DATA STRUCTURES
ASSIGNMENT - VI
(FOR THE EIGHTH, THE NINTH, AND THE TENTH LAB SESSIONS)

Assignments to be completed during lab sessions

1. Given the inorder and the postorder traversal of a binary tree, write a function to generate the tree.
2. Given the inorder and the preorder traversal of a binary tree, write a function to generate the tree.
3. Write a function to print the key values in a binary tree with inorder traversal.
4. Write a function to print the key values in a binary tree with preorder traversal.
5. Write a function to print the key values in a binary tree with postorder traversal.
6. Write a function to find the node with the minimum key value in a given binary tree.
7. Write a function to find the node with the maximum key value in a given binary tree.
8. Write a function to find the node with a given key value in a given binary tree.
9. Write a function to find the height of a given node in a binary tree.
10. Write a function to create the mirror of a binary tree.
11. Write a function to make a clone of a given binary tree.
12. Write a function to compare the key values of twp trees. It should return 1 if they are the same and 0 otherwise.
13. Write a function to delete a given binary tree.
14. Write a function to find the inorder predecessor of a given node in a binary tree.
15. Write a function to find the inorder successor of a given node in a binary tree.
16. Write a function to count the number of leaf nodes in a given tree.
17. Write a function to count the number of nonleaf nodes in a given tree.

18. Write a function to print the key values of the leaf nodes in a given tree following inorder traversal.

19. Write a function to print the key values of the nonleaf nodes in a given tree following inorder traversal.

20. Write a function to insert a new node corresponding to a given key in a binary search tree using recursion.

21. Write a function to insert a new node corresponding to a given key in a binary search tree without using recursion.

22. Write a function to find the node with the minimum key value in a given binary search tree.

23. Write a function to find the node with the maximum key value in a given binary search tree.

24. Write a function to find the node with a given key value in a given binary search tree.

25. Write a function to delete a node corresponding to a given key from a binary search tree.

26. Write a function to insert a node in an AVL tree.

27. Write a function to delete a node from an AVL tree.

Additional assignments

1. Write a function to print the key values in a binary tree with inorder traversal. You cannot use recursion.

2. Write a function to print the key values in a binary tree with preorder traversal. You cannot use recursion.

3. Write a function to print the key values in a binary tree with postorder traversal. You cannot use recursion.

4. Given the postorder traversal of a binary search tree, write a function to generate the tree.

5. Given the preorder traversal of a binary search tree, write a function to generate the tree.

6. Consider a tree, such that, its internal nodes are from a set of binary operators $\mathcal{O} = \{*, /, +, -\}$ and the leaf nodes are either a constant (real value) or a variable from a set of variables $\mathcal{X} = \{\texttt{x0}, \texttt{x1}, \texttt{x2}, \cdots, \texttt{xn}\}$. Here, `n` is the number of variables and a variable name has the format: `xi`, where `i` is a positive integer. An example of a string (array of characters) of a preorder traversal of such a tree is

   $$- \ * \ \texttt{x4} \ 3 \ / \ \texttt{x0} \ \texttt{x2}$$

   For the input $\mathbf{x} = (6.0, 1.1, 2.0, 3.0, 1.0, 9, 0)^T$, the tree (expression) is evaluated as:

   $$(\texttt{x4} \ * \ \texttt{3}) \ - \ (\texttt{x0} \ / \ \texttt{x2}) = (\texttt{1.0} \ * \ \texttt{3}) \ - \ (\texttt{6.0} \ / \ \texttt{2.0}) = \texttt{0.0}$$

   Here, in the string, each token (operator or operand) is separated by whitespace.

   (a) Write a function to construct a binary tree from its preorder traversal stored in a string.

   (b) Write a function to evaluate the expression stored in a tree for a given set of values of the variables.

7. Consider the following structure and perform the following tasks on it.

```
struct student_t {
    int   roll;   /* The roll no of the student.           */
    char *name;   /* A pointer to the name of the student. */
    int   marks;  /* The marks of the student.             */
};
```

   (a) Write a function to insert objects of type `struct student_t` in binary search trees considering either `roll` as the key-value or `marks` as the key value. The same function should work for both scenarios. In other words, using the same function, one can construct a binary search tree that considers `roll` as the key-value or `marks` as the key value. You need to add the objects in the nodes as their satellite data. The satellite data cannot be the part of a node; instead, a node should have a `void`/ `char` pointer that points to the satellite data.

   (b) Given a key-value, write a function to find its associated satellite data in a binary search tree.

   (c) Given a key-value, write a function to delete a node from a binary search tree. You need to provide the satellite data to the caller function, either by returning a pointer to it or by some other means.

   (d) Write a function to visit each node in a binary search tree following inorder traversal and perform an operation on the satellite data associated with each node. The operation might be any generic operation. For instance, it could add some grace marks, say 1 or 2, only if they need one or two marks to get the next grade. Therefore, a student who got 79 or 68 will have increased marks of 80 or 70, respectively. Another example would be to change the roll number of each student by adding 1900000 to it so that a student with roll number 1234 will have the modified roll number 1901234.