**Decoding Huffman codes**

In 1952 David Huffman discovered an optimal compression algorithm for data. The algorithm assigns binary codes to items of data and generates a unique coding scheme based on that data. After transmittal, the binary code and its coding scheme are decoded to get back the original message. Today's .zip files and .jpegs use variants of the Huffman compression scheme.

In this lab you will only decode a Huffman message. As an example, suppose the message is

01010111110111110000010011001001110011

p00

M010

0110

a0111

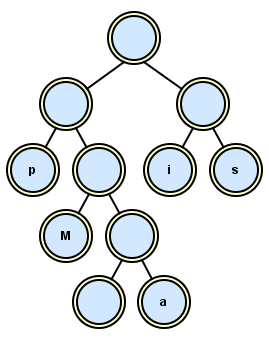
i10

s11

and its Huffman coding scheme is

Then the decoded message is:

\_\_\_ \_\_\_ \_\_\_ \_\_\_ \_\_\_ \_\_\_ \_\_\_ \_\_\_ \_\_\_ \_\_\_ \_\_\_ \_\_\_ \_\_\_ \_\_\_ \_\_\_ \_\_\_



In this lab the scheme is a text file which you use to construct a binary tree. The zeroes and ones describe the path to get to the leaf, where a "0" means a left child and a "1" means a right child. To be consistent with the test in the autograder, all the non-leaf nodes must store empty strings ("") and each leaf node must store the letter as a string ("M"). The scheme above turns into a tree looking like this:

After the scheme builds the tree, the tree decodes the binary message. You follow the zeroes and ones until you reach a node, a leaf, which stores the letter as a string. As you can see for yourself, each path is unique, i.e., each letter’s left-right code arrives at exactly one letter.

**Assignment**

The shell deHuffman prompts the user to read two files, the binary message and the Huffman coding scheme. (Hint: use nextLine, not next, to read the lines of the file.) Write one method that builds the tree and another method that decodes the binary message.

The phrase we used above is stored as message.maips.txt and scheme.maips.txt. Set a debugger in your deHuffman, run it on maips, and see if your tree looks like the tree above. Then decode the other text files in the folder. You know your code is correct if everything comes out in readable English.