

```
1 #include <iostream>
2 using namespace std;
3
4 class BTNode{
5
6     private:
7         int nodeid;
8         int data;
9         int levelNum;
10        BTNode* leftChildPtr;
11        BTNode* rightChildPtr;
12
13    public:
14
15        BTNode() {}
16
17        void setNodeId(int id){
18            nodeid = id;
19        }
20
21        int getNodeId(){
22            return nodeid;
23        }
24
25        void setData(int d){
26            data = d;
27        }
28
29        int getData(){
30            return data;
31        }
32
33        void setLevelNum(int level){
34            levelNum = level;
35        }
36
37        int getLevelNum(){
38            return levelNum;
39        }
40
41        void setLeftChildPtr(BTNode* ptr){
42            leftChildPtr = ptr;
43        }
44
45        void setRightChildPtr(BTNode* ptr){
46            rightChildPtr = ptr;
47        }
48
49        BTNode* getLeftChildPtr(){
50            return leftChildPtr;
51        }
52
53        BTNode* getRightChildPtr(){
54            return rightChildPtr;
55        }
56
57        int getLeftChildID(){
58            if (leftChildPtr == 0)
59                return -1;
60
61            return leftChildPtr->getNodeId();
62        }
63
64        int getRightChildID(){
```

```

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

```

```

127         setLeftLink(middleIndex, rootIDLeftSubtree);
128         arrayOfBTNodes[rootIDLeftSubtree].setData(array[rootIDLeftSubtree]);
129         ChainNodes(array, rootIDLeftSubtree, leftIndex, middleIndex-1);
130     }
131
132
133     if (rightIndex > middleIndex) {
134         int rootIDRightSubtree = (rightIndex + middleIndex + 1)/2;
135         setRightLink(middleIndex, rootIDRightSubtree);
136         arrayOfBTNodes[rootIDRightSubtree].setData(array[rootIDRightSubtree]);
137         ChainNodes(array, rootIDRightSubtree, middleIndex+1, rightIndex);
138     }
139
140 }
141
142
143
144 void printLeafNodes() {
145
146     for (int id = 0; id < numNodes; id++) {
147
148         if (arrayOfBTNodes[id].getLeftChildPtr() == 0 && arrayOfBTNodes[id].
149             getRightChildPtr() == 0)
150             cout << arrayOfBTNodes[id].getData() << " ";
151     }
152
153     cout << endl;
154 }
155
156
157 };
158
159
160 int main() {
161
162     int numElements;
163     cout << "Enter the number of elements: ";
164     cin >> numElements;
165
166     BinarySearchTree bsTree(numElements);
167
168     int array[numElements];
169
170     for (int index = 0; index < numElements; index++) {
171         cout << "Enter element at index " << index << ": ";
172         cin >> array[index];
173     }
174
175     bsTree.constructBSTree(array);
176
177     cout << "Leaf nodes: ";
178     bsTree.printLeafNodes();
179     cout << endl;
180
181     return 0;
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

```