Lab Assignment 4: Huffman Encoding

Input: You will be given a text message as input.

Output: You will have to give output a concatenation of four strings on a single line. The four strings will be as follows:

(1) Number of nodes in the Huffman Tree.

(2) Postorder traversal of the Huffman Tree. For nonleaf nodes you should print "0" (zero). For leaf nodes you should print the corresponding character.

(3) Inorder traversal of the Huffman Tree. For nonleaf nodes you should print "0" (zero). For leaf nodes you should print the corresponding character.

(4) Binary encoding (using the Huffman code) of the text message.

Procedure: (1) Read the input and make a frequency table of the characters.

(2) Create a min-priority-queue using a min-Heap. You will have to implement the following functions: Min-Heapify, Build-Min-Heap, Heap-Extract-Min, Heap-Decrease-Key, and Min-Heap-Insert.

(3) Using the frequency table, initialize the min-priority queue by using the frequency as the key.

(4) Create the Huffman Tree using the Huffman's algorithm:

while (there are at least two nodes in the priority queue)
{ Let n_1 be the first deleted node from the priority queue.
Let n_2 be the second deleted node from the priority queue.
Let n be a new node. Set the pointers of n, n_1, and n_2 so that n_1 is left child of n, and n_2 is right child of n. Set the key of n as $n.key = n_1.key + n_2.key$.
Insert the node n into the priority queue.
}

(5) Count the number of nodes in the Huffman Tree and print the first part of the output.

(6) Perform a postorder traversal of the Huffman Tree and print the second part of the output.

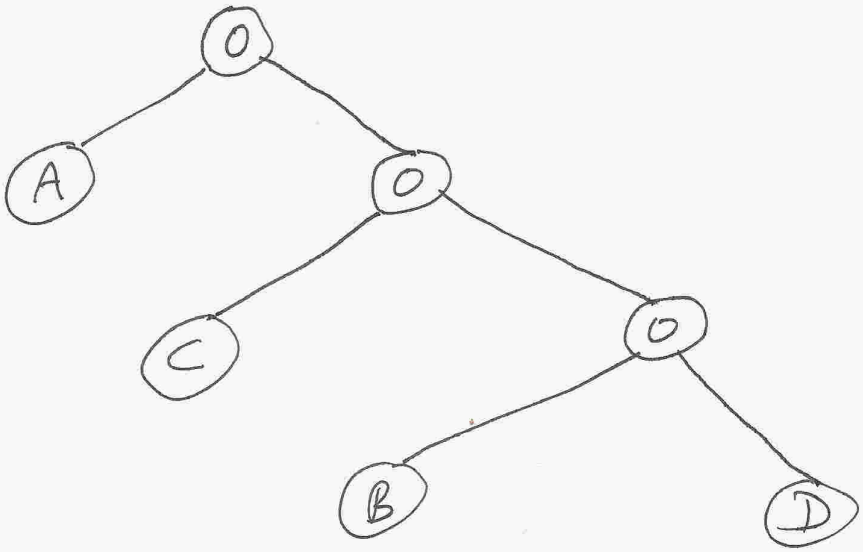
(7) Scan the input message once again, one character at a time, printing the Huffman code corresponding to the character. For printing the Huffman code corresponding to a character, you will have to start from the leaf node corresponding to the character, and move up towards the root.

The code will the binary string in reverse (from root to the leaf node). While moving to a left child, print 0, and moving to a right child, print 1.

Sample Input : ABACCDA

Sample Output : 7ACBD000A0C0B0D01100101011110

The Huffman tree will be :



Postorder Traversal : ACBD000

Inorder Traversal : A0C0B0D

Huffman codes :

A → 0
B → 110
C → 10
D → 111