CS2383 – Fall 2024

Assignment 6 – Symbol tables, Binary Search Trees and Balanced Trees

Due: Thursday Nov. 21, 10am (class time)

IMPORTANT: individual work please!

Tasks:

1. For each of the data structures below (as a symbol table), draw them after inserting each of the following keys in that sequence (starting with an empty structure): 41, 25, 10, 60, 33, 75, 5, 50, 2, 7, 28, 80, 45, 6, and 31.

    a) Sorted array
    b) Binary Search Tree
    c) 2-3 tree
    d) Red-Black tree

2. Implement a program for taking a list of marks as percentages (as int), return the corresponding letter grade for each, and calculate an average GPA for that list. Your program should first request the lowest percentage for each of the grade point scale. Here is what the input/output should look like (format must be exactly like this…underlined text is what the user enters):

```
Min percentage for an A+: 90
Min percentage for an A: 85
Min percentage for an A-: 80
Min percentage for a B+: 75
Min percentage for a B: 70
Min percentage for a B-: 65
Min percentage for a C+: 58
Min percentage for a C: 50
Min percentage for a D: 45
Min percentage for a F: 0
Give the list of percentages (separated by a space):
47  100  90  89  55  70  69
The corresponding letter grades are:
D  A+  A+  A  C  B  B-
And the average GPA is: 3.042857142857143
```

    You must use a TreeMap for doing this (see Java API). You are not allowed to put more than one entry per letter grade into it. The values in the TreeMap must be used in doing the calculations on the list of percentages – you are not allowed to use an "if" statement (or any other conditional statement) in your code. Store all of the following fields in the TreeMap, for each possible letter grade: the letter grade (e.g., "A+", "A", "A-", "B+", …, "C", "D", "F"), the corresponding GPA value (e.g., 4.3, 4.0, 3.7, 3.3, …, 2.0, 1.0, 0.0), and the lowest percentage to get in order to have that grade, as specified by the user (see example above). Note: of course, you can package all fields in an object. Assignments that do not follow the requirements in this paragraph will not get any mark.

Please put all your code in a single file for simplicity. It is OK to have most of the code in the main class, but please use at least 2 different methods (can be static ones, with some static variables) to help better understand your code: one method to set up the TreeMap with the user information, and one method for the computations on the list of percentages.

3. Do a theoretical runtime analysis of your code in Question #2 above (i.e., θ(…)), in terms of m=number of possible letter grades and n=number of percentages in the list. Explain your analysis in detail.

Submission: submit everything on paper. Also upload your Java file (Question #2) in D2L.