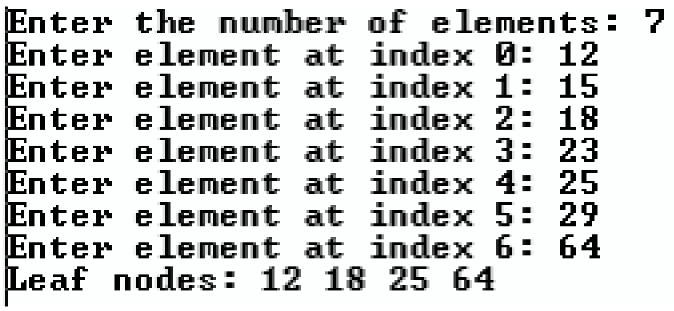
```java
import java.io.*;
import java.util.*;


class BTNode{

    private int nodeid;
    private int data;
    private int levelNum;
    private BTNode leftChildPtr;
    private BTNode rightChildPtr;

    public BTNode(){}

    public void setNodeId(int id){
        nodeid = id;
    }

    public  int getNodeId(){
        return nodeid;
    }

    public  void setData(int d){
        data = d;
    }

    public int getData(){
        return data;
    }

    public  void setLevelNum(int level){
        levelNum = level;
    }

    public  int getLevelNum(){
        return levelNum;
    }

    public  void setLeftChildPtr(BTNode ptr){
        leftChildPtr = ptr;
    }

    public  void setRightChildPtr(BTNode ptr){
        rightChildPtr = ptr;
    }

    public  BTNode getLeftChildPtr(){
        return leftChildPtr;
    }

    public  BTNode getRightChildPtr(){
        return rightChildPtr;
    }

    public  int getLeftChildID(){
        if (leftChildPtr == null)
            return -1;

        return leftChildPtr.getNodeId();
    }

    public  int getRightChildID(){
        if (rightChildPtr == null)
            return -1;
        return rightChildPtr.getNodeId();
    }

}



class BinarySearchTree{

    private int numNodes;
    private BTNode[] arrayOfBTNodes;
    private int rootNodeID;

    public  BinarySearchTree(int n){
        numNodes = n;
        arrayOfBTNodes = new BTNode[numNodes];

        for (int index = 0; index < numNodes; index++){
            arrayOfBTNodes[index] = new BTNode();
            arrayOfBTNodes[index].setNodeId(index);
            arrayOfBTNodes[index].setLeftChildPtr(null);
            arrayOfBTNodes[index].setRightChildPtr(null);
            arrayOfBTNodes[index].setLevelNum(-1);

        }
    }


    public void setLeftLink(int upstreamNodeID, int downstreamNodeID){
        arrayOfBTNodes[upstreamNodeID].setLeftChildPtr(arrayOfBTNodes[downstreamNodeID]);
    }

    public void setRightLink(int upstreamNodeID, int downstreamNodeID){
        arrayOfBTNodes[upstreamNodeID].setRightChildPtr(arrayOfBTNodes[downstreamNodeID
        ]);
    }


    public  void constructBSTree(int[] array){

        int leftIndex = 0;
        int rightIndex = numNodes-1;
        int middleIndex = (leftIndex + rightIndex)/2;

        rootNodeID = middleIndex;
        arrayOfBTNodes[middleIndex].setData(array[middleIndex]);

        ChainNodes(array, middleIndex, leftIndex, rightIndex);

    }


    public  void ChainNodes(int[] array, int middleIndex, int leftIndex, int rightIndex){

        if (leftIndex < middleIndex){
            int rootIDLeftSubtree = (leftIndex + middleIndex-1)/2;
            setLeftLink(middleIndex, rootIDLeftSubtree);
            arrayOfBTNodes[rootIDLeftSubtree].setData(array[rootIDLeftSubtree]);
            ChainNodes(array, rootIDLeftSubtree, leftIndex, middleIndex-1);
        }


        if (rightIndex > middleIndex){
```

```java
            int rootIDRightSubtree = (rightIndex + middleIndex + 1)/2;
            setRightLink(middleIndex, rootIDRightSubtree);
            arrayOfBTNodes[rootIDRightSubtree].setData(array[rootIDRightSubtree]);
            ChainNodes(array, rootIDRightSubtree, middleIndex+1, rightIndex);
        }


    }


    public void printLeafNodes(){

        for (int id = 0; id < numNodes; id++){

            if (arrayOfBTNodes[id].getLeftChildPtr() == null && arrayOfBTNodes[id].
            getRightChildPtr() == null)
                System.out.print(arrayOfBTNodes[id].getData() + " ");
        }

        System.out.println();
    }




}



class BSTImplementation{

    public static void main(String[] args){

    Scanner input = new Scanner(System.in);

    int numElements;
    System.out.print("Enter the number of elements: ");
    numElements = input.nextInt();

    int array[] = new int[numElements];

    for (int index = 0; index < numElements; index++){
        System.out.print("Enter element at index " + index + ": ");
        array[index] = input.nextInt();
    }

    BinarySearchTree bsTree = new BinarySearchTree(numElements);
    bsTree.constructBSTree(array);

    System.out.print("Leaf Nodes: ");
    bsTree.printLeafNodes();
    System.out.println();

    }

}
```



```
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
```