Dijkstra’s algorithm

```plaintext
procedure dijkstra(G, l, s)
  input: graph G = (V,E); node s;
  positive edge lengths l
  output: for each node u, dist[u] is set to its distance from s
  for u in V:
    dist[u] = 1
  dist[s] = 0
  H = makequeue(V)  // key = dist[
  while H is not empty:
    u = deletemin(H)
    for each edge (u,v) in E:
      if dist[v] > dist[u] + l(u,v):
        dist[v] = dist[u] + l(u,v)
        decreasekey(H,v)
```

Another example

```plaintext
procedure dijkstra(G, l, s)
  for u in V:
    dist[u] = 1
  prev[u] = nil
  dist[s] = 0
  H = makequeue(V)  // key = dist[
  while H is not empty:
    u = deletemin(H)
    for each edge (u,v) in E:
      if dist[v] > dist[u] + l(u,v):
        dist[v] = dist[u] + l(u,v)
        prev[v] = u
        decreasekey(H,v)
```

Running time

```
procedure dijkstra(G, l, s)
  for u in V:
    dist[u] = 1
  dist[s] = 0
  H = makequeue(V)  // key = dist[
  while H is not empty:
    u = deletemin(H)
    for each edge (u,v) in E:
      if dist[v] > dist[u] + l(u,v):
        dist[v] = dist[u] + l(u,v)
        decreasekey(H,v)
```

```
Time:
  O(V + E) +
  V x deletemin
  V x insert
  E x decreasekey

Depends on priority queue implementation:
  eg. binary heap O(E log V)
```

Linked list implementation

```
Linked list, unordered

insert:
  decreasekey:
  deletemin:
```

Binary heap

```
Complete binary tree: filled in row by row, left-to-right
Rule: each node’s value is smaller than that of its children
Height: \( \log_2 n + 1 \)
```

```
Binary heap

insert(7)
  decreasekey(19 -> 6)
  deletemin
```
Currency trading

You are a trader dealing in \( n \) currencies, \( c_1, c_2, \ldots, c_n \) (eg. $, £, ¥, €). The conversion rate from \( c_i \) to \( c_j \) is \( r_{ij} \): one unit of \( c_i \) becomes \( r_{ij} \) units of \( c_j \).

What is the best way to convert \( c_1 \) to \( c_n \)?

**Solution:** Create graph \( G = (V,E) \)

\[ V = \{c_1, c_2, \ldots, c_n\} \]

\[ E = \{(c_i, c_j) : 1 \leq i, j \leq n\} \]

Edge lengths \( l(e) = -\log r_{ij} \)

Now: find shortest path from 1 to \( n \)

Claim: the length of a path is

\[-\log (\text{value of converting along path})\]

Problem: some edge lengths are negative!