Course Code : CST 213 **CXDW/RW – 18 / 5018**

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

DATA STRUCTURE AND PROGRAM DESIGN

Time : 3 Hours] [Max. Marks : 60

Instructions to Candidates :—

(1) Attempt **ALL** questions.
(2) All questions carry marks as indicated against them.
(3) Due credit will be given to neatness and adequate dimensions.
(4) Assume suitable data and illustrate answers with algorithms and sample execution trails wherever necessary.
(5) **Mobile phones and/or electronic gadgets are prohibited in the examination hall.**
(6) Use of Non – programmable calculator is permitted.

1. Attempt any **Two** of the following :

(a) Explain time and space complexity of an algorithm. For an iterative algorithm to compute sum-of-the-digits of an N-digit number, find its time complexity [use step-count tabular approach]. 5(CO 1)

(b) What do you understand by abstract data type [ADT] ? Enlist characteristics of an ADT. For an array based implementation of a queue, write the C code or algorithmic pseudocodes to realize a Queue. ADT. Show trace of the queue [with array size 4] for operations : Delete (), Insert (10), Insert (20), Insert (40), frontVal(), Delete(), Delete(), Delete() isEmptyQ(), Insert(30). 5(CO 1)

(c) Convert the expression, (A–B) ^ C*(D % E) + F / B to its equivalent Polish form. Give the algorithm to evaluate the Polish expression. Evaluate the obtained Polish expression for A = 6, B = 2, C = 3, D = 8, E = 3, and F = 6, Show the contents (stack frame) at each stage of evaluation. 5(CO 1)

2. Attempt any **Two** of the following :

(a) Consider a singly linked linear list. Construct algorithms or C–functions to:

(1) Add a node at the end of [append] the list and,

(2) Remove last node in the list. The algorithm or function must receive and return a list pointer. For the list, FIRST, show the list after the following sequence of operations–remove(), add(), add(), add(), add(), remove(), add(), add(), remove(), remove(), add(), employing, these algorithms or functions. The key values resembling node data are inserted in the sequence–99, 88, 66, 88, 55, 77, 41, 21, and so on. 5(CO 1)

(b) Develop C functions to implement a linked stack. Write a program to use these functions to simulate a Stack ADT. Your program should allow printing stack contents as required by user. Your solution must gurantee that operations are O(1). 5(CO 1)

(c) Consider a doubly linked linear list. Design algorithms or C functions to:

(1) Add a node to the list.

(2) Remove the last node of the list and,

(3) Add a node at specific position in the list. Trace the algorithms or functions with appropriate example. 5(CO 1)

3. Attempt any **Two** of the following :

(a) Consider a binary tree. Develop an algorithm or C function to:

(1) Find height of a tree,

(2) Count and print parent nodes in a tree, and

(3) Print level-order traversal of a tree. Trace your algorithms or functions on a binary complete binary tree with 12 nodes. 5(CO 2)

(b) Develop an algorithm or C function to delete a node with data contents donoted by KEY from an ordered rooted binary tree [the BST]. Trace your algorithm on a suitable tree for all possible cases. 5(CO 2)

(c) How do a BST and an AVL Tree differ ? Differentiate between a perfect tree, a complete tree and a full tree. Give appropriate example for each of these trees having a minimum of 10 nodes. 5(CO 2)

4. Attempt any **Two** follwing :

(a) Write a C–function or algorithmic pseudocode to list all the keys in a hash table in lexicographic order. Assume that linear probing is used. 5(CO 3)

(b) State the disadvantages of closed hashing. Also suggest remedies in detail with appropriate examples. 5(CO 3)

(c) Given input {4371, 1323, 6173, 4199, 4344, 9679, 1989, 7638, 6743, 1561, 9876} and a hash function H(x) = x% 13, show the resulting:

(i) Separate chaining hash table [when chains are ordered].

(ii) Open addressing hash table using linear probing.

(iii) Open addressing hash table using quadratic probing. 5(CO 3)

5. Attempt any **Two** of the following :

(a) Write an algorithm for implementing depth first search on an undirected graph. For a graph in Fig 5(a), employ DFS showing intermediate DFS trees. 5(CO 2,CO 4)

(b) Define giving appropriate examples–a multigraph, a subgroup and articulation point. Elaborate on the merits and drawbacks of representing a graph using adjacency matrix and adjacency list. 5(CO 2,CO 4)

(c) Write Kruskal's algorithm to construct a minimum cost spanning tree of a weighted graph. For a weighted graph in Fig. 5(c). construct the MST and find its cost. 5(CO 2,CO 4)

6. Attempt any **Two** of the following :

(a) Differentiate between linear search and binary search. When is linear search advantageous ? Write algorithm for implementing a binary search. Trace your algorithm (step-by-step) in locating the key 33 in the list, Keys[ ] = {44, 22, 11, 78, 33, 88, 33, 22, 55]. 5(CO 4)

(b) Let maxHeap represented as an array-Keys[ ] = {99, 88, 33, 55, 77, 44, 22, 11, 66}. Implement heap sort to order the list. Show intermediate heaps at each stage of the sorting process. 5(CO 4)

(c) Write an algorithm for quick sort [assume pivot as last element of an array]. Show pass-by-pass execution of quick sort for ascending order sequencing of the list, Keys [ ] = {20, 88, 71, 15, 35, 52, 67, 48}

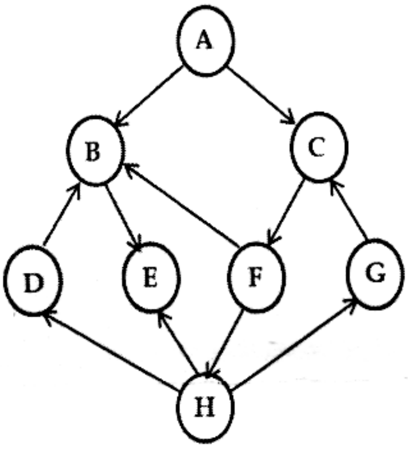

Fig. 5(a)

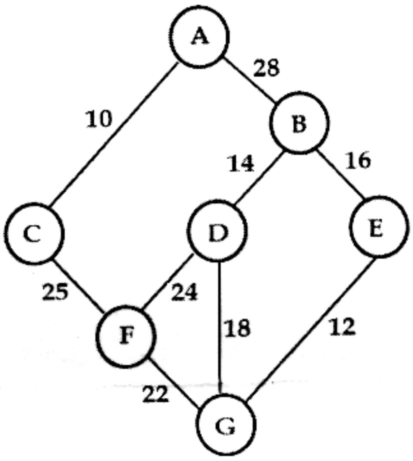

Fig. 5(c)

5(CO 4)