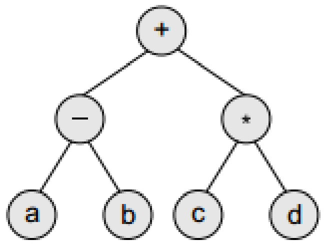
Applications of Binary Tree

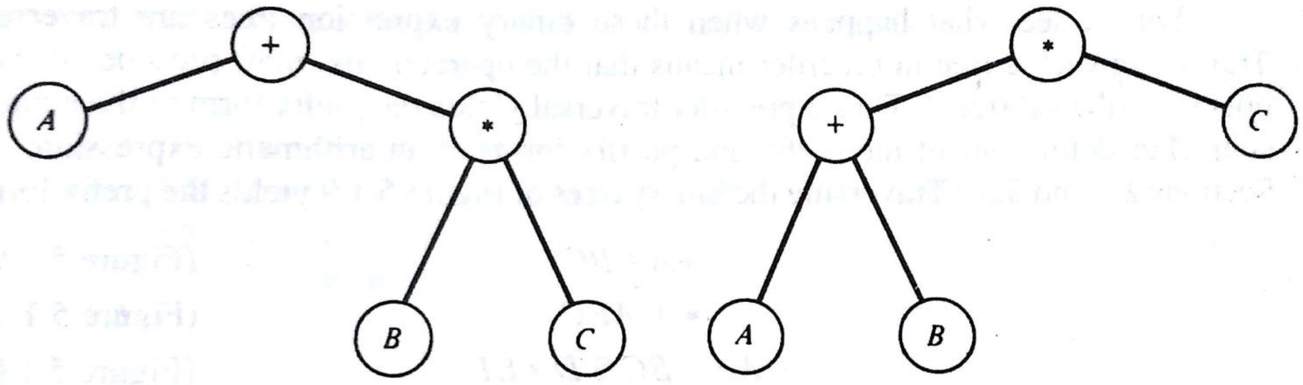
1. Expression Trees

- Expression tree is a strictly binary tree in which each internal node corresponds to operator and each leaf node corresponds to operand
- Binary trees are widely used to store algebraic expressions. For example, consider the algebraic expression given as: Exp = (a – b) + (c * d)
- This expression can be represented using a binary tree as shown in figure below



- The Preorder traversal of a binary expression tree yields the prefix form of the expression.
- The Postorder traversal of a binary expression tree yields the postfix form of the expression.
- The Inorder traversal of a binary expression tree yields the infix form of the expression with the ordering of the operations implied by the structure of the tree, as such tree does not contain parenthesis.

  For example, two expression trees given below, when traversed in inorder, generate same expression i.e. A + B * C. But the evaluation of each of them is different



(a) $A + B * C$ (b) $(A + B) * C$

2. Huffman Coding

Huffman coding is an encoding algorithm developed by David A. Huffman that is widely used as a lossless data compression technique. The idea is to assign variable-length codes to input characters; lengths of the assigned codes are based on the frequencies of corresponding characters. The most frequent character gets the smallest code and the least frequent character gets the largest code.

The variable-length codes assigned to input characters are Prefix Codes, means the codes (bit sequences) are assigned in such a way that the code assigned to one character is not prefix of code assigned to any other character. This is how Huffman Coding makes sure that there is no ambiguity when decoding the generated bit stream.

Let us understand prefix codes with a counter example. Let there be four characters a, b, c and d, and their corresponding variable length codes be 00, 01, 0 and 1. This coding leads to ambiguity because code assigned to c is prefix of codes assigned to a and b. If the compressed bit stream is 0001, the de-compressed output may be "cccd" or "ccb" or "acd" or "ab".

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.

1. Steps to build Huffman Tree (Huffman Algorithm)

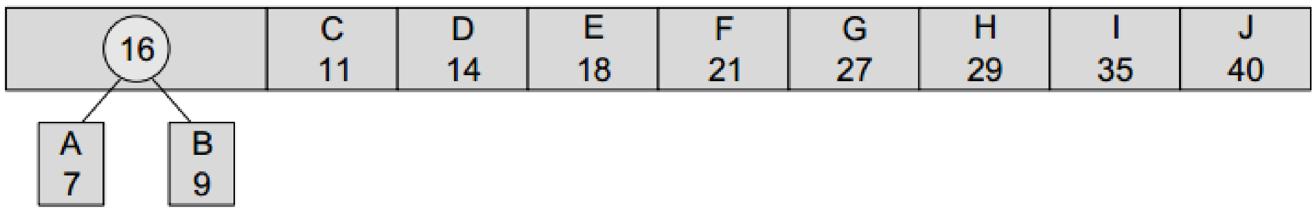
Input is array of unique characters along with their weight or frequency of occurrences and output is Huffman Tree.

