DSA INNOVATIVE ASSIGNMENT

DATA COMPRESSING USING HUFFMAN CODDING

- Huffman coding is a lossless data compression algorithm.
- idea is to assign variable-length codes to input characters, lengths of the assigned codes are based on the frequencies of corresponding characters.

- There are mainly two major parts in Huffman Coding.
  1. Build a Huffman Tree from input characters.
  2. Traverse the Huffman Tree and assign codes to characters.

o Build Huffman Tree :

1. Create a leaf node for each unique character and build a min heap of all leaf nodes.
2. Extract two nodes with the minimum frequency from the min heap.
3. Create a new internal node with a frequency equal to the sum of the two nodes frequencies.
4. Make the first extracted node as its left child and the other extracted node as its right child. Add this node to the min heap.
5. Repeat steps 2 and 3 until the heap contains only one node.
6. The remaining node is the root node and the tree is complete.

o Traverse of Huffman Tree :

1. Traverse the tree formed starting from the root
2. While moving to the left child, write 0 to the array.
3. While moving to the right child, write 1 to the array.
4. Print the array when a leaf node is encountered.

o Used Concept :

1. In this project, we have used the concept of **MIN HEAP** tree.
2. From **MIN HEAP** tree build a tree for create code for each character in the data.
3. From that tree make **2D Array** for store the character and it's compressing code.
4. At the last, print a string of **Compressed Data.**

- **OUTPUT :**

```
--------------------------------------------------------------------------------
Enter the Data : welcome in nirma
--------------------------------------------------------------------------------
Compression Data Table :
character       compress data
n                   000
c                   0010
l                   0011
i                   010
o                   0110
r                   0111
e                   100
                    101
m                   110
a                   1110
w                   1111

--------------------------------------------------------------------------------

Compressed Data : 11111000011001001101101001010100001010000010011111011 0
--------------------------------------------------------------------------------
```