https://course.acciojob.com/idle?question=e552caac-75e2-4f1e-8aee-a49ae6c08134

MEDIUM

Max Score: 40 Points

Insert a node in a BST

You are given the root node of a binary search tree (BST) and a value to insert into the tree. Return the root node of the BST after the insertion. It is guaranteed that the new value does not exist in the original BST.

Input Format

The first line inputs N, the number of nodes, and K, the key.

The second line inputs the value of N nodes of the BST.

Output Format

Print the PreOrder traversal of the BST in a new line.

Example 1

Input

```
7 44
2 81 42 87 90 42 41
```

Output

```
2 81 42 41 44 87 90
```

Explaination

```
  2
    \
    81
   /  \
  42   87
 /  \   \
41  44   90

As 44 is not present in the given nodes, so the tree will change and preorder of the
updated tree is 2 81 42 41 66 44 87 90.
```

Example 2

Input

```
7 25
40 20 60 10 30 50 70
```

Output

```
40 20 10 30 25 60 50 70
```

Explaination

```
As 25 is not present in the given nodes, so the tree will  change and preorder of the
updated tree is 40 20 10 30 25 60 50 70.
```

Constraints:

1 <= N <= 1000

-1000 <= Val[node], K <= 1000

Topic Tags

Trees

My code

```
// in java
import java.util.*;
import java.lang.*;
import java.io.*;

class Node
{
    int data;
    Node next ,prev;

    Node(int data, Node next,Node prev)
    {
        this.data = data;
        this.next = next;
        this.prev = prev;
    }

    Node() {}
}

public class Main
{
  static Node insert(Node root,int n)
  {
    if(root==null)
    {
      root=new Node(n,null,null);
    return root;
    }
   else if(n< root.data)
      root.prev= insert( root.prev, n);
    else   if(n>root.data)
      root.next= insert( root.next, n);
    return root;
  }

  static void preorder(Node root)
  {
    if(root !=null)
    {
      System.out.print(root.data+" ");
      preorder(root.prev);
      preorder(root.next);
    }
  }

        public static void main (String[] args) throws
java.lang.Exception
        {
                //your code here
        Scanner s=new Scanner(System.in);
      int n=s.nextInt();
       int k=s.nextInt();
      //int arr[]=new int[n];
      Node root=null;
      for(int i=0;i<n;i++)
      {
      int m=s.nextInt();
        root=insert( root, m);
      }
      root=insert( root, k);
        preorder(root);
        }
}
```