COMP2119A Introduction to Data Structures and Algorithms
Homework 2 **Due Date:** 7pm, Oct 30, 2018

Rules: discussion of the problems is permitted, but writing the assignment together is not (i.e. you are not allowed to see the actual pages of another student). Please make your answers precise and concise.

Course Outcomes

- **[O1]. Mathematics foundation**
- **[O2]. Basic data structures**
- **[O3]. Problem solving**
- **[O4]. Implementation**

1. **[O1]** (20 pts)

   (a) (5 pts) If we have 60000 items to store in a hash table **H-Open** using open addressing with linear probing and we desire a load factor of 0.75, how big should the hash table be?

   (b) (5 pts) If we have 60000 items to store in a hash table **H-Chaining** using chaining and we desire the expected number of comparisons in a successful search to be 2.5, how big should the hash table be?

   (c) (10 pts) Analyse and compare the memory usage for **H-Open** and **H-Chaining**. Suppose item's size and pointer's size are both 8 bytes.

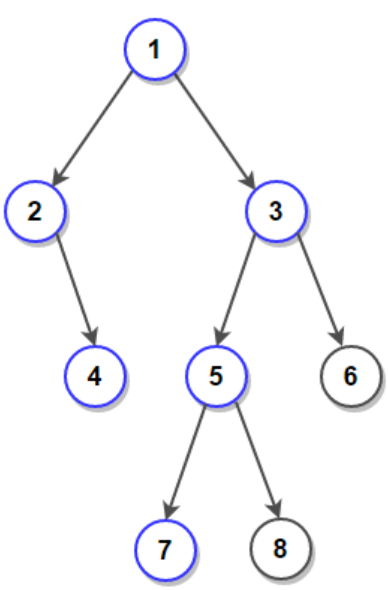
2. **[O2]** (20 pts) Given the following pre-order traversal sequence: $1, 2, 4, 5, 7, 3, 6, 8, 9, 10$ and in-order traversal sequence: $4, 2, 7, 5, 1, 3, 9, 8, 10, 6$ of a binary tree, draw the binary tree.

   Suppose that all nodes in a binary tree have distinct values, given its pre-order traversal sequence and in-order traversal sequence, does there exist a different binary tree with the same pre-order traversal sequence and in-order traversal sequence? Justify your answer.

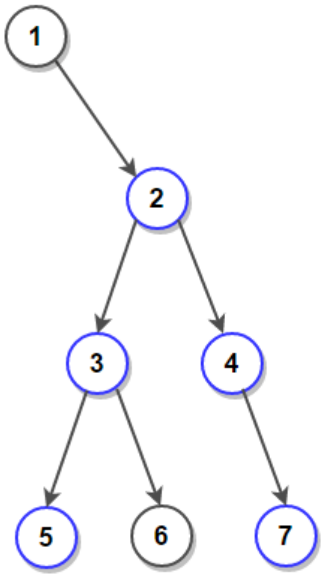
3. **[O3, O4]** (20 pts) Given a binary tree, write an efficient algorithm to compute the diameter of it. The diameter of a binary tree is equal to the number of nodes on the longest path between any two leaves of it.

   For example, below figures shows two binary trees having diameter 6 and 5 respectively(nodes highlighted in blue color.)

   What is the time complexity of your algorithm? How much extra space does your program need? Prove your claim.



Diameter through the root node



Diameter not through root node

4. **[O4]** (20 pts)

   Given a binary search tree, write a function $kthSmallest(TreeNode*,\ int)$ in $C/C++$ style to find the k-th smallest element in it.

   Note: You may assume k is always valid, $1 \leq k \leq$ number of BST's total elements.

```
/**
* Definition for a binary tree node.
**/

struct TreeNode {
  int val;
  TreeNode *left;
  TreeNode *right;
  TreeNode(int x) : val(x), left(NULL), right(NULL) {}
};

int kthSmallest(TreeNode* root, int k) {
// you code starts here




}
```

5. **[O1]** (20 pts) Suppose a given tree T has n_1 nodes that have 1 child, n_2 nodes that have 2 children, . . . , n_m nodes that have m children and no node has more than m children, how many nodes have NO child are there in T?