

```

1  import java.util.Scanner;
2  import unit4.collectionsLib.*;
3  import unit4.utilsLib.BinTreeUtils;
4
5  public class BTNmethods
6  {
7      public static void main(String[] args)
8      {
9          int maxNodes = 15;          // מספר הצמתים בעץ
10         int low = -10;              // ערך מינימאלי של value
11         int high = 10;              // ערך מקסימאלי של value
12
13         BinNode<Integer> bt = TreeUtils.buildRandomTree(maxNodes, low, high);
14         TreeUtils.showTree(bt, "bt");
15
16         System.out.println(numNodes(bt) + " : מספר הצמתים בעץ ");
17         System.out.println(sumNodes(bt) + " : סכום הצמתים בעץ ");
18         System.out.println();
19
20         System.out.println(numPositive(bt) + " : מספר הצמתים החיוביים בעץ ");
21         System.out.println(sumPositive(bt) + " : סכום הצמתים החיוביים בעץ ");
22         System.out.println();
23
24         System.out.println(countLeftSons(bt) + " : מספר הבנים השמאליים");
25         System.out.println(sumLeftSons(bt) + " : סכום הבנים השמאליים");
26         System.out.println();
27
28         System.out.println(countRightSons(bt) + " : מספר הבנים הימניים");
29         System.out.println(sumRightSons(bt) + " : סכום הבנים הימניים");
30         System.out.println();
31
32         System.out.println(countLeaves(bt) + " : מספר העלים");
33         System.out.println(sumLeaves(bt) + " : סכום העלים");
34
35         System.out.println(maxInTree(bt) + " : ערך מקסימאלי בעץ");
36     }
37
38
39
40
41
42
43
44
45
46     /*****
47     * אוסף פעולות שימושיות בעץ בינארי *
48     *****/
49
50     //--- מספר הצמתים בעץ ---
51     public static int numNodes (BinNode <Integer> bt)
52     {
53         if (bt == null)
54             return 0;
55         return 1 +          //-- בגלל הצומת הנוכחי
56             numNodes(bt.getLeft()) +          //-- מספר הצמתים בבן השמאלי
57             numNodes(bt.getRight());          //-- מספר הצמתים בבן הימני
58     }
59
60     //--- סכום הצמתים בעץ ---
61     public static int sumNodes (BinNode <Integer> bt)
62     {
63         if (bt == null)
64             return 0;
65         return bt.getValue() +          // ערך הצומת הנוכחי
66             sumNodes(bt.getLeft()) +          // סכום הצמתים בבן השמאלי
67             sumNodes (bt.getRight());          // סכום הצמתים בבן הימני
68     }
69
70     //--- *** מספר הצמתים המקיימים תנאי *** ---
71     // --- מספר הצמתים החיוביים בעץ ---
72     public static int numPositive (BinNode <Integer> bt)
73     {
74         if (bt == null)
75             return 0;          //--- זהו התנאי המבוקש
76         if (bt.getValue() > 0)          //--- אפשר להחליף אותו בתנאי אחר

```

```

77         return 1 + numPositive(bt.getLeft()) +
78             numPositive (bt.getRight());
79     return numPositive(bt.getLeft()) +
80         numPositive(bt.getRight());
81     }
82
83     // --- סכום הצמתים החיוביים בעץ ---
84     public static int sumPositive (BinNode <Integer> bt)
85     {
86         if (bt == null)
87             return 0;
88         if (bt.getValue() > 0)
89             return bt.getValue() +
90                 sumPositive(bt.getLeft()) +
91                 sumPositive (bt.getRight());
92         return sumPositive(bt.getLeft()) +
93             sumPositive(bt.getRight());
94     }
95
96     // --- מספר הבנים השמאליים בעץ ---
97     public static int countLeftSons (BinNode <Integer> bt)
98     {
99         if (bt == null)
100             return 0;
101         if (bt.getLeft() != null)
102             return 1 + countLeftSons(bt.getLeft()) +
103                 countLeftSons(bt.getRight());
104         return countLeftSons (bt.getRight());
105     }
106
107     // --- מספר הבנים הימניים בעץ ---
108     public static int countRightSons (BinNode <Integer> bt)
109     {
110         if (bt == null)
111             return 0;
112         if (bt.getRight() != null)
113             return 1 + countRightSons(bt.getLeft()) +
114                 countRightSons(bt.getRight());
115         return countRightSons (bt.getLeft());
116     }
117
118     // --- סכום הבנים השמאליים בעץ ---
119     public static int sumLeftSons (BinNode <Integer> bt)
120     {
121         if (bt == null)
122             return 0;
123         if (bt.getLeft() != null)
124             return bt.getLeft().getValue() + // ערך הבן השמאלי
125                 sumLeftSons(bt.getLeft()) +
126                 sumLeftSons (bt.getRight());
127         return sumLeftSons (bt.getRight());
128     }
129
130     // --- סכום הבנים הימניים בעץ ---
131     public static int sumRightSons (BinNode <Integer> bt)
132     {
133         if (bt == null)
134             return 0;
135         if (bt.getRight() != null)
136             return bt.getRight().getValue() +
137                 sumRightSons(bt.getLeft()) +
138                 sumRightSons (bt.getRight());
139         return sumRightSons (bt.getLeft());
140     }
141
142     //--- האם עלה? ---
143     public static boolean isLeaf (BinNode <Integer> bt)
144     {
145         if (bt == null)
146             return false;
147         if (bt.getLeft() == null && bt.getRight() == null)
148             return true;
149         return false;
150
151         /*----- אפשרות נוספת -----
152         * null יכולים להיות שווים רק כאשר הם

```

```
153
154     return (bt.getLeft() == bt.getRight()) ;
155
156     */
157 }
158
159 //--- מספר העלים בעץ ---
160 public static int countLeaves (BinNode <Integer> bt)
161 {
162     if (bt == null)
163         return 0;
164     if (isLeaf(bt))
165         return 1;
166     return countLeaves(bt.getLeft()) + countLeaves(bt.getRight());
167 }
168
169 // --- סכום העלים בעץ ---
170 public static int sumLeaves (BinNode <Integer> bt)
171 {
172     if (bt == null)
173         return 0;
174     if (isLeaf(bt))
175         return bt.getValue();
176     return sumLeaves(bt.getLeft()) + sumLeaves(bt.getRight());
177 }
178
179 //--- עץ בינארי לא ריק ---
180 //--- הערך המקסימאלי בעץ ---
181 public static int maxInTree (BinNode <Integer> bt)
182 {
183     if (bt == null)
184         return 0;
185     int maxLeft = maxInTree(bt.getLeft());
186     int maxRight = maxInTree(bt.getRight());
187
188     return Math.max(bt.getValue(), Math.max(maxLeft, maxRight));
189 }
190 }
191
```