INDIAN STATISTICAL INSTITUTE

Laboratory Test III

M. Tech (CS) - I Year, 2015-2016 (Semester - I)

Data and File Structures Laboratory

Date: 21.09.2015 Total Marks: 130 + 20 = 150 (20 marks for good programming habits)

Note: Follow the file naming convention strictly as mentioned.

You are not allowed to connect and browse the internet during the test. No books and e-books are allowed. You can reuse your own code.

Any instance of malpractice would be dealt with sternly. If you are in doubt about whether your action is improper, better clarify.

(Q1) Write multifile programs and create *makefiles* to compile your programs and submit them along with your programs. We would 'make' your makefiles to compile your programs. [10+10=20]

(Q2) Write a C program (Name your program `labtest3-prob1-cs15XX.c` where `XX` is your roll number) that given two strings $X = [x_1, x_2, \ldots, x_n]$ and $Y = [y_1, y_2, \ldots, y_m]$, finds the length of the *shortest common supersequence* (SCS) of X and Y using dynamic programming.

The *shortest common supersequence* (SCS) is a minimum length string Z such that both X and Y are subsequences of Z. For example, if $X = [abcbdab]$ and $Y = [bdcaba]$, a SCS is $Z = [abdcabdab]$.

Total marks: [30]

(Q3) Write a C program (Name your program `labtest3-prob2-cs15XX.c` where `XX` is your roll number) that

(a) takes as input an integer n from the user and allocates a dynamic array $\mathcal{A}$ of size n and reads data into it.

(b) sorts the data in $\mathcal{A}$.

(c) Construct a binary tree T using linked structures for a set $S = \{x_1, \ldots, x_n\}$ ($x_i \in \mathbb{R}$) of n reals in the following way:

(i) Compute the median x_{mid} of S and put it into the root of T.

(ii) Partition S into two parts S_L and S_H where $S_L = \{x_i : x_i \in S \text{ and } x_i \leq x_{\text{mid}}\}$ and $S_H = \{x_i : x_i \in S \text{ and } x_i > x_{\text{mid}}\}$

(iii) Compute the median of S_L and put it into the root of the left subtree of T.

(iv) Compute the median of S_H and put it into the root of the right subtree of T.

Continue the above steps recursively until the partitioned sets contain only one element. As an example, for $n = 7$ and $S = \{5, 10, 7, 20, 25, -10, 15\}$, the tree T is shown in Figure 1.

(d) prints the tree in a *proper fashion* on the terminal using recursion. The idea of *proper fashion* will be explained during the labtest hours.

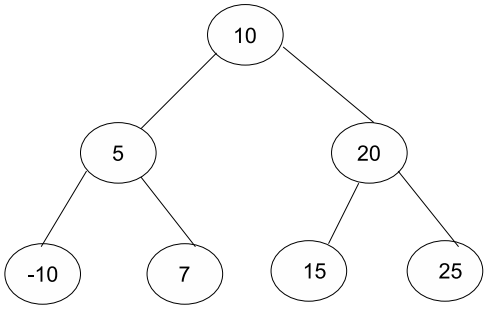

Figure 1: The tree T.

(e) searches the tree T, preferably using recursion, and finds out efficiently the elements in T that lie within an interval $[x, x']$ and whose level is any element of the set $\{h, h-2, h-4, \ldots\}$, i.e. alternate levels starting from the leaf at the highest level, h. For the given example in Figure 1, if the interval within which to search is $[4, 21]$, then your answer should be $\{7, 15, 10\}$.

If you search the entire tree using $O(n)$ time, then the program is inefficient. You should aim to solve the problem in $O(\log n + \ell)$, where ℓ is the number of elements of T lying within the interval.

[**Hints:** First try to find a node in T with value x_1, such that $x \leq x_1 \leq x'$, using binary search. Search recursively in the left subtree $T_L(x_1)$ of the node having x_1 with the range $[x, x_1]$ and in the right subtree $T_R(x_1)$ of the node having x_1 with the range $[x_1, x']$.

Now, let y_1 be value of a node in $T_L(x_1)$ and your search path goes left from y_1. Can you say anything about the values in $T_R(y_1)$? Similarly, let y_2 be value of a node in $T_R(x_1)$ and your search path goes right from y_2. Can you say anything about the values in $T_L(y_2)$? See Figure 2 for an idea.]

Total marks: [5+10+35+30=80]

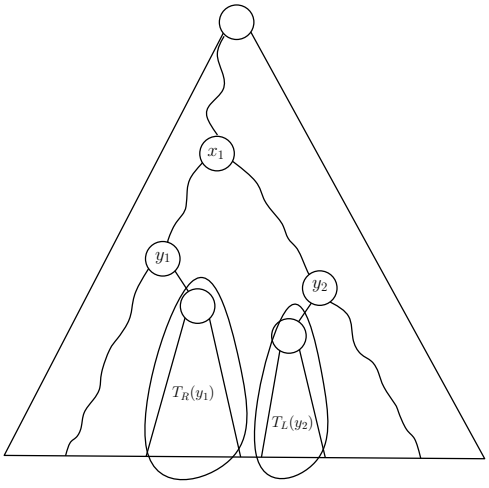

Figure 2: The search technique.

Submission instruction: Submit all your source codes to the directory `labtest3` under the home directory of the user `pdslab`. For example, if you are copying a file `labtest3-prob1-cs15XX.c`, you should use the command

```
cp labtest3-prob1-cs15XX.c ~pdslab/labtest3/
```