Reg. No.										

MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
A Constituent Institution of Manipal University

III SEMESTER B.TECH. (COMPUTER SCIENCE AND ENGINEERING)
END SEMESTER EXAMINATIONS, NOV/DEC 2016
SUBJECT: **DATA STRUCTURES (CSE 2103)**
REVISED CREDIT SYSTEM
(28/11/2016)

Time: 3 Hours MAX. MARKS: 50

Instructions to Candidates:

- Answer **ALL** the questions.
- Missing data may be suitably assumed. You may draw pictorial representations and algorithms to justify your code.

1A. Write a recursive program that finds quotient when *a* is divided by *b* where a and b both are positive integers. Draw the call tree for invoking the call *quotient (8, 2).* With an example, identify conditions under which recursion is to be avoided? 3

1B. Design functions to implement the following operations in dynamic memory allocation.
i) Return the pointer after allocating memory for M pointers and N data locations
ii) Fill each block of N data locations with values 0, 1…N-1
iii) Free the allocated memory 3

1C. An array A has to be dynamically allocated to hold M pointers. Each location A[i] of the array needs to hold the data of N locations. If we dynamically allocate memory, the memory locations need not be continuous. Write a function to allocate memory such that memory locations are continuous. Also, write a function to free the memory. 4

2A. Write an algorithm that checks whether a sequence of integers read from the keyboard form a palindrome. Return zero, if the sequence does not form a palindrome. Write a C function for the algorithm and trace its output with sequence {44, 12, 64, 32, 32, 64, 12, 44}. Note that you may use stacks and/or queues in an array implementation but cannot use any other structures such as arrays, trees or linked lists. The input sequence is not available as an array but should be read from the keyboard and cannot be stored in a temporary array. All the functions to implement stack/queue functionalities need not be shown. Provide type definition for your data structure. 3

2B. Complete the following table by translating the given infix form. Also draw the expression tree in each case.

Infix expression	Postfix expression	Prefix expression	Expression tree
P*Q+R/S			
P*(Q+R)/S			
P*(Q+R/S)			

3

2C. Given an infix expression, write an algorithm to convert it to prefix. Use this algorithm to trace the infix expression ((a +(b –c) * d) $ e + f) in the form of a table. What content is finally remained in the stack on encountering end of input which is then attached to the postfix string? 4

3A. For each of the following cases, write the relevant data structure. You may assume data structures such as Unsorted array, Stack, Queue, Singly linked list, Doubly Linked List, Circular singly linked list, Circular doubly linked list and Tree. Give reason for each.
i)A book store requires that customers who come first will be served first
ii) A student list must be maintained so that any element can be accessed randomly
iii) A task needs to remember the operations it performed in opposite order
iv) The size of a list is unknown. The entries need to be entered as they come in. Entries must be deleted when they are no longer needed
v) A list must be maintained so that elements can be added to the beginning or ending — 3

3B. Write C functions to implement stack using singly linked list without a header node and the list is not circular. The prototype of stack operations are *void push (int data, struct node ** s)* and *int pop (struct node ** s)* — 3

3C. You are given two sorted stacks A and B as singly linked lists without a header node and the list is not circular whose prototypes are shown in Question 3B. The node pointers S1 and S2 point to the top of each stack. In both stacks, minimum element is on top of the stack. Create a new sorted stack C with minimum element on top using a C function. You can only make use of stack data structure shown above and no other data structure including arrays is allowed. Make use of push and pop functions defined in Question 3B and any other required functions need to be shown completely. — 4

4A. What are the two ways of representing trees in memory? Which one is preferred and why? Write an iterative algorithm to find the height of a binary tree. Justify your answer with a full binary tree of height 3. Consider height as the number of edges on longest path from the root to a leaf node. — 4

4B. For an infix expression 3 - 2 + 4 * 5 / 1, pictorially show the expression tree. Derive a postfix expression from the expression tree. Write a recursive function *float Compute (struct tree* root)* to evaluate the postfix expression whose root is passed as an argument. Show the trace of this function on the expression tree in the form of a Call Tree clearly showing all the intermediate values. — 3

4C. You are given a full binary tree BT of height h wherein h is computed by taking the level of the root as 0. It is required to transform the binary tree BT to a threaded binary tree ThBT with left and right threads scheme. In terms of height of the tree, find out general expressions for the following i) CountLink which gives total number of links of ThBT and ii) CountThreads which gives the total number of threads that arise in ThBT. Write the code snippet to get inorder successor of a non-leaf node given a pointer *peep* pointing to a given node of ThBT. — 3

5A. Given a static element set, arrive at an expression for finding the total cost of a BST that includes both successful and unsuccessful searches. Use this expression to define an Optimal BST. Assume the root is at level 0. — 3

5B. It is required to sort n elements using insertion sort to sort in ascending order. For the elements {89, 45, 68, 90, 29, 34, 17}, mark the count of iteration, and the element being inserted till all the elements are sorted. — 2

5C. What restrictions apply to the following data structure? Explain with an example and their corresponding properties
i) Graphs
ii) Red black trees denoting black nodes by double circle and red nodes by single circle
iii) M-way search trees — 1+2+2