Computer Engineering Department, SVNIT, Surat
Exam: End Semester, Dec -2020
Sub: Data Structures (CS210)
Date : 11th December 2020

Section B
Marks 25

Writing Time: 4:00 to 5:15 PM **Uploading Time: 5:15 to 5:30PM**

structions:

1. Copy from any book or online material or other answer-book is strictly prohibited. No marks for copied work
2. Use your own examples for explaining the theories
3. Submit Section A before starting the Section B
4. Timely uploading of each section is mandatory, late receipt will not be considered in any condition/situation (Each section to be uploaded separately)
5. Answers to be hand written on Answer-sheet like pages
6. Answers must be uploaded in sequential order of the questions
7. It is compulsory to mention **Stu Code :Question number : PageNumber** On **top right of each page**

Q.4 Construct a data structure "m-min-heap" which is similar to an ordinary binary min-heap except for the number of child nodes. In your constructed "m-min-heap", one node should be able to point m child nodes instead of two. In your answer, discuss how you will represent this "m-min-heap" using an array. In other words, how an internal node will refer to its child nodes in an array. [8]
For example, in an ordinary binary min-heap (stored from A[1] to A[n]), an internal node A[i] uses A[2*i] and A[2*i + 1] to refer to its child nodes.

Write algorithm for implementing the following operations.
(a) Insert a element into "m-min-heap"
(b) Delete an element from "m-min-heap"
(c) Search an existing element in "m-min-heap"

Q.5 Create a dynamic memory allocation system based on two linked lists, i.e., ALLOCATE and FREE. Initially, the FREE list should contain 50 nodes, where each node can store an integer identifier and a 10-character student name. If the name of the student is long then the next free nodes should be used to store the remaining part. A node that contains the remaining part of a student name will set its own integer identifier to 0. [9]

Write an algorithm to perform various operation like Delete existing student, Insert new student information or search and view particular student information.

Insert:

- The FREE list (if a sufficient number of nodes are available) will provide nodes to store the data of the new student.

- After insertion, the ALLOCATE list will point to these nodes and the FREE list will be updated to point to the remaining available nodes.
- In a case when the number of nodes is insufficient, your program will print a message.

Search:

- Fo searching a student, the user can input a student id (which must be a positive integer value) or a student name.
- Searching of the existing student will be performed in the ALLOCATE list.

Delete:

- For the delete operation, nodes corresponding to the student will be deleted from the ALLOCATE list.
- After deletion, the FREE list will point to these nodes.

Show FREE and ALLOCATE list aftere each of the following process

1. Insert 10, Sundar
2. Inserrt 23, RajanKrisnam
3. Delete Sundar

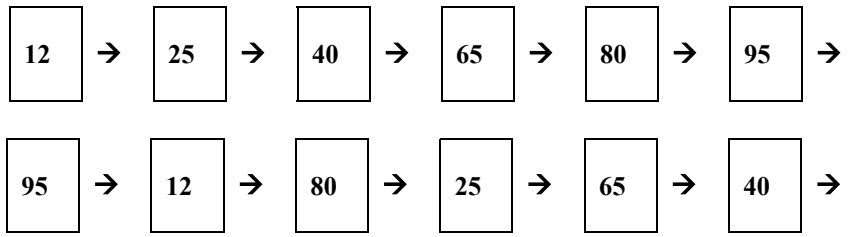
OR

Q.5 A You are given a Singly link list E1→E2→E3→……..En-1→En. [6]

Write an algorithm to regenerate the link list by merging two list as

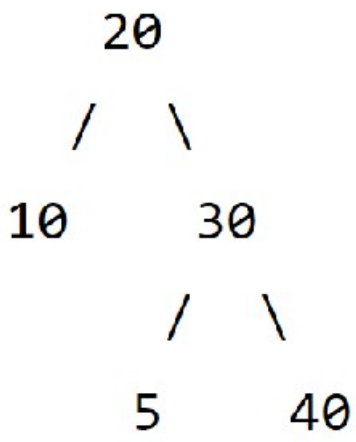
En→E1→En-1→E2→……

Dry Run your algorithm and show the intermediate result after each step of your algorithm



Q.5 B You are given Binary tree as an input. Write an algorithm to check weather the given Binary tree is BST or not? [3]

Test your algorithm for the following input tree.



Q.6 Suppose you have a binary tree. [Use linked or the sequential storage for the tree] [8]

Write Algorithms for following Functions:

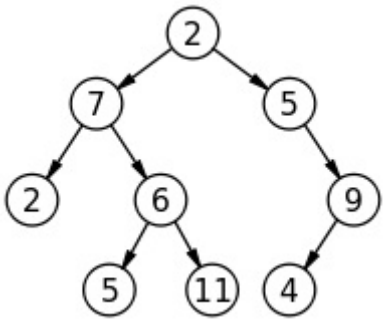
Ances(X) : List a set of ancestor nodes of X (containing node X, X's parent, X's grandparent, etc)

U_Ances(X, Y) : List a set containing all nodes which are either the part of Ances(X) or Ances(Y).

Sum_U_Ancess(X, Y) : calculates the sum of the values of all nodes present in set U_Ances(X, Y).

Single() : List a set of nodes without any siblings

Example :



Ances(6) = 2, 7
Anses(9) = 2, 5
U_Ances (6,9) = 2,7,5
Single() = 9, 4

Simulate your Algorithms for the given tree.