Department of Computer Science
University of the Western Cape

Module:	CSC211 (2025)	Assessment:	Term 2 Practical 2
Lecturer:	Mr. C. K. Baker	Total:	50 marks
Duration:	Mon 2pm – Fri 2pm		

Instructions

The set of natural numbers (ℕ) contains an infinite amount of interesting mathematical properties. For this practical, you are required to implement Binary Search Trees (BST) to efficiently manipulate sets of natural numbers.

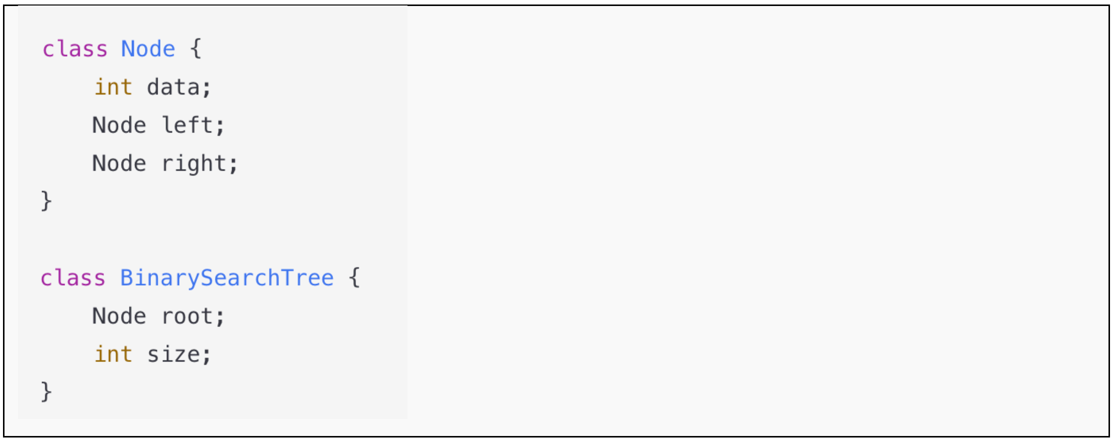
There are 3 questions. Submit a compressed (.zip) file with all your code. Use Java comments to acknowledge sources consulted. The submission file should be named XXYYZZZ. zip where XXYYZZZ corresponds to your student number.

N.B.

- work incrementally and systematically
- follow the marking rubric for mark allocation per question
- the data to test your work is provided in a text file called nat-nums.txt
- check the format of the example output
- save and submit a working draft on iKamva once you have completed a question
- ensure all files are correctly named and submitted by Friday at 14:00
- a penalty will apply for late submission
- a penalty will apply if you are not able to explain your code

Question 1 – setting up [15 marks]

Study the class definitions below.



```
class Node {
    int data;
    Node left;
    Node right;
}

class BinarySearchTree {
    Node root;
    int size;
}
```

1.1 Create two files: Node.java and BinarySearchTree.java

1.2 Copy the above class definitions into their corresponding files.

1.3 Write the following constructors for the Node class:

- default constructor
- loaded constructor

1.4 Write the following constructors for the BinarySearchTree class:

- default constructor
- loaded constructor

1.5 Use encapsulation to hide attributes of the Node class. Then, write accessor (getter) and mutator (setter) methods for the attributes of the Node class.

1.6 Use encapsulation to hide attributes of the BinarySearchTree class. Then, write accessor (getter) and mutator (setter) methods for the attributes of the BinarySearchTree class.

Question 2 – building the BST [15 marks]

2.1 Implement the search(int id) method:

- The method should return true if the node with the given ID exists in the tree; otherwise, return false

2.2 Implement the insert(Node n) method:

- insert a new node into the tree, maintaining the binary search tree property
- update the size attribute after insertion

2.3 Implement the delete(int id) method:

- delete the node with the specified id from the tree, ensuring the binary search tree property is maintained
- use in-order predecessor for Case 4 deletion
- update the size attribute after deletion

Question 3 – application [20 marks]

Create a file named Main.java in which to write code for the following.

3.1 Write code to read and build BSTs from data in the file <nat-nums.txt>

- create a new BST for each line
- node values are separated by commas
- transform the raw data into Node objects
- build BSTs using insert(Node n)

3.2 Implement the find_kth_smallest( ) method

- perform an in-order traversal of each BST and keep track of the number of nodes visited
- when the k^{th} node is reached, print out that node's value
- if k is greater than BST size, print out “Input not valid”

Data

nat-nums.txt

```
1, 7, 4, 10, 3, 9, 11, #6
5, 8, 6, 3, 2, 4, 11, 14, #2
16, 11, 14, 9, 2, 17, 20, #5
10, 6, 12, 8, 4, 15, 7, #4
5, 1, 8, 7, 6, 13, 2, 11, #9
4, 9, 2, 6, 3, 7, 10, #4
1, 5, 7, 3, 6, 11, 8, 12, #6
2, 13, 5, 8, 11, 14, 16, 20, #
1, 4, 10, 6, 7, 3, 18, 20, #3
9, 7, 5, 12, 3, 17, 2, 19, 8, #2
```

Example output

```
BST 1: k = 6 and 6th smallest node = 9
BST 2: …
…
etc.
```

Total: 50 marks