COP 3530 Data Structure and Algorithm Analysis
Homework 4

Feng-Hao Liu

In this assignment, you are given several classes in the cpp file “BTree.cpp”. Your task is to complete the implementation of the classes specified as below. You need to submit the following two files:

- A *single* cpp file that contains everything about your source code. It must be compilable and executable. Do not submit things irrelevant (such as .exe).
- A *single* pdf file that describes your methods and analysis.

You can discuss with anyone, but you should **write your own** code and writeup. In the pdf document, you **must** mention/acknowledge all people that you have discussed with. For example, you can say Idea A was from the discussion with Person B. Details are described below.

1 Your Task

You are given a class “TNode” that contains one integer value, and three pointers – one to the parent, one to the left child, and one to the right child. You need to complete the class “BTree” and two other functions specified in the cpp file.

Task 1: Write a function isValidBT that given inputs as a binary tree in the array form, and outputs whether the array forms a correct binary tree. The inputs contain a pointer p to an array, and an integer n denoting the number of slots used in the array representation. Here the array has size n, and we use $p[0]$ to store the empty symbol for the binary tree as discussed in the class.

Task 2: Implement the constructors (default and copy) of DLinkedList, and the destructor. You need to make sure that the copy constructor makes a separate copy of the list.

In addition to the normal copy constructor, here we ask you to implement a special “copy constructor” that takes input a binary tree of the array form. The inputs have the same format as Task 1, and you need to copy the binary tree, and convert the array form to the linked-list.

Task 3: implement the function convertpos that takes input an integer position, and returns the TNode pointer that points to the position-th node in the tree. We discussed about how to do this in the class.

Task 4: Implement the add2left, add2right functions. The functionalities are just as the names. Here you need to consider how to handle different input of positions, i.e. places that you want to add the node/tree to. Task 2 can be useful in this case.

Task 5: Write the removeleaf function. You need to first check whether the input is a leaf or not. If that is a leaf, then remove it. Otherwise do nothing.

Task 6: Write a function Toarray that converts the BTree into the array form. You can start by determining the size to the reference n, and new an array of size n. Then you need to produce the correct array according to your BTree.

Task 7: Write swapnodes and the three tree traversal algorithms. For the traversal algorithm, you can just print the notes according to the order. Here you just need to swap the values for the swapnodes. This task should be very simple.

Task 8: Write a test function, and a short report about your implementation of the above tasks.