National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Data Structures**	**Course Code:**	**CS 201**
Program:	**BS(Computer Science)**	**Semester:**	**Fall 2018**
Duration:	**180 Minutes**	**Total Marks:**	**80**
Paper Date:	**21-Dec-2018**	**Page(s):**	**8**
Section:	**ALL**	**Section:**	
Exam:	**Final Exam**	**Roll No:**	

Instruction/Notes: Answer in the space provided
You can ask for rough sheets but **they will not be graded or marked**
In case of confusion or ambiguity make a reasonable assumption.
Questions are not allowed

Good luck!

Question 1: **(Marks: 5 * 7)**

a. Suppose, you are going to design an application for stock exchange, where prices of different shares (stocks) will be stored and processed. Select the best suitable and most efficient data structure for following requirements. Also, provide time complexity in big (Oh) form, for all required operations.

1. User can buy a share, which price is closest to the price provided by user.
2. User can view a list of all the shares, which prices lie in provided range.
3. User can search and sale five of those shares, which price is less than or equal to the given price.

b. WhatsApp is a mobile app that allow its users to send and receive text, audio and video messages. The WhatsApp team wish to find whether a messages reach back to its sender or not?
For example, a person P sends a message to his friends and they forward the message to their friends and so on. Given the users of WhatsApp and the information of all the messages sent by its users to other users, we need to determine that a message is sent back to its user or not for a particular user. What data structure is most suitable to represent this data and why? How would you solve this problem with your suggested data structure? Briefly explain your idea.

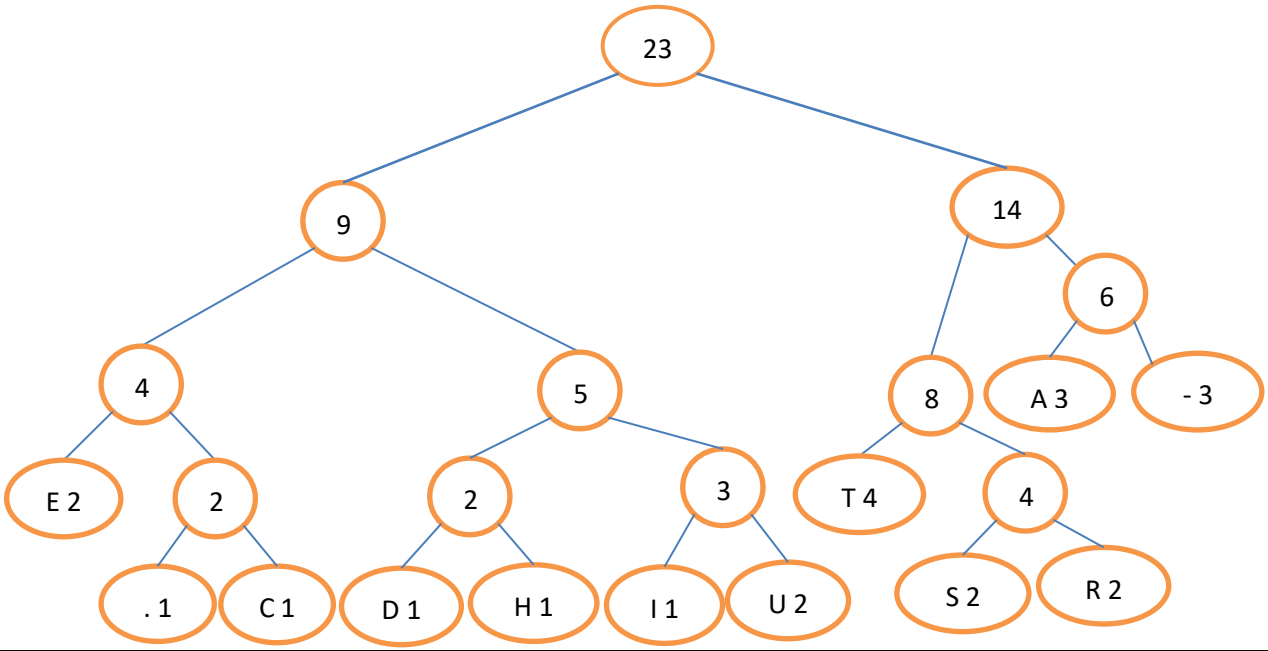
c. Suppose the department of computer science is offering n sections of the course Linear Algebra. The timetable has already been setup by the computer science department. The department of mathematics can spare only 2 teachers to teach Linear Algebra to computer science students. How would you determine that 2 teachers are sufficient to teach all n sections given the time table of all Linear algebra classes? You can assume that a teacher can teach multiple sections of the course as long as their class timings does not overlap in the time table. What data structure is most suitable to represent this data and why? How would you solve this problem with your suggested data structure? Explain your idea (not code) in 3-4 lines

d. Suppose you are implementing a web browser that has a collection of malicious websites and whenever a user try to visit a web page, your browser must check whether that website is malicious or not. What data structure is most suitable to represent this data and why? How would you solve this problem with your suggested data structure? Briefly explain your idea.

e. For which character frequencies of encoding data, Huffman algorithm always generates a skewed binary tree.

Roll Number:_____ Section:_____

f. Decode the string of binary data given below using following tree generated by Huffman Algorithm.



