COMS W3137, Homework 4, DUE: 4/7/15, in class

Non-programming Problems, 7 points each.

1. Exercise 6.7b

2. Exercise 6.8

3. Exercises 5.1 (assume table size of 10)

4. Exercises 5.2. Rehash using a table size of 19 and a hash function of h(x) = x mod 19.

5. Exercise 5.6

6. Suppose you want to create a “dictionary” of images for lookup. Input is a thumbnail image of each larger image that we want to store. The lookup method will use the thumbnail to index the larger image. Each thumbnail image is 20 pixels wide by 10 pixels high, and contains 256 colors/pixel. Assume we want to store 1000 images. Suggest a hashing function that can be used to perform the lookup given a thumbnail image.

Programming Problems:

Note that up to 10% of your points may be deducted for failing to make it easy to understand and run your submission. This means a) documenting the expected behavior and needed libraries in a readme file, and b) submitting a build script if compilation is more complicated than simply invoking the javac compiler. There is a placeholder directory for each of the parts in the COMS3137 github repository.

1. (20 pts) create a perfect Hash Function for the 46 JAVA language reserved words below. If you can’t find a perfect function, then your grade on this question will be a percentage of the total points evaluated this way:

$$6/\max(6,\text{collisions}) * 20$$

Your program must print out a list of each word and its index value in the table, and mark any words which are in collision. You must also print out the exact hash function you used. Note: you may only use a TABLESIZE of 46, and no double hashing is allowed. Keep in mind that generating 46 if statements to check for a certain string is NOT considered a hash function! You need to come up with a simple and efficient function on your own - you may not use

various hash functions that are floating around on the web. As usual, any hash function in the text or discussed in class is usable. A file with the reserved words can be found at:

https://github.com/CURG/COMSW3137/blob/master/homeworks/homework_4/problem_1/reservedwords.txt

2. (38 points) Huffman codes: Reference, class notes and section 10.1.2 (page 433) of Weiss. Write a program to create a Huffman code for a message. Input to the program should be a text message to be encoded. Your program should do the following:

a. (8 points) In a GUI window (see figure 1), input a message to be encoded and compute character frequencies for each character in the message. Store these frequencies in an array indexed by each character's code, and print out each character code with its frequency. You can print the frequencies out in a GUI window or on the terminal.

   Message: abaccda

   CHAR ASCII FREQ
   a 97 3
   b 98 1
   c 99 2
   d 100 1

b. (25 pts) Create a priority queue (heap) prioritized by the smallest frequency for the characters in the message. Using the priority queue, create a Huffman tree for the message Your program should display the tree in the GUI window (see figure 1). Each node of the tree should display the character and its frequency if a leaf node, and if an interior node it should print the combined frequencies of its two children. You may use the heap code in the Weiss text to implement the Priority Queue but you may not use any Java Collections API PriorityQueue class or methods.

c. (5 points) Display the Huffman code word below each leaf node of the tree. Use a binary 1 for a left child and a binary 0 for a right child (see Figure 1).

d. Extra Credit: (3 points) In the GUI, display the binary digits of the encoded message using the Huffman code you created.

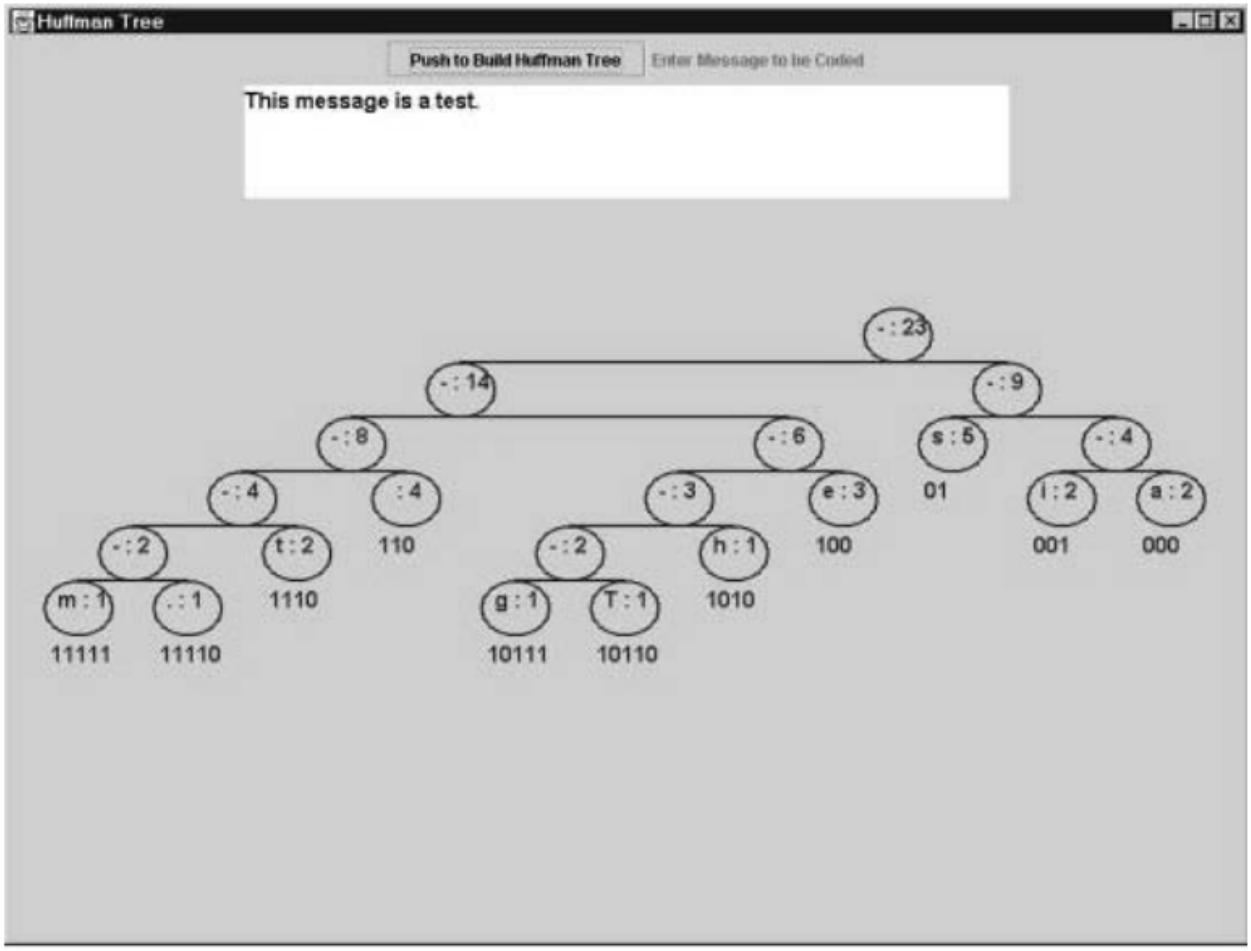

Figure 1: GUI for Huffman Code Tree