American University of Armenia, CSE
CS 121 - Data Structures (B)
Homework Assignment 6 Fall 2018

Due Date: Sunday November 25 by 23:59 electronically on moodle

For all of the questions below, you are expected to use your own implementation of any data structures used in your solutions. You may use the implementation used by the course textbook (either Java or C++) or implement it by yourself. You may obtain these files from the textbook companion website (found in the initial pages of the textbook).

1. Given an array A of n integers, write a Java/C++ code that takes A and using a heap prints the first maximum $k < n$ elements in $O(k \lg n)$-time.

2. Write a method/function `insert` in Java/C++ that takes input a key k and a value v, and inserts the node with the respective $< key, value >$ pair into a binary search tree. Your method/function should be added to the `LinkedBinaryTree` class that you have previously implemented. You may change the nested Node class in `LinkedBinaryTree` so that instead of it taking an element value e in its constructor, it takes an `Entry<K, V>`.

3. Write a method/function `remove` in Java/C++ that takes input a key k, and removes the node with the respective key from the binary search tree. If the node X associated with key k is an internal node with two children, you should output "X is an internal node with two children". Otherwise, X must be removed. Your method/function should be added to the `LinkedBinaryTree` class that you have previously implemented. You may change the nested Node class in `LinkedBinaryTree` so that instead of it taking an element value e in its constructor, it takes an `Entry<K, V>`.

4. (a) How efficiently could an AVL tree implement the `insert(k,v)` and `removeMin()` methods of a priority queue? Justify your answer.

   (b) Dr. Amongus claims that the order in which a fixed set of entries is inserted into an AVL tree does not matter - the same AVL tree results every time. Is Dr. Amongus right? Justify your answer.