VE281 Writing Assignment Five

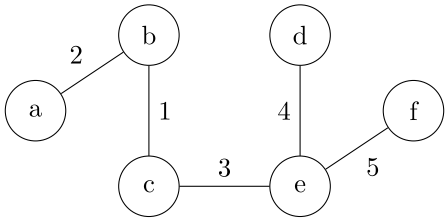
Liu Yihao 515370910207

Ex. 1

In Kruskal's algorithm, we take the shortest edge and connect two nodes if it doesn't form a cycle.

1. Connect b and c
2. Connect a and b
3. Connect c and e
4. Connect e and f
5. Connect e and d

The minimum spanning tree is



Ex. 2

```
**Input:**
  A directed acyclic graph $G = (V, E)$ with real-valued edge weights
  Two distinct nodes $s$ and $d$
**Output:**
  A longest weighted path from $s$ to $d$ if exists

  $L \leftarrow G$ sorted in topological order
  Remove nodes located before $s$ or after $d$ from $L$
  Remove node $s$ from $L$
  $s.distance \leftarrow 0$
  $s.predecessor \leftarrow null$
  for node $v$ in $L$ do
      $v.distance \leftarrow -\infty$
      $v.predecessor \leftarrow null$
      for edge $(u, v)$ in edges with end node $v$ do
          if $u.distance + (u, v).weight > v.distance$ then
              $v.distance \leftarrow u.distance + (u, v).weight$
              $v.predecessor \leftarrow u$
          end if
      end for
  end for
  if $d.predecessor == null$ then
      print “No path exists”
  else
      print $d.predecessor$ recursively in reverse order
  end if
```

The time complexity is $O(V + E)$.

Ex. 3

```
Input:
  A directed graph $G = (V, E)$ with real-valued edge reliability in the range $[0, 1]$
  Two distinct nodes $s$ and $d$
Output:
  A most reliable path from $s$ to $d$ if exists

  for node $u$ in $G$ do
      $u.reached \leftarrow false$
      $u.probability \leftarrow 0$
      $u.predecessor \leftarrow null$
  end for
  $s.probability \leftarrow 1$
  push node $s$ into set $S$
  while Set $S$ is not empty do
      $u \leftarrow$ pop the node with largest reliability in $S$
      $u.reached \leftarrow true$
      for edge $(u, v)$ in edges with start node $u$ do
          if not $v.reached$ and $u.probability * (u, v).reliability > v.probability$ then
              $v.probability \leftarrow u.probability * (u, v).reliability$
              $v.predecessor \leftarrow u$
          end if
      end for
  end while
  if $d.predecessor == null$ then
      print “No path exists”
  else
      print $d.predecessor$ recursively in reverse order
  end if
```

Ex. 4

Input:
A connected, undirected graph $G = (V, E)$

Output:
A path that traverses edge in E exactly once in each direction.

```
for node $u$ in $G$ do
    $u.reached \leftarrow false$
    $u.depth \leftarrow 0$
end for
$s \leftarrow$ an arbitrary node in $G$
DFS($s$)

function DFS(node $u$)
    $u.reached \leftarrow true$
    for edge $(u, v)$ in edges adjacent to $u$ do
        if not $v.reached$ then
            $v.depth \leftarrow u.depth + 1$
            traverse $u \rightarrow v$
            DFS($v$)
            traverse $v \rightarrow u$
        else if $v.depth > u.depth$ then
            traverse $u \rightarrow v$
            traverse $v \rightarrow u$
        end if
    end for
end function
```

Ex. 5