BidirectionalDijkstra(G, s, t)

```
G^R ← ReverseGraph(G)
Fill dist, dist^R with +∞ for each node
dist[s] ← 0, dist^R[t] ← 0
Fill prev, prev^R with None for each node
proc ← empty, proc^R ← empty
do:
  v ← ExtractMin(dist)
  Process(v, G, dist, prev, proc)
  if v in proc^R:
    return ShortestPath(s, dist, prev, proc, t, ...)
  v^R ← ExtractMin(dist^R)
  repeat symmetrically for v^R as for v
while True
```

Relax($u, v, \texttt{dist}, \texttt{prev}$)

if $\texttt{dist}[v] > \texttt{dist}[u] + w(u, v)$:
  $\texttt{dist}[v] \leftarrow \texttt{dist}[u] + w(u, v)$
  $\texttt{prev}[v] \leftarrow u$

Process($u, G, \texttt{dist}, \texttt{prev}, \texttt{proc}$)

```
for (u, v) ∈ E(G):
    Relax(u, v, dist, prev)
proc.Append(u)
```

ShortestPath(s, dist, prev, proc, t, dist^R, prev^R, proc^R)

```
distance ← +∞,  u_best ← None
for u in proc + proc^R:
    if dist[u] + dist^R[u] < distance:
        u_best ← u
        distance ← dist[u] + dist^R[u]
path ← empty
last ← u_best
while last ≠ s:
    path.Append(last)
    last ← prev[last]
path ← Reverse(path)
last ← u_best
while last ≠ t:
    last ← prev^R[last]
    path.Append(last)
return (distance, path)
```