CS2010 Homework Tutorial 2

A0108165J

TONG Haowen Joel

1

```
             +
          /      \
         -         d
       /   \
    +       *
   /  \   /   \
  /    a  b      c
 /  \
x   y
```

2a

Encoding	Letter
0110	I
111	
10111	L
1001	O
1100000	V
010	E
111	
10111	L
0001	U
00000	C
100011	Y

2b

Encoding	Letter
1101	t
0010	r
0010	r
100010	g

3

a. 4 b. no, it is not full c. yes, it is complete d. no, it is not a binary search tree

```
              8
          /        \
        7         10
      /           /   \
    5          9     11
  /   \
1     6
```

4

- In-order: 1,5,6,7,8,9,10,11
- Pre-order: 8,7,5,1,6,10,9,11
- Post-order: 1,6,5,7,9,11,10,8

5

Please see folder qn 5.

```
    public String postorderToString() {
      StringBuilder sBuilder = new StringBuilder();
      postOrderRecursive(sBuilder, this.root);
      return sBuilder.toString();
    }

    private void postOrderRecursive(StringBuilder sBuilder, Node<E> node) {
      if (node.left != null) {
        postOrderRecursive(sBuilder, node.left);
        sBuilder.append(" ");
      }

      if (node.right!= null) {
        postOrderRecursive(sBuilder, node.right);
        sBuilder.append(" ");
      }

      sBuilder.append(node.toString());
    }
```

6

Creating a Hoffman Tree

1. Sort the list of chracters and frequencies by frequency.

2. Take the lowest 2 frequencies create a tree with the sum of them as root. For instance,

```
           A + B
           /  \
           A  B
```

3. Remove the 2 lowest frequencies from the list.

4. Now, the lowest frequency in list is C. Take C and repeat steps 2 to 3 with existing tree.
5. Repeat step 4 until done.

Traversing left appends a 0 to string; traversing right appends 1 to string.