National Institute of Technology Calicut
Department of Computer Science and Engineering
Third Semester B. Tech.(CSE)
CS2092D Programming Laboratory
Assignment #9

Submission deadline (on or before):

- 18.11.2020, 10:00 PM

Policies for Submission and Evaluation:

- You must submit your assignment in the Eduserver course page, on or before the submission deadline.
- Ensure that your programs will compile and execute without errors in the Linux platform.
- During the evaluation, failure to execute programs without compilation errors may lead to zero marks for that evaluation.
- Detection of ANY malpractice related to the lab course can lead to awarding an F grade in the course.

Naming Conventions for Submission

- Submit a single ZIP (.zip) file (do not submit in any other archived formats like .rar, .tar, .gz). The name of this file must be

  `ASSG<NUMBER>_<ROLLNO>_<FIRST-NAME>.zip`

  (Example: $ASSG1\_BxxyyyyCS\_LAXMAN.zip$). DO NOT add any other files (like temporary files, input files, etc.) except your source code, into the zip archive.

- The source codes must be named as

  `ASSG<NUMBER>_<ROLLNO>_<FIRST-NAME>_<PROGRAM-NUMBER>.c`

  (For example: $ASSG1\_BxxyyyyCS\_LAXMAN\_1.c$). If you do not conform to the above naming conventions, your submission might not be recognized by our automated tools, and hence will lead to a score of 0 marks for the submission. So, make sure that you follow the naming conventions.

Standard of Conduct

- Violation of academic integrity will be severely penalized. Each student is expected to adhere to high standards of ethical conduct, especially those related to cheating and plagiarism. Any submitted work MUST BE an individual effort. Any academic dishonesty will result in zero marks in the corresponding exam or evaluation and will be reported to the department council for record keeping and for permission to assign F grade in the course. The department policy on academic integrity can be found at: http://cse.nitc.ac.in/sites/default/files/Academic-Integrity_new.pdf.

General Instructions

- Programs should be written in C language and compiled using C compiler in Linux platform. **Submit the programs for question 1 through the submission link in Eduserver. You should complete the program for question 2 before the next lab session. During the evaluation we may be asking you to modify any of these two programs**.

QUESTIONS

1. The BINARY SEARCH TREE (BST) data structure supports many of the dynamic-set operations. A BST is organized as a binary tree in which each node is an object that contains a *key* value. In addition to a *key* and satellite data, each node contains attributes *left*, *right*, and p that point to the nodes corresponding to its left child, its right child, and its parent, respectively. If a child or the parent is missing, the appropriate attribute contains the value NIL. The root node is the only node in the tree whose parent is NIL. The keys in a binary search tree are always stored in such a way as to satisfy the ***binary-search-tree*** property:

   - Let x be a node in a binary search tree. If y is a node in the left subtree of x, then $y.key \leq x.key$. If y is a node in the right subtree of x, then $y.key \geq x.key$.

   Write a program to create a BINARY SEARCH TREE T and perform the operations *insertion*, *deletion*, *search* and *traversals*(inorder, preorder and postorder) on T. Input should be read from console and output should be shown in console. Your program should include the following functions.

   - MAIN() - creates the Binary Search Tree T with T as the root node (which is NIL initially) and repeatedly reads a character 'a', 'd', 's', 'i', 'p', 't', or 'e' from the console and calls the sub-functions appropriately until character 'e' is entered.
   - CREATENODE(k) creates a new node with *key* value k and returns a pointer to the new node. All the pointer attributes of the new node are set to NIL.
   - INSERT(T, x) - inserts the node x into the BST T.
     Note: The caller of this function is assumed to create the node x using the CREATENODE() function.
   - DELETE(T, x) - deletes the node x from the BST T.
     Note: The caller of this function is assumed to invoke SEARCH() function to locate the node x.
   - SEARCH(T, k) - searches for a node with key k in T, and returns a pointer to a node with key k if one exists; otherwise, it returns NIL.
   - INORDER(T) - performs recursive inorder traversal of the BST T and prints the data in the nodes of T in inorder.
   - PREORDER(T) performs recursive preorder traversal of the BST T and prints the data in the nodes of T in preorder.
   - POSTORDER(T) performs recursive postorder traversal of the BST T and prints the data in the nodes of T in postorder.

