0001

National University of Computer and Emerging Sciences

Lahore Campus

# Data Structures and Algorithms (CS2002)

## Final Exam

**Total Time (Hrs):** 3
**Total Marks:** 80
**Total Questions:** 4

Date: December 28th 2024

**Course Instructor(s)**

Ms. Shazia Haque, Mr. Ahmad Hamza

Roll No _____ Section _____ Student Signature _____

**Do not write below this line**

**Answer all parts of a question together on the answer sheet.**

***PLO 2, C4, CLO #1:* Analyze the efficiency of algorithms and data structures in terms of time and space complexity.** [15 marks]

**Q1:**

i. **Compare** the following time complexity functions and arrange them in ascending order of speed of execution (slowest execution time to fastest time).

A. $f(n) = 3n^2 + n$ $O(n^2)$

B. $g(n) = \log n + 7n$ $O(n)$

C. $h(n) = 5^n + 5$ $O(5^n)$

$O(n) < O(n^2) < O(5^n)$

ii. **Examine** the following code snippet for time complexity expression and express it in terms of Big-Oh

```
temp = 0;
for (i=1; i<=n; i++)
{
    for (j=1; j<=i; ++j)
        temp++;
    for (k=1; k< = n; k=k*2)
        temp++;
}
```

iii. While doing worst case analysis of an algorithm, I found it takes $2^{n+1}$ steps to complete the job, where n is the input size. Can I say it is $O(2^n)$? **Explain**

***PLO3, C5, CLO #2:* Design linear data structures and their associated algorithms.** [20 marks]

**Q2:**

i. **Generate** the postfix expression against the following infix:

A && B || C || !(E>F)

(A+B) * (C-D)

A & B C || E F > ! || &

A & B & C || E F > ! ||

ii. Suppose we are given the following specification for a Stack class.

```
template<class DT>
class Stack
{  public:
        Stack();
        bool isEmpty( );        //  return true if stack is empty
          bool isFull( );     // return true if stack is full
          void push(DT x);// push x on the stack
          DT pop();// returns the top element from Stack
};
```

**Construct** the following non member iterative functions.

a. The following function takes two objects of Stack class as parameters and returns true if they are the same and false otherwise. You are not allowed to use any other data structure other than the two stacks themselves.

```
template<class DT>
bool Compare (Stack<DT> s1, Stack<DT> s2);
```

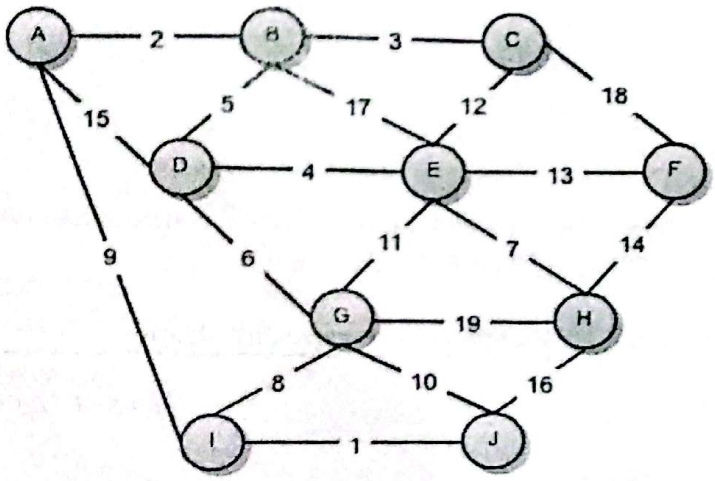
b. A palindrome is a number or a string that remains the same when it is reversed e.g., (civic, level, radar, and 1221) are all palindromes. Your task is to construct the following function that determine whether the string (str) passed in as parameter is a palindrome or not using one object of Stack class given above.

```
bool IsPalindrome(char * str);
```

***PLO 3, C5, CLO #3:* Design tree and graph data structures and their associated algorithms**

**Q3:** [15 marks]

i. **Generate** the output when the graph below is traversed in breadth first manner, breaking ties in alphabetical order and starting from node D



