Reg. No.										

MANIPAL INSTITUTE OF TECHNOLOGY

MANIPAL

A Constituent Institution of Manipal University

III SEMESTER B.TECH. (COMPUTER SCIENCE & ENGINEERING)

END SEMESTER EXAMINATIONS, NOVEMBER 2017

SUBJECT: DATA STRUCTURES [CSE 2103]

REVISED CREDIT SYSTEM
(21/11/2017)

Time: 3 Hours MAX. MARKS: 50

Instructions to Candidates:

- Answer **ALL** the questions.
- All the programs should be written in 'C' language.
- Missing data may be suitable assumed.

1A. Write a recursive function **sumofSquares()** that takes a positive integer *N* as parameter and returns the sum of squares of first '*N*' integers. Write a main function to test your function. Trace your function for the input value of *N* = 5, showing the sequence of calls made and value of the parameter at each call, in the form of a call tree. **4M**

1B. Write a single C statement for each of the following:

(i) Allocate dynamic memory for an array of 200 integers and initialize them to zero and assign it to an integer pointer, *intPtr*.

(ii) Define a self-referential structure ***FRACTION*** consisting of following data members - numerator and denominator (both of type int).

(iii) Define a user-defined enumerated data type ***progLang***, the variables of which can assume the values PASCAL, C, CPLUSPLUS, JAVA, CSHARP and PYTHON. **3M**

1C. What does the following declarations mean?

(i) const int **a*;

(ii) int (**aFive*)[5];

(iii) int (**p*)(char); **3M**

2A. Write a function **infixtoPostfix()** to covert a given infix expression into its equivalent postfix form. Assuming that the functions for stack operations are readily available, write all the other necessary functions. **4M**

2B. Given two stacks implemented using static arrays and the operations on them having the following prototype:

void **Push1**(int); void **Push2**(int); int **Pop1**(); int **Pop2**();
int **IsEmpty1**(); int **IsFull1**();int **IsEmpty2**(); int **IsFull2**();

Write functions to implement queue operations using the above two stacks. **3M**

2C. Consider a networking application where a server has to respond for requests from multiple clients. Each client request, demands some amount of server time. Assume that the server receives the requests from various clients (C1, C2, C3 and so on) as they arrive and first responds to the client which demands least amount of time. Device a suitable data structure for the server to *receive* requests and *respond* to the client requests. A request consists of *client id* and *amount of server time* required. **3M**

3A. Write the functions to implement the following operations on a Singly Linked List (SLL) - to insert an item at front and to reverse the list. Consider only the following declarations for the functions, where ***Nodeptr*** is an user-defined pointer data type to struct *node* consisting of int *data* and pointer to next node, *next*. ***Nodeptr*** *first* points to the first node of the list.

(i) ***Nodeptr*** **InsertFront**(***Nodeptr*** *first*, int *item*);

(ii) ***Nodeptr*** **Reverse**(***Nodeptr*** *first*); **4M**

3B. Given two Circular Doubly Linked Lists (with Header Nodes), *A* and *B* representing two long positive integers, write a function to add them and store the result in another Circular Doubly Linked List with Header Node. The function for addition should have the following prototype: ***Nodeptr*** **AddLongInteger**(***Nodeptr*** *A*, ***Nodeptr*** *B*);
where ***Nodeptr*** is an user-defined pointer data type to struct *node* consisting of int *data* and 2 pointers - pointer to next node, *rlink* and pointer to previous node, *llink*. Write all the necessary functions required. **4M**

3C. What is a doubly linked list? How it is advantageous over singly linked list? **2M**

4A. Write recursive functions to create an exact copy of a given binary tree and to traverse the tree in inorder. The functions should have the following declarations only, where ***Nodeptr*** is an user-defined pointer data type to struct *node* consisting of int *data* and 2 pointers - pointer to leftchild, *lchild* and pointer to rchild, *rchild*.

(i) ***Nodeptr*** **copyBinaryTree**(***Nodeptr*** *root*);

(ii) void **inorder**(***Nodeptr*** *root*); **4M**

4B. Write an iterative function to create an expression tree for given postfix expression. Also, write a recursive function to evaluate the expression tree created. The functions should have the following prototype:

Nodeptr **createExpressionTree**(char *postfix*[]);

int **evalExpressionTree**(***Nodeptr*** *root*);

where ***Nodeptr*** is an user-defined pointer data type to struct *node* consisting of char *data* and 2 pointers - pointer to leftchild, *lchild* and pointer to rchild, *rchild*. **4M**

4C. Construct the Binary Search Tree (BST) for the list of letters

J, R, D, G, T, E, M, H, P, A, F, Q

by successively inserting them in the given sequence into an empty BST. **2M**

5A. Write a pseudocode (or function) to sort a given list of integers using Radix Sort. Given the following list of numbers, sort them using Radix Sort showing all the intermediate steps:

170, 45, 75, 90, 802, 24, 2, 66, 77, 55, 74, 48 **4M**

5B. Given a graph, G = (V, E) with n vertices, discuss the different graph representation schemes and compare them. Given the graph representing air routes between various major cities (**Fig. 5B),** represent the graph using above discussed schemes:

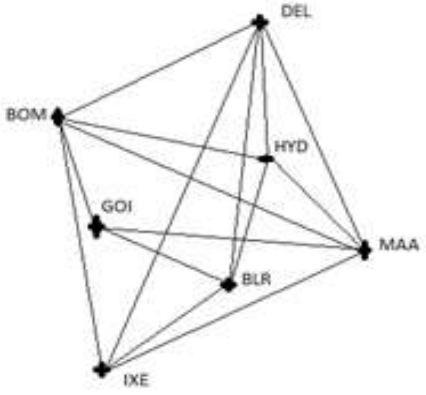

Fig. 5B **4M**

5C. What is a B-Tree? Explain with an example. **2M**