FAST National University of Computer & Emerging Sciences, Karachi
Fall-2019 CS-Department
Final Examination
December 10, 2019 Slot: 12:30PM – 3:30PM

Course Code:CS218	Course Name: Data Structures
Instructor Name: Dr. Muhammad Rafi / Ms. Nida Pervaiz	
Student Roll No: 18K-1100	Section No: Gr-1

- Return the question paper.
- Read each question completely before answering it. There are **7 questions and 4 pages.**
- In case of any ambiguity, you may make assumption. But your assumption should not contradict with any statement in the question paper.
- All the answers must be solved according to the sequence given in the question paper.
- Be specific, to the point while coding, logic should be properly commented, and illustrate with diagram where necessary.

Time: 180 minutes. **Max Marks:** 100 points

OOP & Best Practices	
Question No. 1	**[Time: 20 Min] [Marks: 10]**

a. List all important steps one should consider to implement an assignment operator= () for a class with dynamic memory used for objects. [4]
b. Why can a null pointer not be accessed? [2]
c. Consider the following code segment. What do you think will be the output of this? [4]

```
int main(){
    int num = 10;
    int *p = new int;
    *p=num;
    delete p;
    cout << (p) << endl;
    return 0;
}
```

Arrays & Variants	
Question No. 2	**[Time: 20 Min] [Marks: 10]**

a. Given an array of positive integers, write a function that decide whether the array contains all unique (distinct numbers) or not? Suggest any efficient approach to this problem. [*Hint*: Target for less than O(n^2) running time] [10]

Linked List & Variants	
Question No. 3	**[Time: 25 Min] [Marks: 15]**

a. You are given an object of a singly linked list of length **n**, which contains all integers values. You need to write a function that should return the last node of the list. Only the head pointer is available in the singly linked list. Your function should return pointer to the Node<T> detached from the list, after reproducing the valid singly linked list object. [5]

b. You are given two pointers **p** and **q** of type compatible with doubly linked list node type. These two pointers point to two distinct nodes of the doubly linked list. You need to find whether these two nodes are adjacent in the linked list or not. Return TRUE if they are adjacent, otherwise FALSE. [5]

c. You are given a circular singly linked list head pointer as an available argument for a function that count the number of nodes in the list. Write a function that count the number of nodes in the list (circular list). [5]

Stacks, Queues, Priority Queues and Heaps	
Question No. 4	**[Time: 30 Min] [Marks: 20]**

a. One FASTIAN implement a class for ADT Stack using template based implementation similar to the one we did in the course. The class contains all fundamental functions like: Push (), Pop (), Top (), IsEmpty () and Size (). There is a special function Compare () which take a stack element as input and return 1 if the element is greater, or -1 if it is lesser, or 0 if the two elements are equal. Suppose that there are two stacks instances, S1 and S2 which contains sorted elements (minimum on top), you need to combine the two stacks into another stack such that the elements are also in sorted order (minimum on top). You cannot use any other data structures except Stack to solve this problem. [10]

b. A DoubleQueue is an ADT which gives you, two queues using a linear array of size **2n**. The left hand portion is used as first queue and the right hand portion of the array is used as second queue. Using the following class definition, provide the implementation details for the fundamental operation of queue (Functions in Bold script). [10]

```
template <class T>
class DoubleQueue
{
private:
 T *data;
 int qRptr;
 int qLptr;
 unsigned int Max_size;
public:
 DoubleQueue(unsigned int Max_Size = 100, int QPtrL=n-1, int QPtrR=n);
 ~DoubleQueue();
 DoubleQueue( const DoubleQueue & Value);
 const DoubleQueue & operator = ( const DoubleQueue & Value );
 void Enqueue(const T& e, int Qnum); //Qnum can be 0 or 1 indicating left queue &
                                     right queue respectively
 T & Dequeue(int Qnum);
 T & Top(int Qnum);
 isFull(int Qnum);
 IsEmpty(int Qnum);
//rest of declarations
};
```

