14/11/18

# Sardar Patel Institute of Technology

Bhavan's Campus, Munshi Nagar, Andheri (West), Mumbai-400058-India
(Autonomous College Affiliated to University of Mumbai)

## End Semester Examination

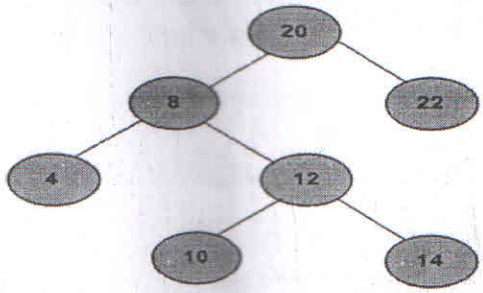
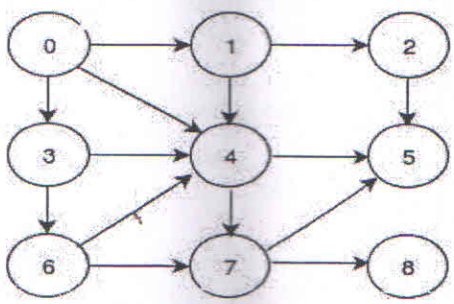
November/December 2018

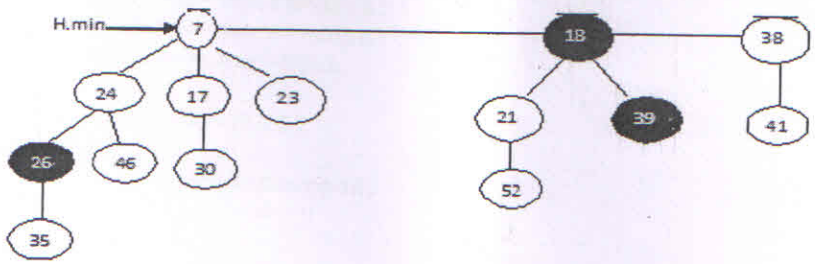
**Max. Marks: 60** | **Duration: 3 hrs**
**Class: S.E.** | **Semester: III**
**Course Code: CE31/IT31** | **Branch: COMP/IT**
**Name of the Course: Advanced Data Structures**

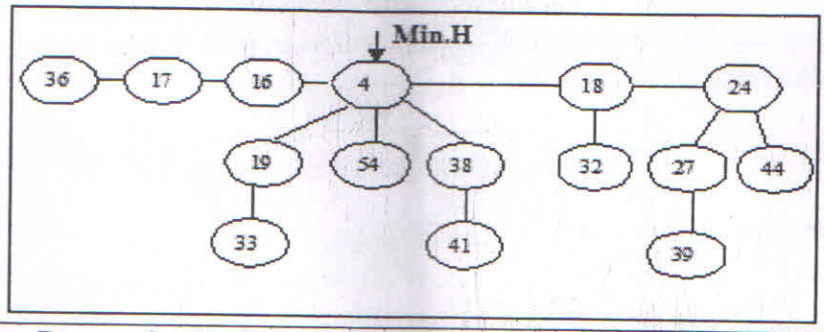
**Instructions:**

**(1) All Questions are Compulsory**
**(2) Draw neat diagrams**
**(3) Assume suitable data if necessary**

| Question No. | Question | Max. Marks | CO |
|---|---|---|---|
| *Q1 (a)* | Write a function to perform following Doubly linked list operation<br>1- create a sorted doubly linked list, at the time of creation of sorted doubly linked list assume duplicates<br>Example: input1: 5,2,2,8,1<br>Output1: 1->2->2->5->8<br>Input2: 9,3,4,4,11<br>Output2: 3->4->4->9->11 | *03* | *CO1* |
| | 2- Merge the 2 sorted doubly linked lists by removing duplicates in it<br>Example: Output of above two lists merged by removing duplicates is: Input1: 1->2->2->5->8<br>Input 2: 3->4->4->9->11<br>Output: 1->2->3->->4->5->8->9->11 | *03* | |
| | **OR** | | |
| | write a function to perform the following operations on polynomial equations using Linked List:<br>1- Create Linked representation of Polynomial equations<br>2- Perform addition of two polynomial equations. Consider sign of coefficient while performing operation<br>3- Display the resulting Polynomial equation | *02*<br>*03*<br>*01* | |
| *Q1 (b)* | Write a program to perform the following operations of Binary Search Tree:<br>1- Create a binary search tree<br>2- Display k'th Largest Element in BST, when modification to BST is not allowed | *06* | *CO2* |

