CS2413: Assignment 5

Total Points: 100

Due: Dec 6, midnight.

Primary Objectives

Be able to implement a binary search tree and four operators: search, insertion, deletion and enumeration.

Be able to implement a stack class to assist the implementation of enumeration.

Project Overview

Our program should first take a sequence of integer keys as input and use them to construct a binary search tree. Each node of the tree stores one input key – for simplicity, we do not include satellite data this time.

Then, we should be able to search for a key from the tree, insert a key into the tree or delete a key from the tree. After any of the three operations, our program should automatically enumerate all nodes in the current tree – which may be updated after insertion or deletion, or remain the same after search – following the post order. In particular, we should implement a stack class to assist the post-order enumeration.

Project Specification

Our program should take three sets of inputs:

(i) an arbitrary sequence of non-repeated integer keys, ended with 's'. These keys will be used to construct the initial tree, i.e., be inserted into the tree in order.

(ii) an integer indicating which operation to perform on the current tree

- 1 indicates searching for a key

- 2 indicates inserting a key

- 3 indicates deleting a key

(iii) a key x associated with the operator

- if the task is search, then x can be arbitrary and won't be processed inside the program

- if the task is insertion, then x is the key to insert

- if the task is deletion, then x is the key to delete

Our program should give two possible outputs

- if the operation is successful, output the post-order enumeration of all nodes after the operation[1]

- if the operation fails, then output -1. We say the operation fails if (a) search for x but it is not in the tree, (b) insert x but it is already in the tree, or (c) delete x but it is not in the tree.

[1] For the search operation, if you find the key, just enumerate all nodes in the current tree. In practice, you are supposed to return the satellite data as we did in hw4; but we don't include that data for simplicity this time.

Example inputs and outputs are in Figures 1, 2 and 3. In particular, after inputting 14 27 36 9 7 11 s, our program is supposed to construct a binary search tree as in Figure 4.

1. Example of Successful Search

INPUT

14 27 36 9 7 11 s

1

27

OUTPUT

7 11 9 36 27 14

2. Example of Failing Search

INPUT

14 27 36 9 7 11 s

1

8

OUTPUT

-1

Fig. 1. Search

3. Example of Successful Insertion

INPUT

14 27 36 9 7 11 s

2

20

OUTPUT

7 11 9 20 36 27 14

4. Example of Failing Insertion

INPUT

14 27 36 9 7 11 s

2

36

OUTPUT

-1

Fig. 2. Insertion

5. Example of Successful Deletion

INPUT

14 27 36 9 7 11 s

3

11

OUTPUT

7 9 36 27 14

6. Example of Failing Deletion

INPUT

14 27 36 9 7 11 s

3

8

OUTPUT

-1

Fig. 3. Deletion

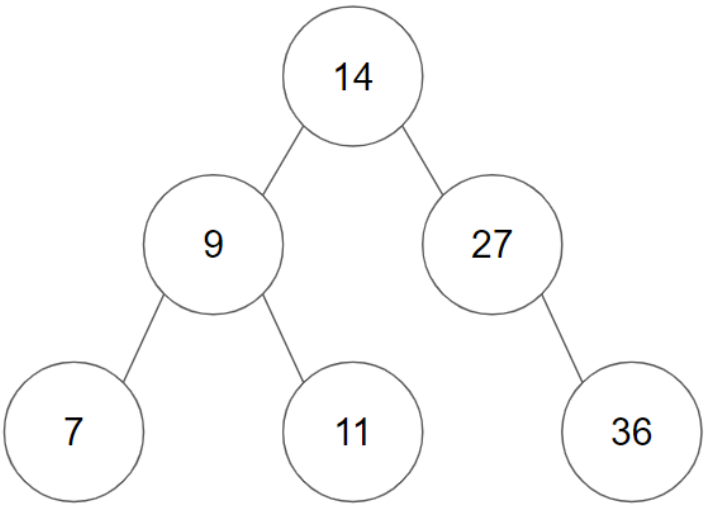

Fig. 4. Initial Binary Search Tree Constructed from 14 27 36 9 7 11 s.

Additional Requirements and Tips

You are free to choose proper data structures in implementation, except that you need to use stack to implement post-order enumeration.

To enable Gradescope grading, please only use 'cin' and 'cout' for data input and output.

Please name your submitted code as `cs2413_hw5.cpp`.

Rubrics

- search output: 35 points.

- insertion output: 20 points.

- deletion output: 35 points.

- Documentation: 10 points.