CSE-310 : Data Structures and Algorithms

Prof. Andréa W. Richa
Homework 5: Red-Black Trees
Hard copy *due at the beginning of class on Tuesday, March 13, 2012*
Electronic copy *due by 10:00am on Tuesday, March 13, 2012 on Blackboard*

This is a programming assignment. You are asked to implement some of the operations related to red-black trees. you *must* use either **C or C++** for this programing assignment. Although you may develop your code on your personal computer, you have to make sure that it compiles and runs correctly on **general.asu.edu**. You need to submit this assignment both electronically using the *drop-box* on my.asu.edu, and in *hard copy.* We will grade your assignment using the soft copy you submitted, while mark the comments and grades on the hard copy.
The following are some detailed specifications:

1. You should implement a data structure for tree node with data field of type **int**; and color field of type int, taking values of either **0** (for red) or **1** (for black). It should also have the fields for **parent**, **lchild** and **rchild**.
2. You should implement a function named **RBread** to read in a red-black tree from the input file that is stored in ***pre-order*** format (see below for details).
3. You should implement a function named **RBwrite** to write out the current red-black tree to the output file, where the tree is stored in a ***pre-order*** format.
4. You should implement a function named **RBinsert** to insert a new node with the data field as a parameter and perform the fix-up if necessary.
5. You should implement a main function which takes the name of the input file from command prompt. The program should then read the initial tree configuration from the input file and then read the subsequent insert operations and apply them to the tree. The resulting tree should be printed in an output file by the name "output.txt".

In the pre-order representation of a red-black tree, we list the data-bearing tree nodes in pre-order. For each data-bearing tree node, we list its color code (0 for red, 1 for black) and then its data, separated by white

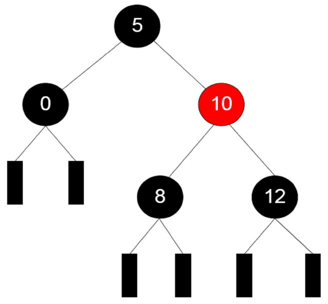

Figure 1: A sample RB tree

space. Use a separator such as white space or a new line character after each node.

A sample red-black tree is illustrated in Figure 1. For that tree, the pre-order representation is the following:

```
1 5
1 0
0 10
1 8
1 12
```

Format of the Input File: On the first line of the input file will be a single integer N that represents the number of data-bearing nodes in the initial red-black tree. The following N lines will each contain two integers separated by a space character which are the pre-order representation of initial red-black tree as shown in the example. A 0 represents the color red and 1 represents the color black. After that there is a single integer C on the next line which represents the number of additional nodes that need to be inserted into the red-black tree. The subsequent C lines will each contain a single integer for each element that should be inserted into the red-black tree.

Format of the Output File: This file should contain every red-black tree after each insertion is performed, separated by a blank line, and represented in pre-order form. For a red-black tree with N+C data-bearing nodes it should contain N+C lines each containing two integer values separated by a blank space. The first integer is the color code of the node (0:red 1:black) and the next integer is the data.

You should also prepare a README.txt file which contains the following information:

- Your name
- The language you are using
- How to compile the program

In order for you to test whether your program is working correctly, we have provided a sample input and output file on the blackboard, under the folder for Hw5. We recommend testing your program on a few other test cases as well before submitting the final version.