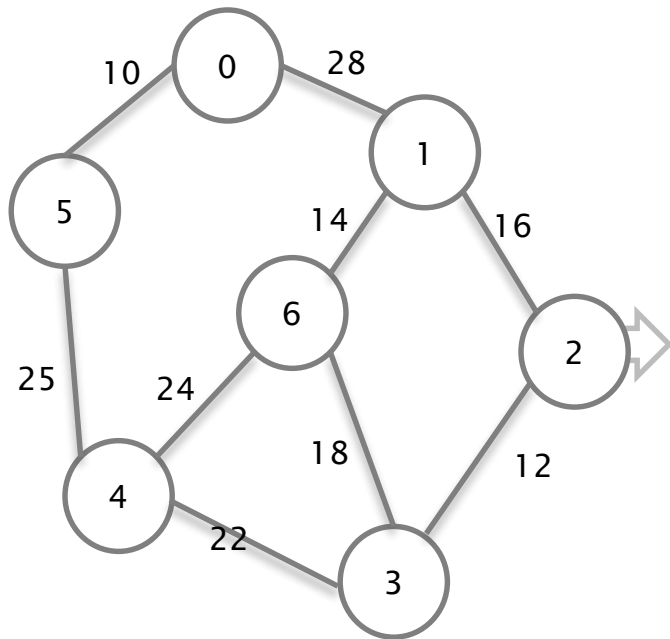
	dist	arcs	vertex	weight	link
[0]->	0		1	28	5 10 .
[1]->	28		0	28	2 16 6 14 .
[2]->	44		1	16	3 12 .
[3]->	56		2	12	4 22 6 18 .
[4]->	35		3	22	5 25 6 24 .
[5]->	10		0	10	4 25 .
[6]->	42		1	14	3 18 4 24 .



Dijkstra - exemplo

Marca o vértice 3 como visitado

Não há mais vértices não visitados – FIM!



	dist	arcs	vertex	weight	link
[0]->	0		1	28	→ 5 10 .
[1]->	28		0	28	→ 2 16 → 6 14 .
[2]->	44		1	16	→ 3 12 .
[3]->	56		2	12	→ 4 22 → 6 18 .
[4]->	35		3	22	→ 5 25 → 6 24 .
[5]->	10		0	10	→ 4 25 .
[6]->	42		1	14	→ 3 18 → 4 24 .

