University of Science and Technology of Hanoi
Address: USTH Building, 18 Hoang Quoc
Viet, Cau Giay, Hanoi

Algorithms and data structures

Tutorial 6: Data Structure: Trees

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

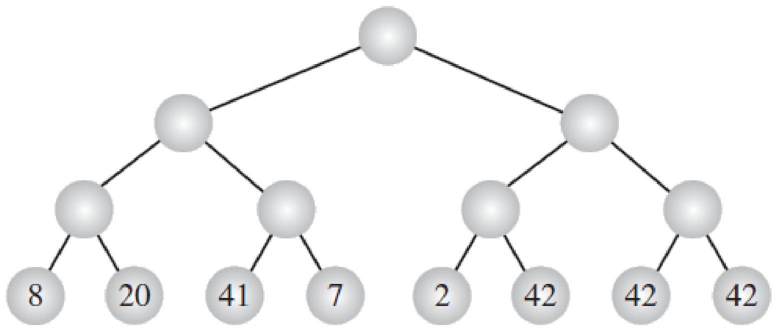
Please use Google classroom for labwork submission; join the class by the following link: https://classroom.google.com/c/NTQ1NjMwOTMwNzU0?cjc=gqtgfva

There are 6 Labworks in this course. After each lab work session:

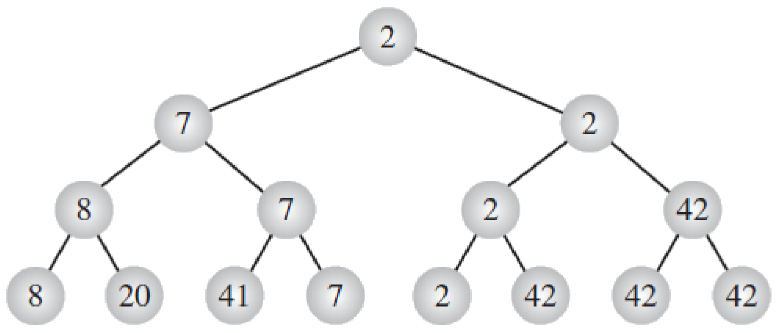
- You need to complete one of the given exercises and upload your files to the assignment "Labwork No. - Group No. - Version 1 (Attendance)". Submission must be done within 30 mins after the lab; otherwise, it will be considered a late submission.
- You will have one week (or 7 days) to complete the remaining exercises and upload your files to the assignment "Labwork No. - Group No. - Version 2 (Complete)"
- Compress all code source files in a zip file and rename it as FULLNAME-ID-Lab#no.zip (e.g NguyenVanA-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. The penalty will be applied for a late submission.

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

Exercise 1:



(a)



(b)

A binary tree can be used to sort n elements of an array data. First, create a complete binary tree, a tree with all leaves at one level, whose height h = (lg n) + 1, and store all elements of the array 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 = (lg(5)) + 1 = 4, and E = 42. Then, starting from the bottom of the tree, assign to each node the minimum of its two children values, 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 of its parent node.

If a node is added into the tree, it will be a leaf node. Normally a node with value E is replaced with 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.

- 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 learnt 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 subtree with the found node as the root of this subtree' or else return -1.
- write a function to insert new nodes into the tree and another one to remove nodes from the tree