Problem A
BST 實作

Time limit: 1 second
Memory limit: 256 megabytes

題目內容

請用指標實作一個二元搜尋樹，內存入數字，需實作四種方法

1. 往 BST 中插入節點，需符合 BST 的規則，若該節點已存在，則不進行任何操作
2. 刪除指定節點，刪除後的樹仍須保持 BST 的規則，若節點不存在則不進行動作
3. 查詢節點是否在 BST 中存在
4. 將整顆樹依照規定進行 Traversal

輸入格式

第一行輸入一個數字 n ，代表有 n 個操作

接著有 n 行輸入，分別有四種狀況

1. 輸入數字 1 以及 num 中間分別以空白隔開，此操作後將在 BST 中插入一個值爲 num 的節點，若樹中已有值爲 num 的節點，不進行任何操作．
2. 輸入 2 以及 num 中間分別以空白隔開，此操作後將在 BST 中刪除一個值爲 num 的節點，若樹中無值爲 num 的節點，不進行任何操作．
3. 輸入 3 以及 $mode$ 中間分別以空白隔開，若 $mode$ 爲 1 ，則輸出該 BST 的 Preorder Traversal，若 $mode$ 爲 2 ，則輸出該 BST 的 Inorder Traversal，若 $mode$ 爲 3 ，則輸出該 BST 的 Postorder Traversal．各值之間以空白隔開，Traversal 完後需換行．
4. 輸入 4 以及 $nums$ 中間分別以空白隔開，若 $nums$ 在此 BST 中存在則輸出 **True** ，反之則輸出 **False**

輸出格式

在進行 3 和 4 操作時，需要進行輸出，每次操作輸出完需換行

技術規格

- $1 \leq n \leq 10^6$
- $1 \leq num \leq 10^9$
- $1 \leq mode \leq 3$
- $sum(mode) \leq 20$

範例輸入 1

```
5
1 3
1 2
1 5
3 2
4 3
```

範例輸出 1

```
2 3 5
True
```

範例輸入 2

```
7
1 8
1 3
1 6
3 2
1 7
3 2
4 9
```

範例輸出 2

```
3 6 8
3 6 7 8
False
```

Note

```
struct node{
    int num;
    node *left;
    node *right;
}typedef node

node *insert(node *root, int num){

}

node *Delete(node *root, int num){

}

void Traversal(node *root, int mode){

}
```

可參考此格式