NEPAL COLLEGE OF INFORMATION TECHNOLOGY

Level: Bachelor Assessment Year : 2024
Programme: BE (Software Day/Morning) Full Marks: 100
Course: Data Structure and Algorithm Pass Marks: 45
Time : 3hrs.

Attempt all the questions.

1. a). Suppose the given algorithm and compute its total time T(n) for worst and average case. [7]

Algorithm	Cost
for i=1 to n	C1
for j=1 to n	C2
printf("Pokhara University")	C3

b) Convert infix to postfix: K+L-M*N+(O^P). Also for K=3, L=5, M=2, N=2, O=5 and P=2,evaluate the postfix expression. [8]

2. a) Why is recursion needed? Write an algorithm for TOH with an example for having 4 disks [7]

b) Write a C/C++ program for inserting the data in a single linked list. [8]

3. a) Construct an AVL tree by inserting the following data: [8]
31,22,3,15,6,27,9,10,11,12,15.
Also explain why we need B-trees.

b) Construct Huffman tree using Huffman algorithm and also find the Huffman code of the following character frequency. [7]

Note: Given below is character : frequency

a : 1, b : 1, c : 2, d : 3, e : 5, f : 8, g : 13, h : 21

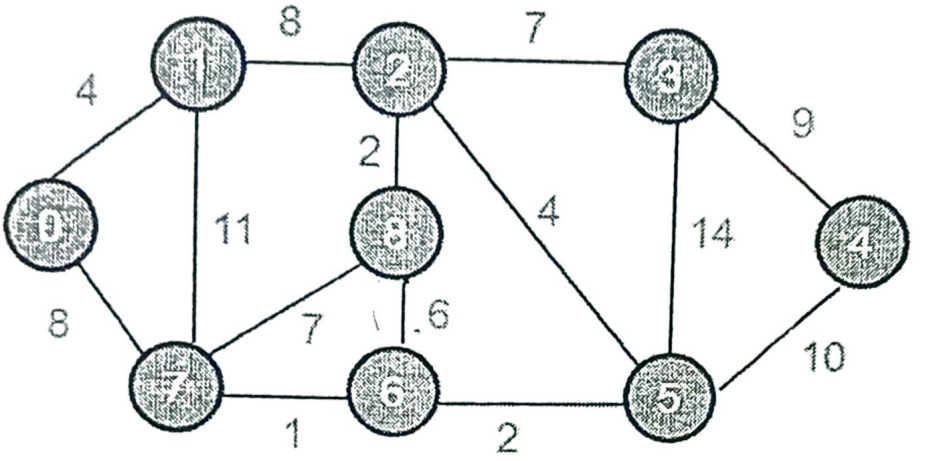
4. a) Differentiate between linear queue and circular queue with an example. [5]

b) Define and implement a simple hash system with a hash function where h(x)= x%7 using c or c++ code if collision occur Use linear probing. [10]

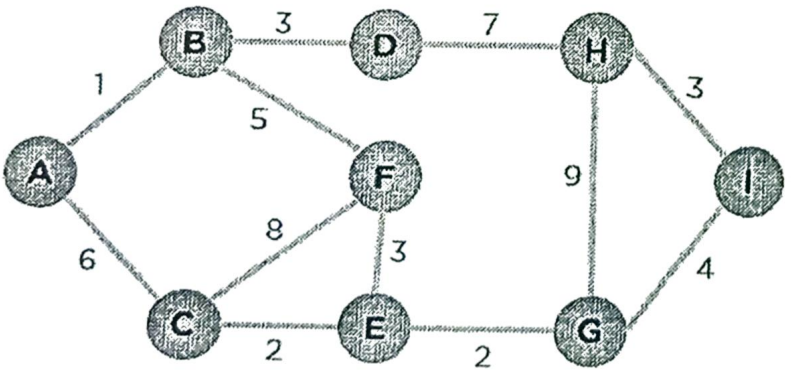
5. a) Sort the following data using quick sort: [7]
21, 43, 51, 32, 20, 35, 8, 12.

5. b) For a given problem, there are different solutions. One solution has the time complexity of O(n) and the other solution has the time complexity of the O(nlogn). Explain which solution would you choose and why? Also contrast the time complexity of selection sort and heap sort. [8]

6. a) What are graph data structures? Why is it needed? Also, find the shortest path using Dijkstra's algorithm where the starting vertex is 1.



b) What is a minimum spanning tree? Find the minimum spanning tree using Prim's Algorithm for the following graph. [7]



Graph G(V, E)

7. Write short notes on: (Any two) [10]
   a) Topological Sort
   b) Backtracking
   c) BFS and DFS