

# BST Number List in Console/Terminal

## CptS 321 Homework Assignment

### Washington State University

Submission Instructions: (see syllabus, will work with Git to submit)

- Your code shall live in a directory called “BSTtree”
- The instructor shall push these instructions to a new branch called “BSTtree” in your course repository. This repository is a private one that lives on the EECS GitLab server:
  - [gitlab.eecs.wsu.edu](https://gitlab.eecs.wsu.edu)
- All repositories are named using the same pattern of: cs321-`{EECS_USERNAME}`. For example, my username is ‘acrandal’ so my repository is cs321-acrandal
- If you’ve filled out the username survey your repository should be created for you by the TAs and ready to use.
- Once you’ve finished the assignment and pushed your project to the repository, give it a Git Tag of the name “BSTtree-v1.0”. We’ll show how to do this in class.
  - <https://git-scm.com/book/en/v2/Git-Basics-Tagging>

Assignment Instructions: Read all the instructions *carefully* before you write any code.

Create a C# console application that fulfills the following requirements:

- 1) Get a list of integer numbers from the user on A SINGLE LINE
  - a) The numbers will be in the range [0,100]
  - b) The numbers will be separated by spaces
  - c) You may assume that the user enters a correctly formatted input string that meets these requirements
  - d) You may use [Console.ReadLine](#) or one of the other methods we might discuss in class to get input from the user
  - e) Use the [Split](#) function on the entered string for easy parsing
    - i) split on the space character
- 2) Add all the numbers to a binary search tree in the order they were entered
  - a) Don’t allow duplicates
    - i) Insert code should detect the duplicate value and return immediately
- 3) Display the numbers in sorted order (smallest first, largest last)
  - a) This is done with an inorder tree traversal
- 4) Display the following statistics about the tree:
  - a) Number of items (note that this will be less than or equal to the number of items entered by the user, since duplicates won’t be added to the tree). Write a method

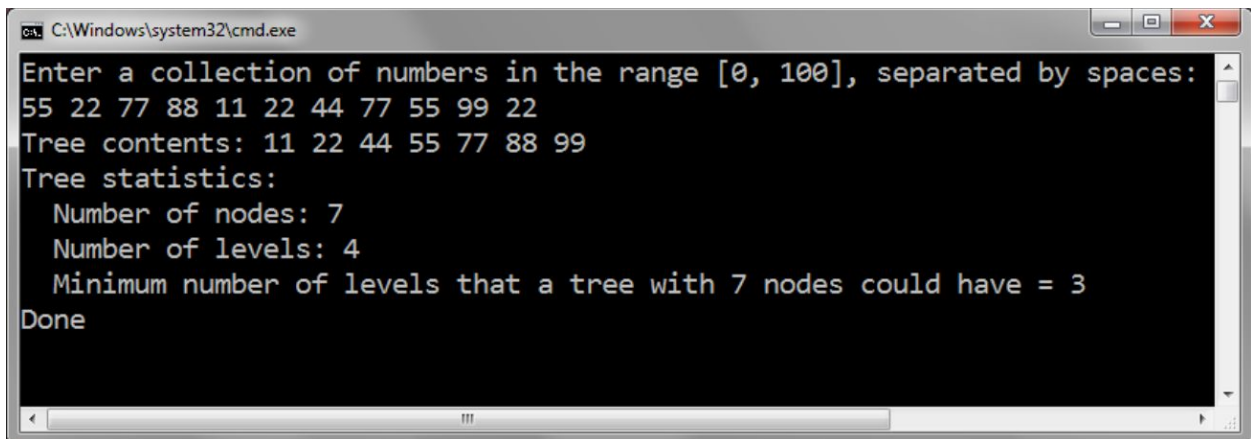
called Count() that determines this from your BST, NOT the array returned from the split. In other words, you must have a Count method in your BST implementation.

- i) This can either be done by keeping track of inserts or as a tree traversal
- b) The tree's depth.
  - i) A tree with no nodes at all has 0 levels. A tree with a single node has 1 level. A tree with 2 nodes has 2 levels. A tree with three nodes could have a 2 or 3 levels. You should know why this is from your advanced data structures prerequisite course.
  - ii) You can calculate the number of levels by finding the maximum depth of any node in the tree. Remember: The root is depth 0, so you're counting the number of edges from a given node the root to get the depth. You could also just report the height of the root node, that's also equal to the number of levels.
- c) Theoretical minimum number of levels that the tree could have given the number of nodes it contains (figure out the formula to calculate this)
  - i) Base this calculation on your Count() function

Point breakdown of 10 total:

- 9 points: Fulfill all the requirements above with no inaccuracies in the output and no crashes
- 1 point: Code is clean, efficient, and well commented

Sample Output:



```
C:\Windows\system32\cmd.exe
Enter a collection of numbers in the range [0, 100], separated by spaces:
55 22 77 88 11 22 44 77 55 99 22
Tree contents: 11 22 44 55 77 88 99
Tree statistics:
  Number of nodes: 7
  Number of levels: 4
  Minimum number of levels that a tree with 7 nodes could have = 3
Done
```