BITS Pilani, Pilani Campus
2nd Semester 2017-18
CS F211 Data Structures & Algorithms
Lab Test II - (15th April, 2018)

Maximum Marks: 20 + 40 = 60 **Duration: 160 + 10 = 170 min**

Instructions

- In this lab test you have to solve two problems given in two different files. Total time for solving the problems and uploading solutions is 160 minutes with at most 10 additional minutes for ensuring final uploading of the solution on the portal. Portal closing timer is shown at top-right corner of your page.
- You may make multiple submissions on the online portal, but only the latest submission shall be considered for evaluation. It is your responsibility to save and upload files at regular intervals.
- For each of the given two problems, upload all relevant source code (*.c) and header (*.h) files as a single submission. Do not upload any other file.
- If your program is having a "*compilation error*" or "*segmentation fault*" or other similar errors, we shall **evaluate your code for 50% of the marks only**. So, try to keep your code as clean as possible. Also note that **commented code** will **NOT** be considered for evaluation. Codes not compiling will have higher penalty as compared to codes with segmentation fault.
- Each problem is associated with a zip file which contains pre-filled code for solving the problem:
  - Write your code only in the designated areas in each of the file.
  - Do NOT rename any file, or create a new file.
  - Please note that each function may be tested separately for various test cases. So, don't make changes to the signatures and return types of the pre-defined functions.

Problem 1 of 2

[Expected Time: 45 minutes.] **[20 Marks]**

a) Given an array `list` with `p` non-negative integers construct a binary tree `BT` using the following procedure:

- Let the first element of `list` be the root value of `BT`.
- Now, make a random choice.
  - With probability 1/3, the left subtree of `BT` is empty, so the right subtree contains all of the remaining `(p-1)` values.
  - With probability 1/3, the right subtree is empty, that is, the left subtree contains the remaining `(p-1)` values.
  - Finally, with probability 1/3, both the subtrees are non-empty. In this case, each subtree consists of a half of the remaining `(p -1)` values: the left subtree will be the first half, and the right subtree will be the second half.
- Recursively, construct the left and/or right subtrees, as necessary.

b) Iteratively, traverse the constructed binary tree that reproduces the values in the same order as `list`.

Complete definition of following functions in `tree.c`. Write any additional functions and data structures in `extras.c` and `extras.h`.

a) `unsigned int* createList (unsigned int p)`
Use rand() to generate a list of p unsigned integers.

b) `tree constructTree (unsigned int *list, unsigned int len)`
Return a binary tree constructed using the approach given above, containing the elements in `list`.

c) `bool matchTreeIterative (tree BT, unsigned int *list, unsigned int p)`
Traverse `BT` (in appropriate order) such that tree nodes visited and their order match the entries in `list` in left-to-right order.

Solution Upload Instruction:

`driver.c`	**Do not edit.**	Upload on the portal by browsing and selecting all of these together.
`tree.c`	Fill in function definitions for (a), (b), and (c).	
`tree.h`	**Do not edit.**	
`extras.c`	For any additional implementation.	
`extras.h`	For any additional interfacing.	

Sample Test Case:

Input:

```
15042018   20
```

Output:

```
1181067988 1989749871 1186522786 894433635  1997593533 1063641343
1927391309 875545658  939412149  724667906  1740471381 1205136578
1192847068 731170945  1211084342 74548284   299260362  659600975
1817776658 1087924507
1181067988 894433635  1997593533 875545658  1927391309 939412149
1063641343 1740471381 724667906  1186522786 1989749871 1192847068
74548284   1211084342 299260362  731170945  1817776658 659600975
1087924507 1205136578
CORRECT
```