TOPIC 5.8 : HUFFMAN DECODING

Problem Statement  
Given a Huffman Tree and a Huffman encoded string, decode the string to get the original message.

Test Case 1  
Input:  
n = 4  
characters = ['a', 'b', 'c', 'd']  
frequencies = [5, 9, 12, 13]  
encoded\_string = '1101100111110'  
Output: "abacd"

Test Case 2  
Input:  
n = 6  
characters = ['f', 'e', 'd', 'c', 'b', 'a']  
frequencies = [5, 9, 12, 13, 16, 45]  
encoded\_string = '110011011100101111001011'  
Output: "fcbade"

Aim  
To write a program that decodes a Huffman encoded string using the given Huffman Tree constructed from characters and frequencies.

Algorithm

1. Start
2. Construct the Huffman Tree using characters and frequencies (same as Huffman Coding)
3. Initialize an empty string for the decoded message
4. Traverse the encoded string bit by bit:
   * Move left if the bit is 0
   * Move right if the bit is 1
   * When a leaf node is reached, append the character to the decoded message and reset to the root
5. Return the decoded message
6. Stop

Input and Output  
A screenshot of a computer

AI-generated content may be incorrect.

Result  
The program successfully decodes a Huffman encoded string using the constructed Huffman Tree.

Performance Analysis  
Time Complexity: O(m + n log n), where n is the number of characters and m is the length of the encoded string  
Space Complexity: O(n) for the Huffman Tree