Assignment-12

Implement a binary tree in Java supporting insertion, deletion, and traversal operations. Ensure the tree can handle basic operations efficiently and provide methods for inorder, preorder, and postorder traversals.

```
class Node {
   int key;
   Node left, right;

   public Node(int item) {
      key = item;
      left = right = null;
   }
}

class Bst {
   Node root;

   //constructor
   Bst() {
      root = null;
   }

   //insert a new key
   void insert(int key) {
      root = insertRec(root, key);
   }



   Node insertRec(Node root, int key) {
      if (root == null) {
         root = new Node(key);
         return root;
      }
```

```
  if (key < root.key)
      root.left = insertRec(root.left, key);
   else if (key > root.key)
      root.right = insertRec(root.right, key);

   return root;
}

void inorderRec(Node root) {
   if (root != null) {
      inorderRec(root.left);
      System.out.print(root.key + " ");
      inorderRec(root.right);
   }
}


void preorderRec(Node root) {
   if (root != null) {
      System.out.print(root.key + " ");
      preorderRec(root.left);
      preorderRec(root.right);
   }
}

void postorderRec(Node root) {
   if (root != null) {
      postorderRec(root.left);
      postorderRec(root.right);
      System.out.print(root.key + " ");
   }
}

void inorder() {
   inorderRec(root);
}

void preorder() {
   preorderRec(root);
```

```
    }

    void postorder() {
        postorderRec(root);
    }
}

public class BinaryTree{
    public static void main(String[] args) {
            Bst tree = new Bst();

            tree.insert(25);
            tree.insert(15);
            tree.insert(10);
            tree.insert(4);
            tree.insert(12);
            tree.insert(22);
            tree.insert(18);
            tree.insert(24);
            tree.insert(50);
            tree.insert(35);
            tree.insert(31);
            tree.insert(44);
            tree.insert(70);
            tree.insert(66);
            tree.insert(90);


        System.out.println("Inorder traversal of the given tree:");
        tree.inorder();
        System.out.println();

        System.out.println("\nPreorder traversal of the given tree:");
        tree.preorder();
        System.out.println();

        System.out.println("\nPostorder traversal of the given tree:");
        tree.postorder();
        System.out.println();
    }
```

}