CSC 225 - SPRING 2018
ALGORITHMS AND DATA STRUCTURES I
PROGRAMMING ASSIGNMENT 3
UNIVERSITY OF VICTORIA

1 Programming Assignment

The assignment is to create a binary search tree (BST) and test whether the tree is a valid AVL tree.

A Java template has been provided containing two classes: `AVL_BST` and `BST`. Your task is to write the body of the `createBST` and the `checkAVL` functions. You have complete freedom in designing the `BST` class, and you can add any helper methods you need in the `AVL_BST` class. You may introduce more private classes, but they have to be declared in the same file. Your code is not required to check for incorrectly formed input data.

You must use the provided Java template as the basis of your submission. You may not change the name, return type or parameters of any of the given functions. The main function in the template contains code to help you test your implementation. You may modify the main function, since the main function will be deleted before marking begins. You can submit ONLY the file named "AVL_BST.java". If you submit multiple files or change the name of the class or the functions (even lowercase to uppercase or vice versa), your submission may generate compilation error during the automated testing and you will be graded accordingly.

2 Examples

The samples below show the correct output for various test inputs.

Input Array	`checkAVL()` **output**
$82, 85, 153, 195, 124, 66, 200, 193, 185, 243, 73, 153, 76$	false
$5, 3, 7, 1$	true
$5, 1, 98, 100, -3, -5, 55, 3, 56, 50$	true
$297, 619, 279, 458, 324, 122, 505, 549, 83, 186, 131, 71$	false
78	true

3 Evaluation Criteria

The programming assignment will be marked out of 50, based on a combination of automated testing and human inspection. You may safely assume that the maximum size of an input array will be 10000.

The mark for each submission will be based on both the asymptotic worst case running time and the ability of the algorithm to handle inputs of different sizes. The table below shows the expectations associated with different scores.

Score	**Description**
0 - 10	Submission does not compile or does not conform to the provided template.
11 - 25	The submission is substantially inaccurate on the tested inputs (fails more than 50% of the test cases).
26 - 30	Any of the two functions `createBST` and `checkAVL` takes more than $O(n \log n)$ time.
31 - 50	Each of the two functions takes $O(n \log n)$ time. The numbers would vary based on how many test cases your algorithm gives correct answer to.

To be properly tested, every submission must compile correctly as submitted, and must be based on the provided template. If your submission does not compile for any reason (even trivial mistakes like typos), or was not based on the template, it will receive at most 10 out of 50. The best way to make sure your submission is correct is to download it from conneX after submitting and test it.

You are not permitted to revise your submission after the due date, and late submissions will not be accepted, so you should ensure that you have submitted the correct version of your code before the due date. conneX will allow you to change your submission before the due date if you notice a mistake. After submitting your assignment, conneX will automatically send you a confirmation email. If you do not receive such an email, your submission was not received. If you have problems with the submission process, send an email to the instructor before the due date.