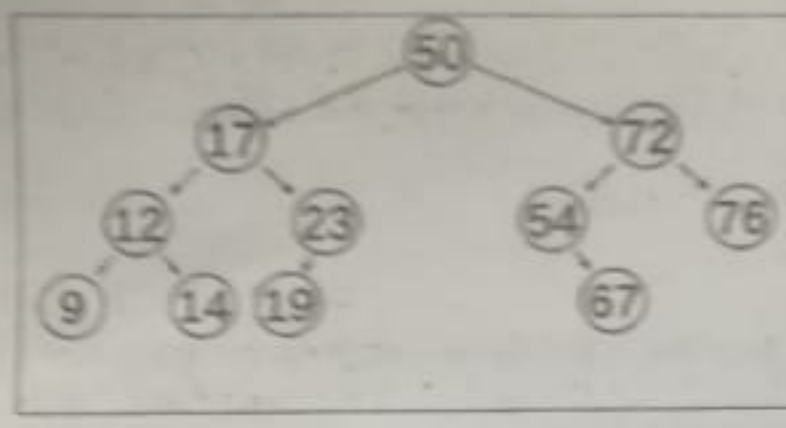
Sorting, Searching & Hashing	
Question No. 5	**[Time: 20 Min] [Marks: 15]**

a. *An IndirectIndexSort is a sorting index produced over a collection. Given an array X [] of n numbers, this sort will produced an array Y[] of indexes of sorted order of these numbers. For example X[] ={13,29,91,34,56} then Y[] ={0,1,3,4,2} Write an efficient implementation of this approach, you cannot move elements of array X. Note that X[] may contains duplicate number and your IndirectIndexSort should be stable in nature.* [10]

b. *Write a function that search and finds the kth smallest element in an unsorted array of n elements. Using any efficient approach that we have discussed during the CS218.* [5]

Trees & Variants	
Question No. 6	**[Time: 30 Min] [Marks: 20]**

a. *Write a function that finds lowest common ancestor of a given pair of node values Key1 and Key2 for a Binary Search Tree given below. Lowest Common Ancestor (LCA) of two nodes N1 and N2 in a tree with key1 and key2 respectively, is the lowest node that has both N1 and N2 as descendants, where we define each node to be a descendant of itself.* [5]



LCA(54,76) is 72;
LCA(23,54) is 50;
LCA(14,23) is 17.

b. Write a function that takes a Binary Tree and return TRUE if it is a Binary Search Tree(BST) or FLASE Otherwise. [5]

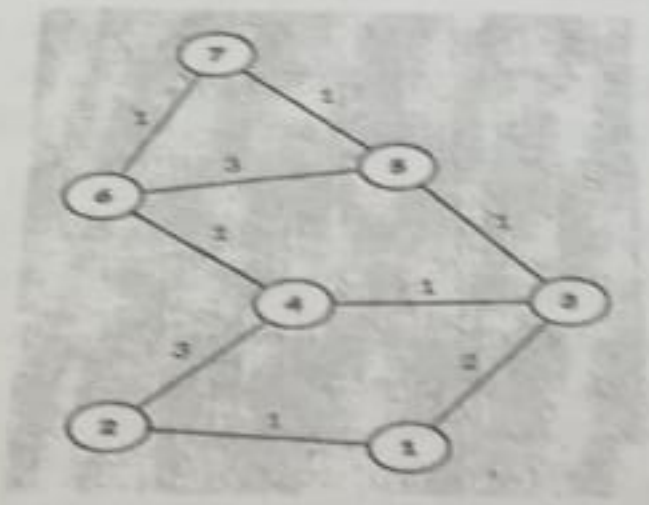
```
bool IsBST(const BTNode<T> *BinaryTree)
```

c. Draw the resultant trees with all intermediate working, if the following sequences of positive integers inserted into an initially empty BST and AVL Tree. [10]
(12, 34, 23,17, 64, 19, 43, 87, 26, 29)

Graphs	
Question No. 7	**[Time: 20 Min] [Marks: 10]**

a. Define Minimum Spanning Tree (MST). Illustrate at least 3 differences between Prim's and Kruskal's algorithm for finding Minimum Spanning Tree. [5]

b. Consider the following graph give below:



Using Prims algorithm find the MST, starting from Vertex 1. [2]
Using Kruskal's algorithm find the MST. [2]
Is it possible to get different MSTs from both the algorithm? [1]