Introduction to Computer Science
HW #5
Due: 2016/05/25

Homework Rules:

Hand-written homework can be handed in **before lecture starts**. Otherwise, you may contact the TA in advance and then bring the hardcopy to the TA in MD-631 (please send e-mail in advance).

As for the programming part, you need to upload it to CEIBA before the deadline. The file you upload must be a **.zip** file that contains the following files:

README.txt

HW01_b04901XXX (a folder that contains all .cpp & .h as required),

1. Do not submit executable files (.exe) or objective files (.o, .obj). Files with names in wrong format will not be graded. You must **remove any system calls**, such as system ("pause"), in your code if any.
2. In README.txt, you need to describe which compiler you used in this homework and how to compile it (if it is in a “project” form).
3. In your .cpp files, we suggest you write comments as detailed as you can. If your code does not work properly, code with comments earns you more partial credits.

Chapter 8 Review Problems (8% each):

23, 24, 28, 29, 34

M1 (5%): Draw a binary search tree (with keys only) which pre-order traversal yields 4, 3, 2, 1, 10, 9, 8, 5, 7, 6, 11, 12.

M2 (5%): What’s the post-order traversal sequence for the BST in **M1**?

Programming Problem (50%):

Write a class **Heap** which is derived from **AbsHeap** (in “heap.h”). Implement the two virtual functions in AbsHeap. This is a binary min heap, so pop() always returns the element with the minimum key. You may assume **all keys are distinct**. You need to complete your code in “heap.h”.

For your test convenience, you may use testHeap.cpp to test your heap. To use them, rename “inputX.txt” to “input.txt”, and check the corresponding “outputX.txt”.

BONUS (5%)

Given two arrays A[] and B[] with *n* elements (both). Your task is to output *n* smallest sums of two elements, and each of which comes from different arrays. For example, if A[] = {0.1, 0.4, 0.9, 0.7}, and B[]={0.3, 0.2, 0.6, 0.5}, your output (standard out) should be 0.3, 0.4, 0.6, 0.6 (separated by “ “). To be more specific, 0.3 comes from 0.1+0.2, 0.4 comes from 0.1+0.3, 0.6 comes from 0.2+0.4, 0.6 comes from 0.1+0.5. Your code reads inputs from a file “bonus.txt”, where the 1st line is *n*, the 2nd line is A array with each element separated by “ “, and the 3rd line is B array. Save your code in “bonus.cpp”.

Your code needs to handle *n*=100000 in reasonable time (couple seconds).

How to submit:

Compress all your files into one single file and then submit electronically via Ceiba by the due date.