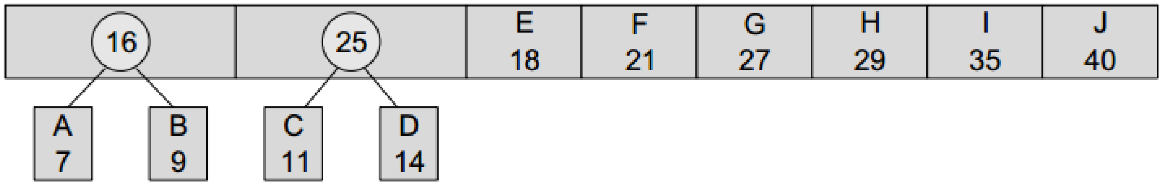
1. Create a leaf node for each character. Add the character and its weight or frequency of occurrence to the priority queue.
2. Repeat Steps 3 to 5 while the total number of nodes in the queue is greater than 1.
3. Remove two nodes that have the lowest weight (or highest priority).
4. Create a new internal node by merging these two nodes as children and with weight equal to the sum of the two nodes' weights.
5. Add the newly created node to the queue.

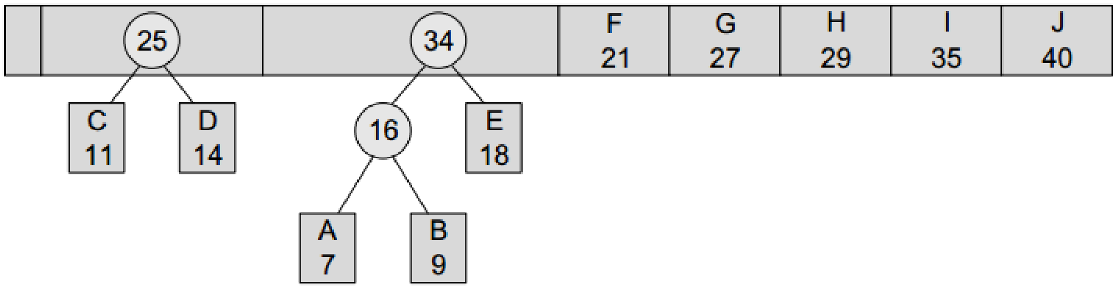
Example: Create a Huffman tree with the following nodes arranged in a priority queue.

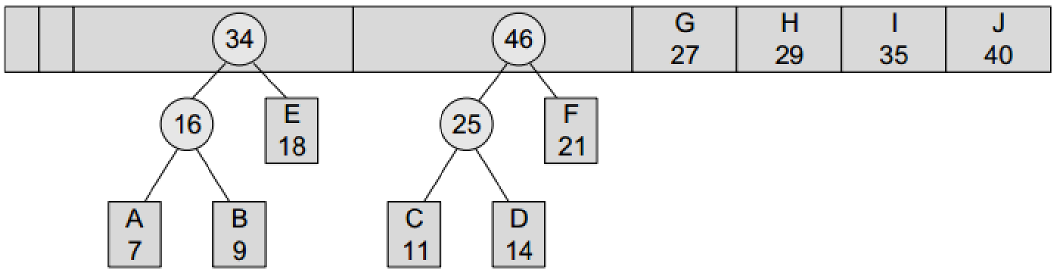
A	B	C	D	E	F	G	H	I	J
7	9	11	14	18	21	27	29	35	40

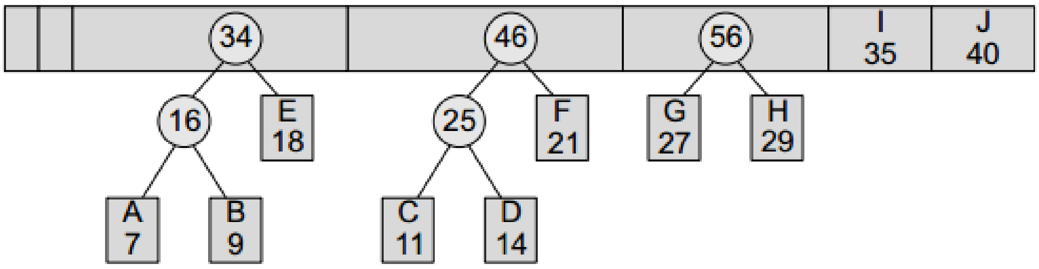
Solution:

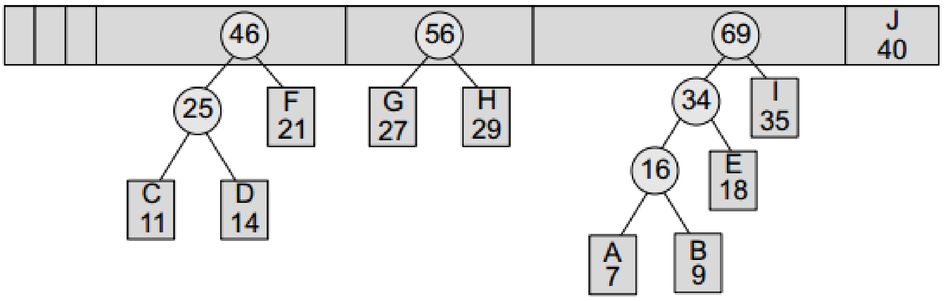


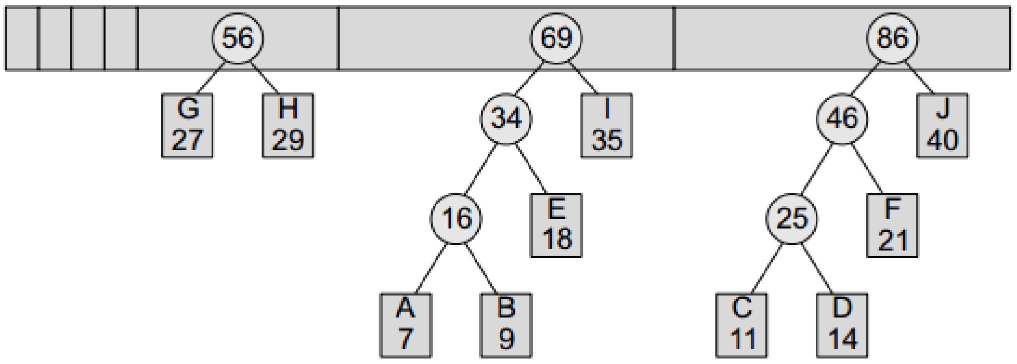

16
25
E 18
F 21
G 27
H 29
I 35
J 40
A 7
B 9
C 11
D 14

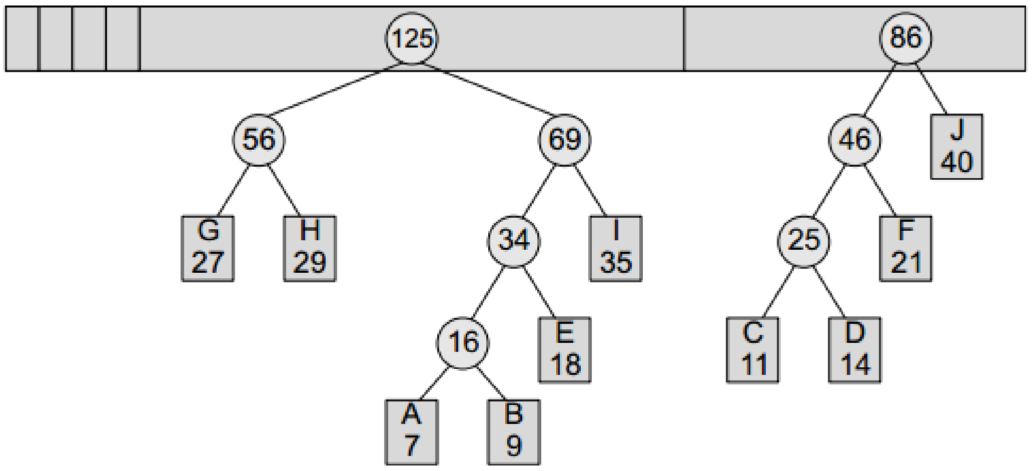

25
34
F 21
G 27
H 29
I 35
J 40
C 11
D 14
16
E 18
A 7
B 9

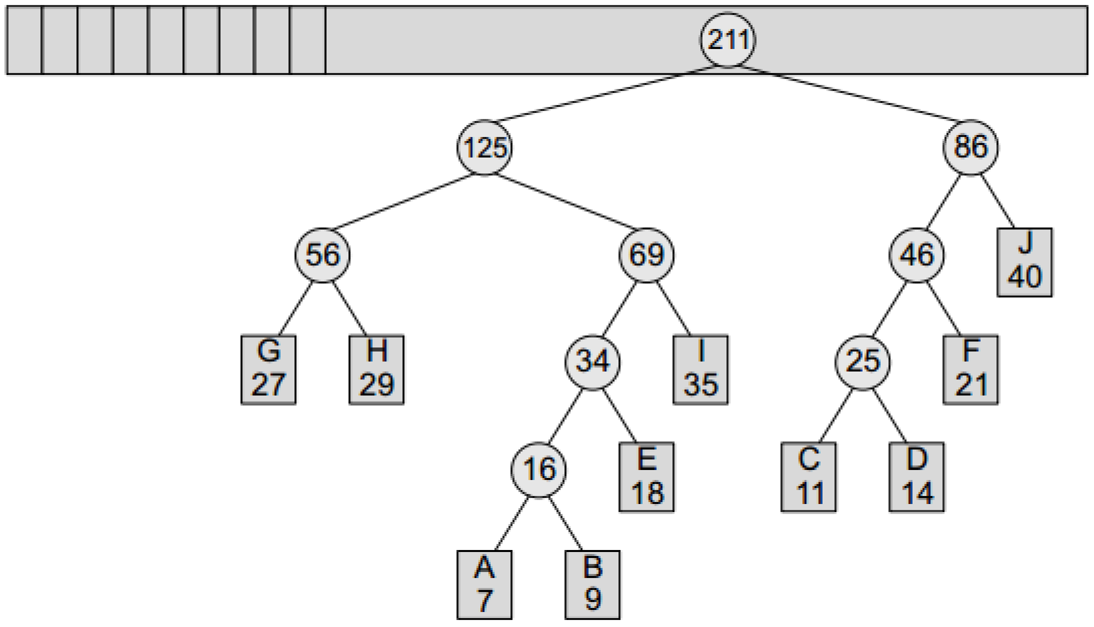

34
46
G 27
H 29
I 35
J 40
16
E 18
25
F 21
A 7
B 9
C 11
D 14


34
46
56
I 35
J 40
16
E 18
25
F 21
G 27
H 29
A 7
B 9
C 11
D 14


46
56
69
J 40
25
F 21
G 27
H 29
34
I 35
C 11
D 14
16
E 18
A 7
B 9

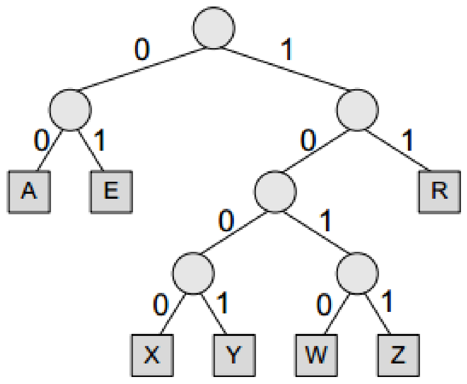

56
69
86
G 27
H 29
34
I 35
46
J 40
16
E 18
25
F 21
A 7
B 9
C 11
D 14


125
86
56
69
46
J 40
G 27
H 29
34
I 35
25
F 21
16
E 18
C 11
D 14
A 7
B 9


211
125
86
56
69
46
J 40
G 27
H 29
34
I 35
25
F 21
16
E 18
C 11
D 14
A 7
B 9

2. Steps to print codes from Huffman Tree

In the Huffman tree, circles contain the cumulative weights or frequency of occurrences of their child nodes. Every left branch is coded with 0 and every right branch is coded with 1. So, the characters A, E, R, W, X, Y, and Z are coded as shown in Table.



Huffman Tree

Character	Code
A	00
E	01
R	11
W	1010
X	1000
Y	1001
Z	1011

Characters with their codes

Some Properties of a Binary Tree

1. The maximum number of nodes at level 'i of a binary tree is 2^i.
2. Maximum number of nodes in a binary tree of height 'h' is $2^{h+1} - 1$.
3. In a Binary Tree with N nodes, minimum possible height is **ceiling (Log_2(N+1)-1).**
4. A Binary Tree with L leaves has at least **ceiling(Log_2L) + 1** levels
5. In Binary tree where every node has 0 or 2 children, number of leaf nodes is always one more than nodes with two children.
6. Minimum number of nodes in a strictly binary tree is 2h + 1, where h is the height of a tree.