Algorithm 1 Dijkstra's algorithm

```
Dijkstra $\left(G = \left(V, E, w : V \times V \to \mathbb{R}^{\geq 0}\right),\ s\right)$ :
        $d[v],\ \pi[v] \leftarrow \infty,\ \text{None}\ \forall v \in V$
        $d[s],\ \pi[s] = 0,\ s$
        $S,\ Q\ = \emptyset,\ \emptyset$
    (1) $Q.\mathsf{add}\left((v, d[v])\right)\ \forall v \in V$
        while $Q \neq \emptyset$
          (2) $u \leftarrow Q.\mathsf{ExtractMin}\left(\right)$
              $S \leftarrow S \cup \{u\}$
              For each $v \in N_G(u)$ :
              (3) If $v$ not in $S$ :
                     A. $\mathsf{Relax}\left(w, u, v, d, \pi\right)$
                    (4) $Q.\mathsf{Update}\left((v, d[v])\right)$
     1. return $d, \pi$
Relax$(u, v, w, d, \pi)$ :
        If $d[v] > d[u] + w(u, v)$:
              $d[v] \leftarrow d[u] + w(u, v)$
              $\pi[v] \leftarrow u$
```

Programming Assignment 6

Recall Dijkstra's algorithm for finding single source shortest paths in a graph with non-negative edge weights. The time-consuming procedures in this algorithm are labeled (1-4).

The $O\left((V + E)\log(V)\right)$ time complexity of Dijkstra's algorithm is dependent on implementing $(1 - 4)$ with special care. Most notably, a naive implementation of queue update may require up to $O(V)$ time at a given iteration - since finding a key in a priority queue is an $O(V)$ time operation.

Programming

Implement Dijkstra's algorithm using $O\left((V + E)\log V\right)$ time. We will use networkx as our graph library, which can be installed via *pip install networkx* or *conda install -c anaconda networkx.* You may also use standard libraries for the Priority Queue data structure (*import heapq*), but you must figure out how to implement update/extract min so that the time complexity of your implementation is $O\left((V+E)\log V\right)$ worst case. Point totals for each part are listed in the python files. Implementation of DIJKSTRA algorithm (8 points), implementation of RELAX Your function signatures DO NOT need to match those given in the pseudocode. Your code DOES NOT need to match the pseudocode - it is meant as a reference. You may write additional helper functions as needed. You may add additional arguments to any function provided in the python file except for the *dijkstra* function, which must take G, w, s as input.