Study.com®

[Computer Science 201: Data Structures & Algorithms]

[Computer Science 201 - Assignment 1: Creating a Binary Search Tree]

Category	**Unacceptable (0-2)**	**Needs Improvement (3-6)**	**Good (7-8)**	**Excellent (9-10)**	**Total**
Program Specification (x3)	The program does not or partially meets the requirements and contains multiple major errors.	The program partially meets the requirements or contains at least one major error.	Your Assignment: The program meets all the requirements but contains one or two minor errors.	The program meets all the requirements and works without any errors.	24/30
Code Efficiency (x1)	The code employs inefficient algorithms and includes unnecessary components.	The code employs inefficient algorithms or includes unnecessary components.	Your Assignment: The code employs efficient algorithms but includes some unnecessary components.	The code employs efficient algorithms and doesn't include unnecessary components.	8/10
Code Readability (x.5)	The code is not easily understandable and contains improper naming and formatting.	Most parts of the code are not easily understandable or contain improper naming and formatting.	The code is mostly understandable and uses proper naming and formatting.	Your Assignment: The code is easily understandable and well-organized and uses proper naming and formatting.	5/5
Documentation (x.5)	No or very few documentation exists.	The documentation is ambiguous or doesn't not explain what the code is accomplishing and how.	The documentation explains what the code is accomplishing and how but doesn't cover all the important parts of the code.	Your Assignment: The documentation clearly explains what the code is accomplishing and how.	5/5
Total Points					42/50

Total Points: 42/50

Grader Notes:

- Program Specifications (-6)
    - You used hard-coding for creating a binary search tree. This approach is not recommended due to the following reasons:
        - Scalability
        - Maintainability
        - Reusability
        - Configuration management
    - Try to make the code working without hard-coding.
        - Hint.

```
 Node createBalancedTreeRec(int[] values, int start, int end) {
     if (start > end) return null;
     int mid = (start + end) / 2;
     Node node = new Node(values[mid])
     node.left = createBalancedTreeRec(values, start, mid – 1);
     node.right = createBalancedTreeRec(values, mid + 1, end);
     return node;
}
int[] values = {1, 2, 3, 4, 5, 6, 7};
```

- Code efficiency (-2)
    - You could improve the efficiency of your code by removing the hard-coding.

- Code Readability
    - Clear and easily understandable code. Good job!

- Documentation
    - The comments in the code clearly explain the purpose and method of each of the important parts. Good job!

- Others
    - You will have a chance to learn data structure and algorithms later. Explore some of the concepts related to them and see how you could improve your code.
    - If you'd like an opportunity to improve your score, I encourage you to resubmit the assignment. Take some time to review the feedback provided and make any necessary revisions.