

CSCI 256: Exercise # 7

Due: Wed, Sept 30, in class

Your Name: _____

Assigned reading for this lecture:

- Kleinberg Tardos Section 2.5 (Priority Queues)

To implement Dijkstra's Algorithm efficiently we need to be able to repeatedly find vertices not in S with the minimum d -value. We also need a way to efficiently update these d -values (as the algorithm finds shorter path estimates). A priority queue is a data structure that supports these operations. In particular, a priority queue is a data structure to store a set of elements A where each $a \in A$ has an associated priority-value or **key**. A priority queue supports the following operations:

- **Insert**: add an element into A with a key k
- **ExtractMin**: delete and return the element from A with min key value
- **ChangeKey**: update key k of an existing element in A

A priority-queue can be implemented using a binary heap. Review the description of binary heap in KT Section 2.5 and answer the following questions.

Questions: (a) State the complexity of performing **ExtractMin** and **ChangeKey** operations in a binary heap.

(b) Consider a binary **min-heap** stored in the array $[2, 5, 4, 9, 7, 6]$. Draw the heap as a binary tree.

(c) Insert the key 3, *Heapify-Up* and show the resulting binary tree and array.