***** Explanation_3******

- The Hufmann problem used class Node with attributes frequency, char, left, right.
- the implementation keep a dictionary of occurrence of each character.
- for key in dict(characters) the code uses min heap data structure to designated appropriate nodes to the characters.
- the root nodes are popped from the min heap and merged(add the freq and create new arbitrary node) till the length of heap is 1
- This 1 node object is a encoded tree.
- this data is then passed to the decoding function to decode the given data.

Time Complexity:
The time complexity is O(nlogn)
Time complexity for huffman_ecoding method —> nlogn
Time complexity for huffman_decoding method —> O(l) where l is the length of code.