   Input format:

   - Each line contains a character from 'a', 'd', 's', 'i', 'p', 't', or 'e' followed by at most one integer. The integers, if given, are in the range $[-10^6, 10^6]$.
   - Character 'a' is followed by an integer separated by space. In this operation, a node with this integer as key is created and inserted into T.
   - Character 'd' is followed by an integer separated by space. In this operation, the node with this integer as key is deleted from T and the deleted node's key is printed.
   - Character *'s'* is followed by an integer separated by space. This operation is to find the node with this integer as key in T.

- Character *'i'* is to perform inorder traversal of T.
- Character *'p'* is to perform preorder traversal of T.
- Character *'t'* is to perform postorder traversal of T.
- Character *'e'* is to 'exit' from the program.

Output Format:

- The output (if any) of each command should be printed on a separate line.
- For option 'd', print the deleted node's key. If a node with the input key is not present in T, then print -1.
- For option 's', if the key is present in T, then print 1. If key is not present in T, then print -1.
- For option *'i'*, print the data in the nodes of T obtained from inorder traversal.
- For option *'p'*, print the data in the nodes of T obtained from preorder traversal.
- For option *'t'*, print the data in the nodes of T obtained from postorder traversal.

Sample Input :

```
a 25
a 13
a 50
a 45
a 55
a 18
i
p
t
s 10
s 25
d 55
d 13
d 10
d 25
i
s 25
e
```

Sample Output :

```
13 18 25 45 50 55
25 13 18 50 45 55
18 13 45 55 50 25
-1
1
55
13
-1
25
18 45 50
-1
```

2. Suppose in a car show room, the car details like model number, model name and price are stored in a BST. Modify the BINARY SEARCH TREE T in Qn No.1 so as to store for each car *model number*, *model name* and *price*. Assume that *model number* of a car is unique and the car details are organised in the BST, based on the value of *model number* of a car. Write a function MODIFY(n, p) that modifies the price of a car with *model number* as n with new price p. Declare *model number* and *price* as integers and *model name* as string.

Input format:

- Each line contains a character from 'a', 'd', 's', 'i', 'm', or 'e' followed by the arguments required for the corresponding operation.
- Character 'a' is followed by an integer n, a string s and an integer p separated by space. In this operation, a new node with *model number* as n, *model name* as s and *price* as p is inserted to T.
- Character 'd' is to delete the node with next integer as *model number* from T. Character 'd' and integer are separated by space.
- Character *'s'* is followed by an integer separated by space. This operation is to find the car details with this integer as *model number* in T.
- Character *'m'* is followed by two integers separated by space. This operation is to modify the price of a car with first integer as *model number* into second integer as new price in T.
- Character *'i'* is to perform inorder traversal of T.
- Character *'e'* is to 'exit' from the program.

Output Format:

- The output (if any) of each command should be printed on a separate line.
- For options 'd', print the deleted car's details in the order *model number*, *model name*, *price* separated by space. If a node with the input key is not present in T, then print -1.
- For option 's', if the key is present in T, then print all the details of that car in the order *model number*, *model name*, *price* separated by space. If key is not present in T, then print -1.
- For option *'i'*, print the data in the nodes of T obtained from inorder traversal. Each car's details are written on a separate line in the order *model number*, *model name*, *price* separated by space.

Sample Input :

a 2000 abv 250000
a 1990 xca 130000
a 2014 vta 500000
a 2019 ctwq 450000
a 1999 fgha 550000
a 2020 ghja 980000
i
s 2010
s 2000
d 1999

d 1990
d 2010
d 2000
i
s 2000
m 2019 650000
s 2019
e

Sample Output :

1990 xca 130000
1999 fgha 550000
2000 abv 250000
2014 vta 500000
2019 ctwq 450000
2020 ghja 980000
-1
2000 abv 250000
1999 fgha 550000
1990 xca 130000
-1
2000 abv 250000
2014 vta 500000
2019 ctwq 450000
2020 ghja 980000
-1
2019 ctwq 650000