CSE225 Data Structures, 2014-2015 (FALL),PROJECT #2
Due November 24, 2014, MONDAY, 01:00 a.m.

Project Demos: 26,27,28 November 2014

This project is a programming assignment in C which builds a review-score list based on the user's reviews collected from WWW. These reviews are in some basic categories according to the scores they have. This program will let the user add, remove, list the review-score list.

The data structure that you use is as detailed below:

1. The review-score number,
2. The number of the total reviews on that review-score list

3. A list of the reviews in that review-score. For each review, you should keep:

    - The id of the review
    - The text of the review

In your implementation you MUST keep the score entries in a binary search tree by their score number. Then, to keep the reviews of each score, you should create a binary search tree in which the reviews are kept by their id s (e.g., if you have five scores in your binary search tree, you will have five more binary search trees: one for the review list of each score.). You may define your binary search tree such that for a node, smaller valued items are kept in the left subtree of the node and greater or equal valued items are kept in its right subtree.

Your system will have the following functionalities; the details of these functionalities are given below:

1. Add a new score node into the master-BST .
2. Add also some reviews into a specific review-score category.
3. Show the average score of all of the reviews.
4. List the id s of reviews which have a particular word.
5. Remove the reviews between in a specific score-scope.
6. Display the tree.

The Details of These Functionalities:

1. This function adds a review-score category to the master-BST whose score-number is specified as parameter. In this function, the review BST is not specified; the reviews BST under that score will be added later. In this system, the numbers of the scores are unique. Thus, if the user attempts to add a score-node into the master-BST with an existing score-number, you should not allow this operation and give a warning message.

2. This function adds a review to the review BST of a score-number node. For that, the score number to which the review is added, the id of the review and the text of the review are specified as parameters. In this function, you should consider the following issues:
• If the score with a specified number does not exist in the master-BST, you should not allow the operation and give a warning message.
• In this system, all id s of the reviews are unique. Thus, if the user attempts to add a review with an existing id, you should not perform the operation and give a warning message.

3. This function shows the average score of all of the reviews. You need to traverse all score nodes including their review-BST and calculate the average as follows:

