Introduction to Algorithms	Problem Set 2
CS 4820 Spring 2016	Lillyan Pan

1 Problem 3

1.1 Algorithm

To ensure the implementation is efficient, we start by setting up an adjacency list to represent the graph, so that Breadth First Search can run in $O(m + n)$ time where m is the number of edges and n is the number of nodes. We will also use an array where the value at index i correlates to the cost of edge c_{i+1} for pre-processing. Let edge $e = (u, v)$.

1. From given graph $G = (V, E)$, create graph G' in which all edges with cost greater that c_e and edge e are removed
2. Use BFS to determine if there is a path from u to v
3. If there is a path from u to v
   - Then nodes u, v can be joined by a path of edges that are all cheaper then e. Hence, e is not a part of the minimal spanning tree
4. Else e is included in the minimal spanning tree

1.2 Runtime

The adjacency list can be made in $O(m)$ time by iterated though all the edges. Similarly, the cost array can be built by iterating through all $e \in E$ and adding c_e to the array, which will take $O(m)$ time. Creating G' has a bound of $O(m+n)$ (if there are no edges in the graph that have a cost greater than c_e). Breadth First Search can run in $O(m+n)$ time where m is the number of edges and n is the number of nodes as the graph is represented as an adjacency list, which is also given in the book. Hence, total running time is bounded by $O(m + n)$.

1.3 Correctness

Lemma 1. *Edge $e = (u, v)$ is contained in the minimum spanning tree T of the graph G if there does not exists a path from u to v made of only cheaper edges compared to c_e.*

Proof of Lemma 1. Assume there exists a path p from u to v that consists of edges that all cost cheaper than e. Now, add e to path p to create cycle C. Hence, e is the most expensive edge in C, and so by the Cycle Property, e is in no minimum spanning tree of G, hence e is not contained in T.

Then, assume there does not exist such a path p. Let S be the set of nodes reachable from u using edges that are cheaper than e. Then let $V - S$ be the set of nodes that contains v. Note, that $V - S$ cannot contain an edge that is cheaper than e, $e' = (u', v')$ in which $u' \in S$ and $v' \in V - S$ as u' and v' would be reachable from u using strictly cheaper edges than e. Hence, e is the cheapest edge crossing from set S to complement $V - S$. Thus, by the Cut Property, e is in every minimum spanning tree of G, which follows that e is contained in T.

Therefore, we have proved Lemma 1 as needed.