Algorithms and data structures

Tutorial 6: Data Structure: Trees

Dr. Doan Nhat Quang: doan-nhat.quang@usth.edu.vn

- You will have one week (or 7 days) to complete the remaining exercises and upload your files to the "Labwork X Group Y" assignment in Google Classroom. (e.g. Labwork 1 Group 1) if you are from Group Y and want to submit labwork X
- Compress all code source files in a zip file and rename it as FULLNAME-ID-Lab#no.zip (e.g NguyenVanA-BI10-070-Lab1.zip). Save your files according to the exercise number i.e Ex1.cpp, Ex2.c, etc. Incorrect filenames will result in no score for the respective exercises.
- Only code source files (.c or .cpp) should be in the zip files. Other files (.exe, .o) MUST be removed from the zip file.
- - Copy/Paste from any source is not tolerated. Penalty will be applied for late submissions.

NOTE: You must follow the guide. Incorrect zip file names, zip files containing other files (.exe), and copy/paste lead to heavy penalties.

Exercise 1:

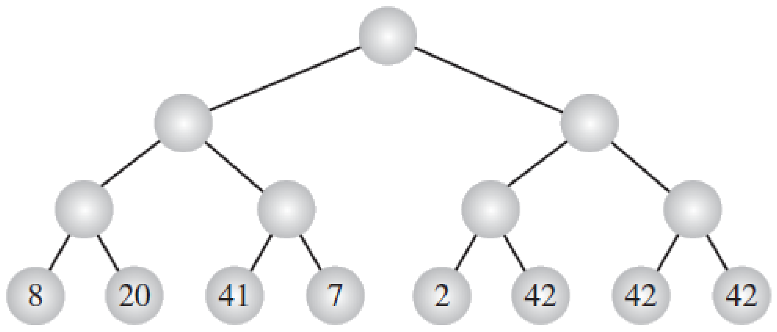
A binary tree can sort n elements of an array of data. First, create a complete binary tree with all leaves at one level, whose height h = (log_2 n) + 1, and store all array elements in the first n leaves. In each empty leaf, store an element E greater than any element in the array.

Figure (a) shows an example for data = {8, 20, 41, 7, 2}, h = ($log_2(5)$) + 1 = 3, and E = 42. Then, starting from the bottom of the tree, assign the minimum of its two children values to each node, as in Figure (b), so that the smallest element e_{min} in the tree is assigned to the root.

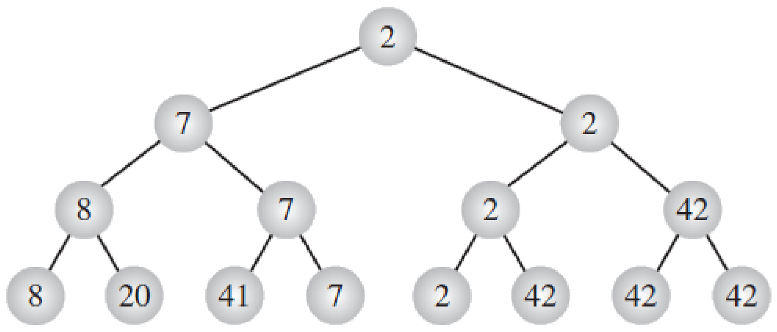
If a leaf node is to be removed, this node is replaced by a new node with the same value as its parent node.

If a node is added to the tree, it will be a leaf node. Normally, a node with value E is replaced with a new value. It's necessary to verify recursively all values of its parent and make any possible modification, if necessary, so that the tree rules are respected.

Implement this tree structure in C/C++ with the necessary functions.



(a)



(b)

- write a function to initialize an array with n random values
- write a function to build this binary tree with the above definition with any data structure learned in lectures
- write a function to display the tree information
- write a function to search an input value using recursion. If found, display all the subtrees with the found node as the root of this subtree or return -1.
- write a function to insert new nodes into the tree and another one to remove nodes from the tree