Lab 13
CSC 1052 - Algorithms and Data Structures II
Grading: 30 points
Due Date: May 5th, 2017

Description: In this lab, you will practice using binary trees in a real-world application. Huffman encoding is a technique for encoding characters using a variable number of bits for each character. More frequent characters are encoded with fewer bits, resulting in a compression of the number of bits required to encode a complete file (this is one technique used in the PKZIP program to compress .zip files). A Huffman encoding of the alphabet {A, B, C, D, E} can be represented as a binary tree like the one shown in Firgure **??**, with the characters stored at the leaves. To find the encoding for any character, you simply follow the path from the root to the leaf containing that character. If you go to a left child, you add a "0" to the encoding. If you go to the right, you add a "1". For example, the encoding for the letter 'E' is "101" (see Figure **??**).

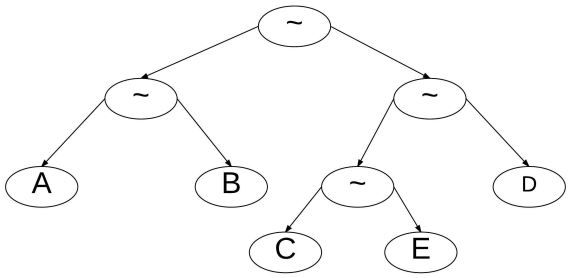

Figure 1: An example Huffman tree.

For this lab, your task is to complete the provided Huffman tree program by implementing the `decode()` method. Your method should read in the input string of 0's and 1's, and starting with the first digit, follow the specified path starting from the root. When you hit a leaf, print the stored character and re-start traversing from the root. You will know when you have correctly implemented the program when you decode the message encoded in the `main()` method.

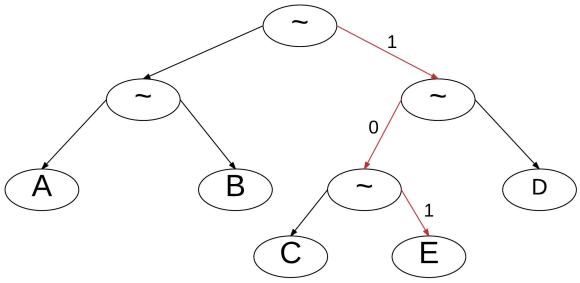

Figure 2: An encoding of the letter 'E'.

Use the template code provided and only modify the `decode()` method. When you are finished, submit the completed file `HuffmanTree.java`.

Rubric:
(5 points) Compiles without errors.
(25 points) Correctly implement the `decode()` method and recover the message.

Deliverables: Submit the lab deliverable on blackboard.