

LinkedBST.java

```

1 /*
2  * Written by JJ Shepherd
3  */
4 public class LinkedBST <T extends Comparable<T>>
5 {
6     private class Node
7     {
8         T data;
9         Node leftChild;
10        Node rightChild;
11        public Node(T aData)
12        {
13            data = aData;
14            leftChild = rightChild = null;
15        }
16    }
17    private Node root;//head
18    public LinkedBST()
19    {
20        root = null;
21    }
22    public void add(T aData)
23    {
24        if(root == null)
25            root = new Node(aData);
26        else
27            add(root,aData);
28    }
29    private Node add(Node aNode, T aData)
30    {
31        if(aNode == null)
32            aNode = new Node(aData);
33        else if(aData.compareTo(aNode.data)<0)//GO LEFT
34            aNode.leftChild = add(aNode.leftChild,aData);
35        else if(aData.compareTo(aNode.data)>0)//GO RIGHT
36            aNode.rightChild = add(aNode.rightChild,aData);
37        return aNode;
38    }
39    public void printPreorder()
40    {
41        printPreorder(root);
42    }
43    private void printPreorder(Node aNode)
44    {
45        if(aNode == null)
46            return;
47        System.out.println(aNode.data);//PROCESS
48        printPreorder(aNode.leftChild);//LEFT
49        printPreorder(aNode.rightChild);//RIGHT
50    }
51    public void printInorder()
52    {
53        printInorder(root);
54    }
55    private void printInorder(Node aNode)
56    {
57        if(aNode == null)

```

LinkedBST.java

```
58         return;
59     printInorder(aNode.leftChild); //LEFT
60     System.out.println(aNode.data); //PROCESS
61     printInorder(aNode.rightChild); //RIGHT
62 }
63 }
64
```