Course Code : CST 213 **KOLP/RW – 19 / 9033**

Third Semester B. E. (Computer Science and Engineering) Examination

DATA STRUCTURES AND PROGRAM DESIGN

Time : 3 Hours] [Max. Marks : 60

Instructions to Candidates :—

(1) All questions carry marks as indicated against them.
(2) Assume suitable data and show algorithm trace wherever necessary.

1. Attempt EITHER **(a)** or **(b)**. Part **(c)** is Compulsory :—

   (a) Define an algorithm. Describe the basic characteristics of an algorithm. Write a C function to compute sum–of–digits of an N–digit number and determine its step count. 5 (CO 1)

   (b) Define a queue. Enlist all allowable operation on a queue. Write C–functions for adding and removing keys to and from an integer queue. Write main( ) to integrate these functions. 5 (CO 1)

   (c) Explain the need for linearization of arrays in computing.

   For a 6 x 7 array Q, and address of Q[0][0] as 1234, determine the address of Q[4][3] when the array is stored using– (1) Row–major order and, (2) Column–major order.

   Propose the indexing function for 2D array for these ordering. 5 (CO 1)

2. Attempt any **Two** :—

   (a) Consider a singly linked linear list. Write C-functions to– (1) Create the list using node insertion at the beginning, (2) Remove the last node from the list, and (3) Print the list. Trace functions with appropriate example. 5 (CO 1)

   (b) Consider a doubly linked linear list. Write C-function or algorithms to reverse the list without creating a new list. Trace your routine appropriately on a list with minimum 7 nodes. 5 (CO 1)

   (c) Consider a singly linked linear list. Write an algorithm to add a node to the list in increasing order of keys (without sorting the list prior to insertion). Test your routine appropriately. 5 (CO 1)

3. (a) What is a binary tree ? Elaborate on full binary tree, complete binary tree and a perfect binary tree. How many nodes are there in a full binary tree ? 5 (CO 2)

(b) Consider a binary tree. Write C-functions to – (1) Count and print intermediate nodes, (2) Check whether two trees are equal, and (3) Destroy the tree. 5 (CO 2)

4. (a) What is hashing ? How it differ from comparison–based search ? Differentiate between linear probing and quadratic probing. Enumerate the problems associated with these approaches. 5 (CO 3)

(b) Assume an initially empty hash table with 13 entries in which the hash function uses the division method. Show the contents of the hash table after the following keys are inserted (in the order listed) using – (1) Linear probing, (2) Open addressing : 765, 431, 96, 142, 579, 226, 903, 388. 5 (CO 3)

5. Attempt EITHER :

(a) How do adjacency list differ from adjacency matrix ? Identify situations when either data structures have advantages over the other.

Devise an algorithm for breadth – first traversal of a graph. For the graph in **Fig. 5 (b)**, show the traversal using BFS (exclude edge weights). Trace the routine for intermediate steps. 10 (CO 2, CO 4)

OR

(b) Elaborate on spanning tree and minimum cost spanning tree. Enlist non–trivial characteristics of a spanning tree.

Compose Kruskal's algorithm for creating MST. For the graph in Fig. 5 (b), construct the minimum cost spanning tree using Kruskal's method. Show intermediate trees at each step.

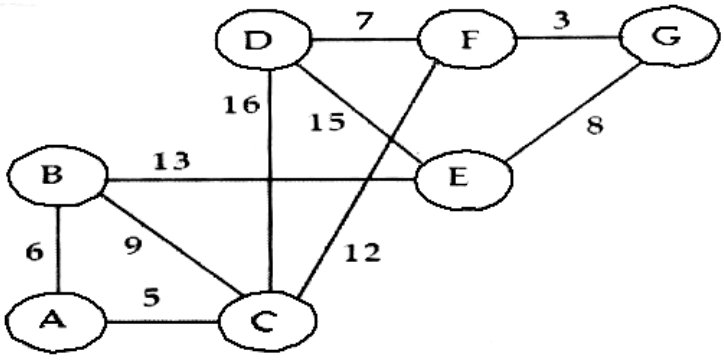

Fig. 5 (b) 10 (CO 2, CO 4)

6. Attempt EITHER :

(a) Propose an algorithm to arrange a large list in increasing order of its key values. The algorithm should ensure the best average time complexity. What other sorting methods have the similar average time bounds ? Trace your algorithm appropriately on a list with minimum 8 keys. 10 (CO 4)

OR

(b) What do you mean by a stable sort ? Identify the applications of stable sort algorithm.

Propose an algorithm to locate a key within a large list in logarithmic time. What are the preconditions for your method ? For a list, L[ ] = {99, 55, 88, 88, 55, 44, 11, 55, 33, 11}, trace your algorithm to locate the key value, 11. 10 (CO 4)