Algorithm 1 d2dDistance(Source door d_s, destination door d_t)

initialize a min-heap H
for each door $d_i \in \mathcal{S}_{door}$ **do**
  if $d_i \neq d_s$ **then**
    $dist[d_i] \leftarrow \infty$
  else
    $dist[d_i] \leftarrow 0$
  enheap($H, \langle d_i, dist[d_i] \rangle$)
  $prev[d_i] \leftarrow$ null
while H is not empty **do**
  $\langle d_i, dist[d_i] \rangle \leftarrow$ deheap(H)
  if $d_i = d_t$ **then**
    return $dist[d_i]$
  mark door d_i as visited
  $parts \leftarrow D2P_{\sqsubset}(d_i)$
  for each partition $v \in parts$ **do**
    for each unvisited door $d_j \in P2D_{\sqsubset}(v)$ **do**
      if $dist[d_i] + G_{dist}.f_{d2d}(v, d_i, d_j) < dist[d_j]$ **then**
        $dist[d_j] \leftarrow dist[d_i] + G_{dist}.f_{d2d}(v, d_i, d_j)$
        replace d_j's element in H by $\langle d_j, dist[d_j] \rangle$
        $prev[d_j] \leftarrow (v, d_i)$