CS 3350 Assignment 5 Due: Friday, April 21, 11:59 p.m.

# Q1: Check if a binary tree is a min-heap or not (10 points)

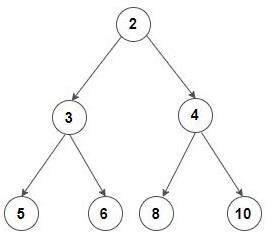
**Goal(s**): Practice with the heap data structure and preorder traversal of trees

## What you are given:

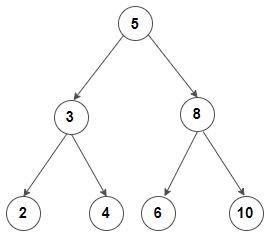
* The file Q1-isHeap.cpp

## What you should do:

Given a binary tree, check if it is a [min-heap](https://www.techiedelight.com/introduction-priority-queues-using-binary-heaps/#Heap) or not. In other words, the binary tree must be a complete binary tree where each node has a higher value than its parent’s value. For example, the following binary tree is a min-heap:



On the other hand, the following binary tree is not a min-heap:



Complete the file Q1-isHeap.cpp. Please note that you must provide a **recursive**

implementation.

# Q2: Find k’th largest element in an array (10 points)

**Goal(s**): Practice with the min-heap data structure.

## What you are given:

* The file Q2-k-largest.cpp

## What you should do:

Given an integer array, find k’th largest element in the array where k is a positive integer less than or equal to the length of array. For example:

**Input:**

arr = [7, 4, 6, 3, 9, 1]

k = 2

**Output:**

The 2nd largest array element is 7

Complete the file Q2-k-largest.cpp. Please note that you must provide an implementation using the min-heap data structure. You may create auxiliary functions as needed and use the heapify() function as presented in the class.

**What to submit**

* Completed Q1-isHeap.cpp
* Completed Q2-k-largest.cpp

**Grading Rubric**

\_\_\_\_\_\_\_ (4) proper documentation/comments where appropriate.

\_\_\_\_\_\_ (8) implementation of Q1.

\_\_\_\_\_\_\_ (8) implementation of Q2