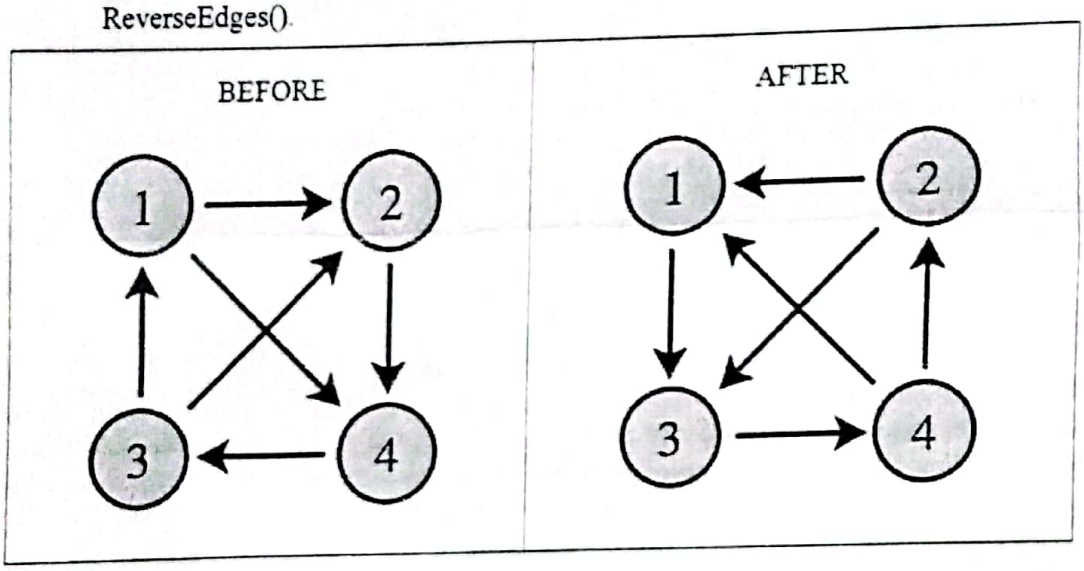
ii. Suppose we are given the following specification for a Graph class.

```
class Graph
{
private:
   int **adjMatrix;
   int numNodes;

public:
   Graph(int nodes);
   void ReverseEdges(); //function to implement

};
```

**Construct** the ReverseEdges() member function that would replace all edges (v, w) with (w, v). The picture below shows a graph before and after a call to ReverseEdges().




***PLO 5, C3, CLO #4:* Apply the best searching/sorting algorithm to solve a problem**

**Q4:** [30 marks]

i. **Apply** quick sort algorithm to sort the array given below in descending order, using the first element of the array as pivot.

| 4 | 1 | 8 | 2 | 9 | 10 | 18 | 12 | 5 | 3 |
|---|---|---|---|---|---|---|---|---|---|

Show the state of the array (on answer sheet) after the first recursive call of the quicksort function.

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|

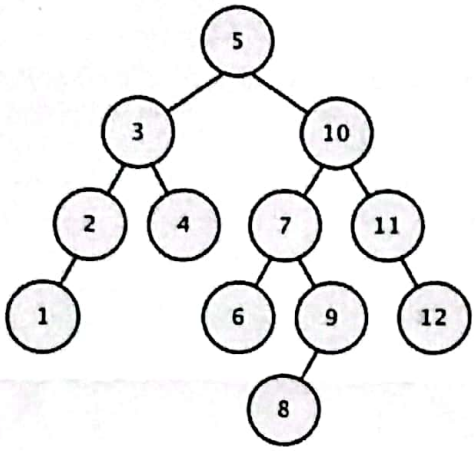
Show the state of the array (on answer sheet) after the second call of the quicksort function.

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|

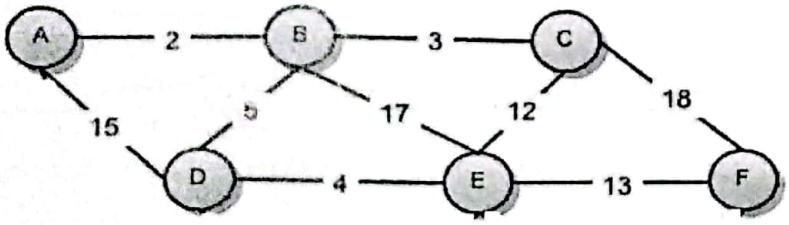
ii. **Using** linear probing as the overflow handling technique insert the following keys in order in a hash table of size 7.

4, 10, 18, 12, 15, 22
What is the search performance of HashTable in terms of Big Oh?

iii. **Draw** the resulting AVL tree after deletion of value 5 from the tree given below.
(a) show the resulting BST after 5 is removed, but *before* any rebalancing takes place. Label each node in the resulting tree with its balance factor.
(b) Now rebalance the tree that results from (a), please **draw** a new tree for each rotation that occurs when rebalancing the AVL Tree (you only need to draw one tree that results from an RL or LR rotation). You do not need to label these trees with balance factors.




iv. **Apply** Dijkstra's shortest path algorithm to the Graph below to determine the shortest path going from node A to F. The state of the various arrays used in the basic implementation of this algorithm after the first iteration is given below. Please show the state of these arrays after every iteration of the algorithm that follow for full credit.




| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| predecessor | A | A | - | A | - | - |
| DistanceFromSource | 0 | 2 | infinite | 15 | infinite | infinite |