CMSC 132 Quiz 3 Worksheet

The third quiz for the course will be on Wednesday, Oct 18, during your lab session. The following list provides more information about the quiz:

- The quiz will be a written quiz (no computer).
- Closed book, closed notes quiz.
- Answers must be neat and legible. We recommend that you use pencil and eraser.

The following exercises cover the material to be included in this quiz. Solutions to these exercises will not be provided, but you are welcome to discuss your solutions with the TA or instructor during office hours. **We strongly recommend you do not use Eclipse to write the code associated with these exercises.** Try to answer the exercises in a piece of paper and then use Eclipse to verify your solutions. This approach will better prepare you for the quiz.

A. **Huffman**

1. Consider the following Huffman tree.

   ![Huffman Tree Diagram]

   i. Decode the sequence “10010110”

   ii. Encode the string “CED”

2. Create a Huffman tree for the symbols A, B, C, and D which have the following frequencies:

   \[
   \begin{align*}
   \text{A} & : 9 \\
   \text{B} & : 4 \\
   \text{C} & : 2 \\
   \text{D} & : 8
   \end{align*}
   \]

3. Create a Huffman tree for the symbols A, B, C, D, and E which have the following frequencies:

   \[
   \begin{align*}
   \text{A} & : 9 \\
   \text{B} & : 2 \\
   \text{C} & : 2 \\
   \text{D} & : 8 \\
   \text{E} & : 8
   \end{align*}
   \]

4. In the previous problem, how many Huffman trees are possible assuming we always assign 0 to the right subtree and 1 to the left subtree?
B. **Heaps**

1. Draw the heaps (in tree representation) created after completing each of the following operations. Assume the heap has the smallest element at top.

   - Heap after inserting 9
   - Heap after inserting 40 in the previous heap
   - Heap after inserting 2 in the previous heap
   - Heap after inserting 7 in the previous heap
   - Heap after inserting 10 in the previous heap
   - Heap after inserting 30 in the previous heap
   - Heap after deleting 2 from the previous heap

2. **MaxHeap**

   The MaxHeap class represents a max-heap. In a max-heap every element has a key value greater than or equal to either of its children. The class definition associated with the MaxHeap class is:

   ```java
   public class MaxHeap<E extends Comparable<E>> {
       ArrayList<E> data = new ArrayList<E>();
   }
   ```

   Complete the implementation of the MaxHeap class, by providing the following methods:
   a. `insert( )` – Inserts an element in the heap.
   b. `removeMax( )` – Removes the largest element from heap and returns that element.
   c. `size( )` – Returns the size of the heap.

C. **Binary Trees**

   The following Java class definition for a binary tree will be used to answer the questions that follow.

   ```java
   public class BinaryTree <E> {
       class Node<T> {
           public T data;
           public Node <T> left, right;

           public Node(T data) {
               this.data = data;
               left = null;
               right = null;
           }
       }
       Node <E> root;
   }
   ```

1. Define a constructor that creates an empty tree.
2. Define a recursive method `numLeaves( )` that returns the number of leaves in the tree.
3. Define a recursive method `isFull( )` that returns true if every interior node in the tree has both a left and right subtree.