COSC 320 - Advanced Data Structures and Algorithm Analysis
Lab 8

Dr. Joe Anderson

Due: 31 October 2019

1 Objectives

In this lab you will focus on the following objectives:

1. Review basic binary tree data structure and operations
2. Develop familiarity with list-concepts and memory management in `c++`

2 Tasks

1. Put your code in a folder called "Lab-8". This folder will be zipped and turned in at the end.
2. Write a class called `BinaryTree` that generalizes a linked list, with the following basic structure:

```
class BinaryTree{
  private:
    struct TreeNode {
      int key;
      TreeNode* left;
      TreeNode* right;
      TreeNode* parent;
    };

    TreeNode* root;

  public:
    /* Fill in with methods */
};
```

3. Implement the `transplant` method as a private method, to be used in the `delete` method.
4. Implement the following public methods. You may include other private methods to carry out the "standard" recursive behavior, such as inorder traversal.
   (a) `insert`, to add a new key to the tree.
   (b) `search`, which, given a key, determines whether there is a node with that key in the tree.
   (c) `minimum`, which returns the smallest key in the tree.
   (d) `maximum`, which returns the largest key in the tree.

(e) `successor`, for node with key k, returns the smallest key in the tree larger than k.

(f) `inorder`, which prints the keys in the tree in ascending order.

(g) `delete`, which removes a given key from the tree, if it exists.

(h) `print`, to display the contents of the tree, in a format of your choice, as long as it is clear.

5. Write a test program to demonstrate (clearly) the correctness of each of the above methods, displaying the tree after modifications.

6. Include a `Makefile` to build your code.

7. Include a `README` file to document your code, any interesting design choices you made, and **answer the following questions completely and thoroughly**:

   (a) Summarize your approach to the problem, and how your code addresses the abstractions needed.

   (b) What is the theoretical time complexity of your algorithms (best and worst case), in terms of the size of the tree? Be sure to vary the parameters enough to use the observations to answer the next questions!

   (c) Use timing tools to study the cost of each of the data structure algorithms. For example, time how long it takes to insert, find, delete 1000, 2000, 3000, etc. elements and calculate the time per operation. Look for how the time scales with the size of the tree. Does the data align with the theoretical guarantees?

   (d) How could the code be improved in terms of usability, efficiency, and robustness?

3 Submission

All submitted labs must compile with your provided `Makefile` and run on the COSC Linux environment.

Upload your project files to MyClasses in a single `.zip` file.

Turn in (stapled) printouts of your source code, properly commented and formatted with your name, course number, and complete description of the code.

Also turn in printouts reflecting several different runs of your program (you can copy/past from the terminal output window). Be sure to test different situations, show how the program handles erroneous input and different edge cases.