Lab Assignment: CS2233

3rd October, 2024

Instructions: Strictly follow the input and output format for each problem. **Your code should run in an infinite loop expecting for query number**.

Input format

- First line will contain n, which indicates the number of elements to be inserted into the tree.
- Second line will contain n one-space-separated integers, which are your tree elements. Assume this to be an array of n integers.
- Queries - Each line contains 2 integers separated by space character. First integer for type of query. 1 for search. 2 for insert. 3 for delete. 4 for displaying the tree in level order. For example.,
- See below for queries:
  - 1 112 $\implies$ search for 112 in the tree. Output - "112 present" / "112 not present" if found / not-found
  - 2 100 $\implies$ insert 100 into the tree. If the element 100 is already present, then print(100 already present. So no need to insert). Otherwise, it should be inserted and output print(100 inserted).
  - 3 100 $\implies$ delete 100 if present and output print(100 deleted). Otherwise, print(100 not present. So it can not be deleted).
  - 4 $\implies$ Print the tree level by level. Each level must be printed in each new line.

Example:
Input and Output:
17
9 8 5 4 99 78 31 34 89 90 21 23 45 77 88 112 32
1 56
(Output for this query. From now on, here, blue text represents the expected output)56 not present
2 21

21 already present. So no need to insert.
2 56
56 inserted
2 90
90 already present.So no need to insert
3 51
51 not present. So it can not be deleted
1 32
32 not present
3 32
32 not present. So it can not be deleted
4
" display the elements in level order"
Note:- Code should run in infinite loop expecting the query.

Max Marks 10

1 Coding Problems:

1. Construct an `(2-4)`-tree.

   The `(2-4)`-tree should be constructed by calling the `insert (root, key)` listed below. Each node of the tree should use the following `struct` data type:

   ```
   struct node
   {
   int data[3];
   struct node *childern[4];
   };
   ```

   You can assume that you have stored the pointer to the `root` node. Please, write the functions for:

   (a) `search (root, key)` – this function takes the pointer to the `root` node, and `key` as input, and returns the pointer to the node where `key` is present. If `key` is not present in the `(2-4)`-tree, then the code should output an error message.

   (b) `insert (root, key)` – this function takes the pointer to the `root` node, and key as input, and inserts the node at the appropriate position. Please run your function for inserting nodes $32, 56, 21, 90$.

   4+6=10 Marks