Algoritmo di Dijkstra

L'algoritmo di Dijkstra calcola il cammino di peso minimo tra la sorgente s e ciascun nodo su un grafo con pesi *non negativi*. Approccio greedy.

```
DIJKSTRA(G, s)
 1  for v ∈ G.V
 2        v.dist = ∞
 3        v.pred = NIL
 4  s.dist = 0
 5  S = ∅                                          // solo per le dimostrazioni
 6  Q = coda di priorità contenente G.V
 7
 8  while Q ≠ ∅
 9        u = EXTRACT-MIN(Q)
10        S = S ∪ {u}
11        for v ∈ G.adj[u]
12              if v.dist > u.dist + (u,v).weight          // rilassa v.dist
13                    v.dist = u.dist + (u,v).weight       // DECREASE-KEY
14                    v.pred = u
```

Complessità

Supponendo che la coda sia implementata come min heap, l'inizializzazione ha costo $\Theta(V)$ (usando BUILD-MIN-HEAP), ed è l'unico punto in cui si inseriscono elementi in Q, quindi il ciclo di estrazione viene eseguito V volte. Il **for** al suo interno effettua complessivamente E iterazioni, ciascuna delle quali ha costo $O(\log V)$ (HEAP-DECREASE-KEY) e anche l'estrazione del minimo è $O(\log V)$. In totale $O(V + V \log V + E \log V) = O((V + E) \log V)$.