| | | | |
|---|---|---|---|
| | For Example:<br>  <br><br>Given a Binary Search Tree (BST) and a positive integer k, find the k'th largest element in the Binary Search Tree.<br>For example, in the following BST, if k = 3, then output should be 14, and if k = 5, then output should be 10 | | |
| *Q2 (a)* | Consider the Graph given below. Assume that the graph represents the daily flights between different cities and we want to fly from city '0' to '8' with minimum stops.<br>1- Suggest a suitable graph traversal algorithm to complete a journey from city 0 to 8 with minimum stops<br>2- Apply the suggested graph traversal algorithm on the given graph with city '0' as starting node and assume the procedure considers increasing order of cities and also Show the status of the data structure used at every step.<br>3- Draw the resulting minimum path tree starting from city '0'<br>4- state the path to fly from city '0' to '8' with minimum stops<br>  <br><br>**OR**<br>Perform the following operations on an AVL tree with the following values<br>15 \| 20 \| 24 \| 10 \| 13 \| 7 \| 30 \| 36 \| 25<br>1- Build an AVL tree and State the kind of rotation applicable at each insertion and Show the updated AVL Tree after each insertion of keys.<br>2- Delete(24) , Delete(20) State the kind of rotation applicable after each deletion and Show the updated AVL Tree after deletion of keys | *01*<br>*05*<br>*01*<br>*01*<br>*05*<br>*03* | *CO2* |

| | | | |
|---|---|---|---|
| ***Q2 (b)*** | 1- Perform insertion of given keys into hash table of size 13, use Linear probing for resolving the collision if occurred. After successfully allocating all the keys show the final hash table.<br><br>18 \| 41 \| 22 \| 44 \| 59 \| 32 \| 31 \| 73<br><br>2- Perform deletion of the following keys<br>Delete (31), Delete (32). Show the steps of deletion of keys and final updated table after deletion.<br>3- On hash table after successful deletion Delete (31), Delete (32) perform Search operation of following keys and show the steps of searching the keys Search(73), Search(31) | ***04***<br><br>***02***<br><br>***02*** | ***CO5*** |
| ***Q3 (a)*** | Write a Program to sort the array elements using heap sort with Buid_Max _Heap() and Max_Heapify() as sub functions.<br><br>**OR**<br><br>Perform Decrease Key operation on the given Fibonacci heap; consider darkened nodes are marked nodes. Decrease key of node with key 46 to 15 and of node with key 35 to 5. Show the updated Fibonacci heap at every step and write the applied rule/ justification at each stage.<br><br>  | ***06*** | ***CO4*** |
| ***Q3 (b)*** | Construct a Binary Tree for the inorder and Post order traversal sequence given below. Show updated tree at every steps<br>Inorder: INFORMATION<br>Post order: INOFMAINOTR | ***04*** | ***CO2*** |
| ***Q4*** | Extract the Minimum node from the given Fibonacci heap and update the heap. Show updated heap at each step | ***08*** | ***CO4*** |




| | | | |
|---|---|---|---|
| *Q5 (a)* | Create a B tree of order 5 by inserting the following elements<br>3 14 7 1 8 5 11 17 13 6 23 12 20 26 4 16 18 24 25 19 | *08* | *CO2* |
| *Q5 (b)* | Perform following KD tree operations on the given 2D points and show the final KD tree after each operation.<br>1- Perform insertion of following points<br>Insert: (3, 6), (17, 15), (13, 15), (6, 12), (9, 1), (2, 7), (10, 19).<br>Show the updated KD-tree after insertion of each point and also show the final partioned state space<br>2- Perform deletion of following points from above created KD tree and show the updated KD tree after deletion of each poin<br>Delete: (3, 6), (13, 15) | *04*<br><br>*02* | *CO3* |