CMPT 202
Quiz 3

April 11, 2019

Name:

Show your work. Proving $f(n)$ has big O complexity $g(n)$ means that $Kg(n)$ is always greater than $f(n)$ from some value n_0 onwards. K is usually a big positive number.

1. What is the maximum number of nodes a complete binary can have if its height is 2? (A tree of height 2 has three levels.) How many in a tree of height 8? (You may leave your answer in the form X^Y)

2. Prove that $f(n) = n - 1$ has big O complexity $O(n)$.

3. Prove that $f(n) = n$ has big O complexity $O(n^2)$

4. *Circle the correct answer:* The big O complexity of binary search on an array is: O(1) O(log n) O(n) O(n^2) other .
   If you circled *other*, write your answer here:

5. In a binary search tree, the root node contains 5 as data. What possible values can the right subtree contain?

6. Write an example of a recursive method below. Label the base case and the recursive step(s).

7. Complete the method below if necessary. The method should perform an in-order traversal.
   You may assume that the node class has public leftChild and rightChild instance variables, and that these instance variables are of type Node.

```
public static void traverse(Node root) {
        if(root == null) {
                return;
        }
        else {


                System.out.println("Visited: " + root.content);


        }
}
```

8. Select the data structure that has the fastest lookup time, assuming that they all contain the same amount of data :

   (a) A complete binary search tree

   (b) A binary search tree that is not complete

   (c) A hash table with frequent collisions

   (d) A hash table with no collisions