# Ve281 Data Structures and Algorithms
# Written Assignment Four

**This assignment is announced on Oct. 25th, 2018. It is due by 5:40 pm on Nov. 2nd, 2018. The assignment consists of four problems.**

1. (20%) For the following tree, show the order in which the nodes are visited during the following tree traversals (The nodes in the tree are from A to I):

   (a) Pre-order depth-first traversal.

   (b) Post-order depth-first traversal.

   (c) In-order depth-first traversal.

   (d) Level-order traversal.

```
        A
       / \
      /   \
     /     \
    /       \
   B         H
  / \       /
 /   \     /
C     E   I
 \   / \
  D F   G
```

2. (40%) In class, we showed a recursive way to realize in-order depth-first traversal of a binary tree. In this problem, we ask you to design a **nonrecursive** algorithm that performs in-order depth-first traversal. Assume the tree is stored using a linked structure with node as

```
struct node {
  int key;
  node *left, *right;
}
```

You can either describe your algorithm in plain English or write pseudo-code. If you choose to write pseudo-code, you should write in a way that can be easily understood. Otherwise, you will get a zero for the problem. (Hint: consider using a stack as an auxiliary data structure.)

3. (20%) Min Heap

   (a) (10%) Suppose that we are inserting the keys 3, 1, 4, 1, 5, 9, 2, 6, 5, 4 **one by one** into an initially empty min heap. Show the resulting min heap in the form of a tree.

   (b) (10%) For the min heap you obtained for Problem (3a), show the resulting heap after calling two `dequeueMin` operations.

4. (20%) Min heap initialization

   Given a sequence of keys 3, 9, 7, 2, 5, 2, 8, 6, 1, 4, show the resulting min heap if we initialize it with the efficient algorithm we talked in lecture that takes $O(n)$ time complexity on an array of $n$ elements. Show the intermediate steps in the form of a tree. Do not just write the final result.