01101111010100101101110011100011110110001010111010011010011011101011101000000010
Decoded Message:

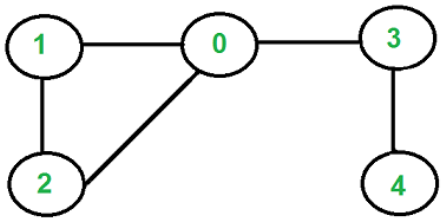
g. Suppose we want to reverse first k elements in a FIFO queue. For example if the input queue is: {10, 20, 30, 40, 50, 60} and k=4. Then output queue must be: {40, 30, 20, 10, 50, 60}. The only functions available for the Queue are:

Enque() – inserts a data item at the end of the queue
Dequeu() – Removes an element from the start of the queue and return the removed element
Isempyt() – Return true if queue is empty and false otherwise
getSize() – returns the number of elements in the queue
How would you reverse the first k elements of the queue using the above functions only? Explain your idea in 3-4 lines
Hint: You may use some other standard data structure to solve this problem

Roll Number:_____ Section:_____

Question 2: **(Marks: 12+3)**

The diameter of a graph is the maximum of shortest paths between any pair of vertices (u, v). The diameter of an unweighted disconnected graph is infinity. Consider the graph on left. The possible vertex pair and their distance (shortest paths) are Dist(0,1) = 1, Dist(0,2) = 1, Dist(0,3)=1, Dist(0,4)=2, Dist(1,2)=1, Dist(1,3)=2, Dist(1,4)=3, Dist(2,3)=2, Dist(2,4)=3, Dist(3,4)=1. Hence, the diameter of following graph is three (3).



Write a C++ function ***ComputeDiameter()*** that takes an unweighted, undirected graph G as parameter and returns the diameter of G.

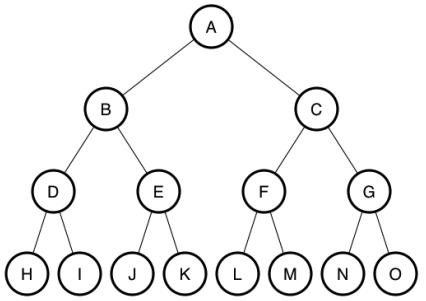
What is the time complexity of ***ComputeDiameter()?***

Roll Number:_____ Section:_____

Question 3: **(Marks: 12+3)**

A node **Z** in a BST is called common ancestor of nodes **X** and **Y** iff both **X** and **Y** exist in the subtree rooted at node **Z**. For example in the tree below B is a common ancestor of both D and K. Write an ***efficient*** C++ function **CommonAncestor()** (member function of class BST) that take the root of a binary tree two integers **X, Y** as parameter and returns all the common ancestors (pointers to the nodes of common ancestors) of **X** and **Y** if both **X** and **Y** exist in the tree. Also note that order of common ancestors must be from nearest ancestor to farthest. For example the common ancestors of L and M are the nodes with data A, C and F. So you should order the ancestors such that F should come first then C and then A.

```
class BSTNode
{
      BSTNode (int );
      BSTNode  * left;
      BSTNode * right;
      int data;
}
```



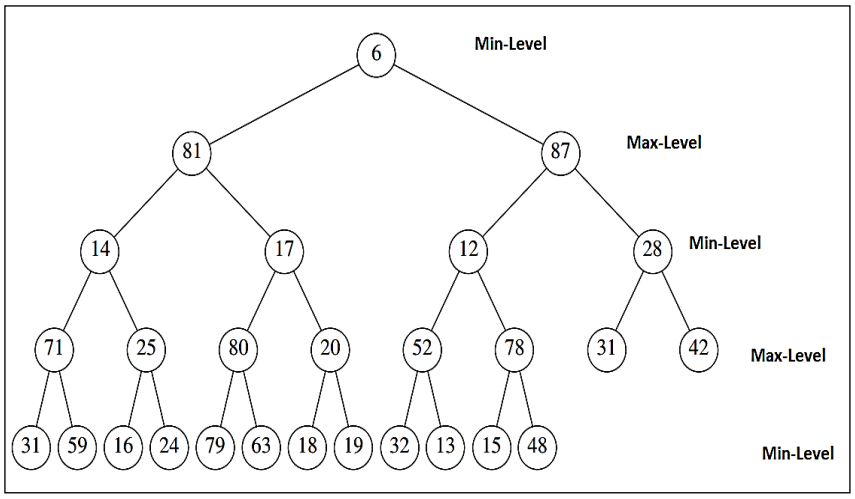
What is the time complexity of your function?

Question 4: **(Marks: 8+7)**

Min-Max Heap is a data-structure that allow us to find both the minimum and the maximum element of the heap in constant time and perform Insert, Delete-Min and Delete-Max operations in O(lgn) time.

The structure is identical to a binary heap, but the heap-order property is that: root is at depth 0, and for any node, *X*, at **even depth**, the element stored at *X* is smaller than parent but larger than the grandparent

For any node, *Y*, at **odd depth**, the element stored at *Y* is larger than the parent but smaller than the grandparent.



It is now obvious that the minimum element in the heap can be found at the root, and that the maximum element is one of the root's two children.

As Min-Max heap is a complete binary tree so we implement it using **arrays.**

```
class MinMaxHeap{
 public:
      void MinMaxHeap(int capacity = 100);
      void deleteMin();
      void PrintSecondMax();
 private:
      int currentSize; // Number of elements in Min-Max heap
      int * data; // The Min-Max heap array
      int capacity;
};
```

a. Write a C++ function deleteMin to extract minimum element from the Min-Max heap in O(lgn). Note the deleteMin should not violate the Min-Max heap property described above.
Hint: the above DeleteMin operation is more or less similar to deletion in simple heaps.

b. Write a function with name PrintSecondMax to print the second maximum value in the Min-Max heap.