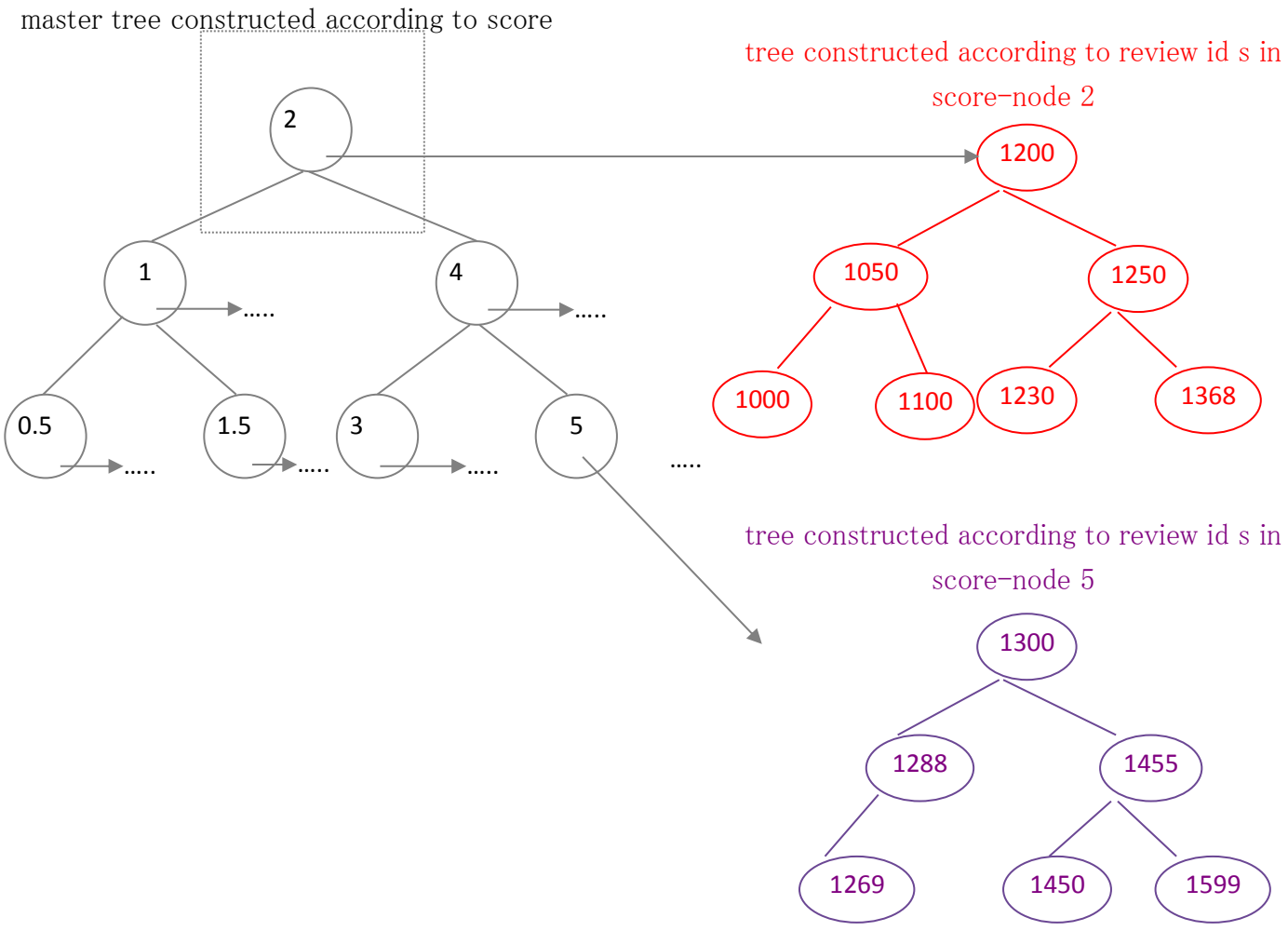
$$S_{mean} = \frac{R}{N}$$

where R is the total review-score calculated from all of the reviews from all of the review-score numbers, and N is the total number of reviews.

4. In this function you should list all reviews with a specified word. The word is specified as parameter by the user. The output should include the id s of the reviews. Note that if there is no such a review, you should display ---none--- .

5. This function removes the specified score node including all of its reviews from the master-BST, whose score-number is specified as a parameter.(Actually the user needs to enter a scope including starting score and ending score that means all of the scores including their reviews in that scope will be deleted).For instance if the user enters 1 and 3 this means that the program is expected to delete all the scores (including 1 and 3) and also their corresponding reviews.
6. This function will display the master-BST and the corresponding review-BST s in a clearly&suitable format in the terminal.

SCHEMA OF THE MASTER-BST AND THE BST OF REVIEWS :



In your demo, we will run your program by **selecting each of above function-options with a different set of input files.** We want to see if they are **working correctly or not**, **and your demo grade will be calculated based on number of your functions which are working correctly.**

Of course, other questions based on your implementation and coding structure will be asked you. These questions will be those kinds of questions which could be answered by only the students who really implement his/her project.

If your file could not be compiled then you will get zero, unfortunately.

The main goal of this project is to be familiar with BST. So, if you use any other data structure instead of BST then you will get zero, unfortunately.

In this project you are expected to develop an algorithm that is capable of finding a solution to the above problem and ***implement this algorithm in ANSI C that runs under UNIX***. (You can also use cygwin or wagrant in order to compile and run your codes.)

You are responsible for demonstrating your program to your TA Berna Altinel on the scheduled day.

CODE SUBMISSION:

You should use the following email address in order to submit your code:
datastr.mufe at gmail dot com

NAMING STANDARD IS:

name_surname_2.c
for instance:
burak_yilmazlar_2.c

Your any submission after the project submission due date, will not taken into consideration.

You are required to exhibit an ***individual effort*** on this project. Any potential violation of this rule will lead everyone involved to **failing from the course** and necessary disciplinary actions will be taken.

Good luck!!!