CSE 330 – Winter 2020 – Selected Samples of Finals Questions

See answers below …

Q1: A vector with the following sequence of values is given: 15,21,8,11,27,3,16,19,7. Which is a true statement?

(a) It can represent a complete binary tree with leaves 8, 11, 3, 16, 7.
(b) It can represent a complete binary tree with leaves 19, 7, 27, 3, 16.
(c) This vector cannot represent any binary tree structure.
(d) This vector represents a binary tree, but not one that is complete.

… correct answer: (b)

Q3: A binary search tree (BST), representing a set data structure, tends to exhibit its lowest-cost behavior for finding, inserting and removing of a target values when

(a) The values stored in the set have been inserted in ascending or descending order of magnitude.
(b) The values stored in the set have been inserted in some random order.
(c) The values stored have been inserted so that the largest value has been added first, and the smallest last.
(d) The order in which values have been inserted into the set's underlying BST has no effect on the efficiency of the set operations.

… correct answer: (b)

Q5: When a new value is inserted into a priority queue structure, implemented as a "heap", what is true about the location of the new value within the structure?

(a) The new value will be the value of a leaf node.
(b) The new value will be the value of an internal node.
(c) The new value will become the new root value.
(d) All of the above are possible.

… correct answer: (d)

Q8. Given is the following variant of member function `insert` of class `Node<T>`:

```
void Node<T>::insert(T& x){
    if (x < value) {
       if (!leftChild) {
          Node<T> * newchild = new Node<T>(x,0,0,this);
          leftChild = newchild;
       }
       else
          leftChild->insert(x);
    }
    if (x > value) {
       if (!rightChild) {
          Node<T> * newchild = new Node<T>(x,0,0,this);
          rightChild = newchild;
       }
       else
          rightChild->insert(x);
    }

}
```

What will this function do when x is equal to the value stored in calling node?

(a) Nothing.
(b) Another copy of value x will be inserted into the data structure.
(c) The function will recurse into an infinite loop.
(d) The code will break at run-time.

… correct answer: (a)

Q10. Imagine that our familiar `class Node<T>` has a mysterious member function `g()` which is defined as follows.

```
int Node<T>::g() {
   if (!leftChild and !rightChild) return 1;

   if (leftChild and !rightChild) return 1 + leftChild->g();
   if (!leftChild and rightChild) return 1 + rightChild->g();

   int lft = 1 + leftChild->g();
   int rgt = 1 + rightChild->g();

   if (lft >= rgt)
      return lft;
   else
      return rgt;
}
```

What does member function g() compute?

(a) The number of leaf nodes in a binary tree rooted in the calling Node.
(b) The total number of nodes in a binary tree rooted in the calling Node.
(c) The depth of the binary tree rooted in the calling Node.
(d) The number of left and right subtrees of the binary tree rooted in Node.

… correct answer: (c)