

```
1  import java.io.*;
2  import java.util.*;
3
4
5  class BTreeNode{
6
7      private int nodeId;
8      private int data;
9      private int levelNum;
10     private BTreeNode leftChildPtr;
11     private BTreeNode rightChildPtr;
12
13     public BTreeNode () {}
14
15     public void setNodeId(int id){
16         nodeId = id;
17     }
18
19     public int getNodeId(){
20         return nodeId;
21     }
22
23     public void setData(int d){
24         data = d;
25     }
26
27     public int getData(){
28         return data;
29     }
30
31     public void setLevelNum(int level){
32         levelNum = level;
33     }
34
35     public int getLevelNum(){
36         return levelNum;
37     }
38
39     public void setLeftChildPtr(BTreeNode ptr){
40         leftChildPtr = ptr;
41     }
42
43     public void setRightChildPtr(BTreeNode ptr){
44         rightChildPtr = ptr;
45     }
46
47     public BTreeNode getLeftChildPtr(){
48         return leftChildPtr;
49     }
50
51     public BTreeNode getRightChildPtr(){
52         return rightChildPtr;
53     }
54
55     public int getLeftChildID(){
56         if (leftChildPtr == null)
57             return -1;
58
59         return leftChildPtr.getNodeId();
60     }
61
62     public int getRightChildID(){
63         if (rightChildPtr == null)
64             return -1;
```

```

65
66     return rightChildPtr.getNodeId();
67 }
68
69 }
70
71
72
73 class BinarySearchTree{
74
75     private int numNodes;
76     private BTNode[] arrayOfBTNodes;
77     private int rootNodeID;
78
79     public BinarySearchTree(int n){
80         numNodes = n;
81         arrayOfBTNodes = new BTNode[numNodes];
82
83         for (int index = 0; index < numNodes; index++){
84             arrayOfBTNodes[index] = new BTNode();
85             arrayOfBTNodes[index].setNodeId(index);
86             arrayOfBTNodes[index].setLeftChildPtr(null);
87             arrayOfBTNodes[index].setRightChildPtr(null);
88             arrayOfBTNodes[index].setLevelNum(-1);
89
90         }
91     }
92
93
94     public void setLeftLink(int upstreamNodeID, int downstreamNodeID){
95         arrayOfBTNodes[upstreamNodeID].setLeftChildPtr(arrayOfBTNodes[downstreamNodeID]);
96     }
97
98     public void setRightLink(int upstreamNodeID, int downstreamNodeID){
99         arrayOfBTNodes[upstreamNodeID].setRightChildPtr(arrayOfBTNodes[downstreamNodeID
100     ]);
101
102
103     public void constructBSTree(int[] array){
104
105         int leftIndex = 0;
106         int rightIndex = numNodes-1;
107         int middleIndex = (leftIndex + rightIndex)/2;
108
109         rootNodeID = middleIndex;
110         arrayOfBTNodes[middleIndex].setData(array[middleIndex]);
111
112         ChainNodes(array, middleIndex, leftIndex, rightIndex);
113
114     }
115
116
117     public void ChainNodes(int[] array, int middleIndex, int leftIndex, int rightIndex){
118
119         if (leftIndex < middleIndex){
120             int rootIDLeftSubtree = (leftIndex + middleIndex-1)/2;
121             setLeftLink(middleIndex, rootIDLeftSubtree);
122             arrayOfBTNodes[rootIDLeftSubtree].setData(array[rootIDLeftSubtree]);
123             ChainNodes(array, rootIDLeftSubtree, leftIndex, middleIndex-1);
124         }
125
126
127         if (rightIndex > middleIndex){

```

```

128         int rootIDRightSubtree = (rightIndex + middleIndex + 1)/2;
129         setRightLink(middleIndex, rootIDRightSubtree);
130         arrayOfBTNodes[rootIDRightSubtree].setData(array[rootIDRightSubtree]);
131         ChainNodes(array, rootIDRightSubtree, middleIndex+1, rightIndex);
132     }
133
134
135 }
136
137
138 public void printLeafNodes(){
139
140     for (int id = 0; id < numNodes; id++){
141
142         if (arrayOfBTNodes[id].getLeftChildPtr() == null && arrayOfBTNodes[id].
143             getRightChildPtr() == null)
144             System.out.print(arrayOfBTNodes[id].getData() + " ");
145     }
146
147     System.out.println();
148 }
149
150
151
152 }
153
154
155
156 class BSTImplementation{
157
158     public static void main(String[] args){
159
160         Scanner input = new Scanner(System.in);
161
162         int numElements;
163         System.out.print("Enter the number of elements: ");
164         numElements = input.nextInt();
165
166         int array[] = new int[numElements];
167
168         for (int index = 0; index < numElements; index++){
169             System.out.print("Enter element at index " + index + ": ");
170             array[index] = input.nextInt();
171         }
172
173         BinarySearchTree bsTree = new BinarySearchTree(numElements);
174         bsTree.constructBSTree(array);
175
176         System.out.print("Leaf Nodes: ");
177         bsTree.printLeafNodes();
178         System.out.println();
179
180     }
181
182 }

```

```

Enter the number of elements: 7
Enter element at index 0: 12
Enter element at index 1: 15
Enter element at index 2: 18
Enter element at index 3: 23
Enter element at index 4: 25
Enter element at index 5: 29
Enter element at index 6: 64
Leaf nodes: 12 18 25 64

```