Lab 10: Red-Black Tree

In this lab, we will implement *delete* function to extend Red-Black Tree ADT.

1. Input

Two lines of numbers will be given as input. The numbers in the first line is for building Red-Black tree. By inserting the numbers in order, you can build the Red-Black tree. The numbers in the second line is for deleting the given elements from your Red-Black tree. If the number is not in the Red-Black tree, you should send an error message. If the number is in the Red-Black tree, you should print the tree by using inorder traversal after deleting it from.

Input.txt

```
7 5 3 10 23 4 20 21 22 23 24 25
21 3 5 11 23
```

2. RB tree ADT

(1) Data Specification for the objects

```
struct RBNode;
typedef struct RBNode *RBTree;

typedef struct RBNode{

        ElementType Element;
        int         red;    /* red=1 when the node is red */
        RBTree      left;
        RBTree      right;
        RBTree      parent;
}RBNode;
```

(2) Function specification

- RBTree delete(ElementType X, RBTree T)
- printInorder (RBTree T)

3. Program description

- name : p10.c
- input : two lines of numbers in a file (an input file name is given as a command line argument.)
- output : the corresponding result in the standard output

Submit to the course website (https://portal.hanyang.ac.kr) your source code and a written report. Your report should include the description of your own implementation.