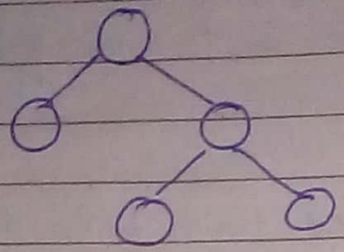
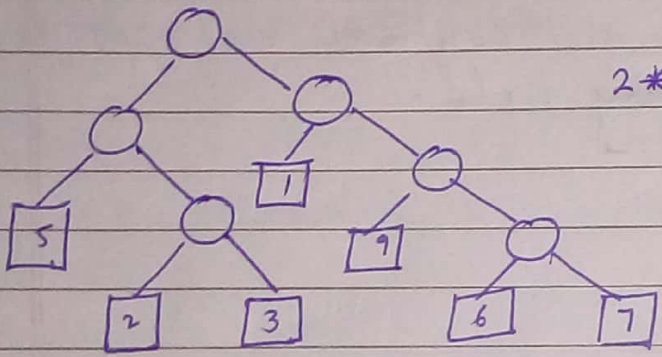
2 - TREE

A tree having every node either does not have any child or two children.



Weighted Path Length

Only the leaf nodes are considered



2*5 + 3*2 + 3*3 + 2*1
+ 3*9 + 4*6 + 4*7

HUFFMAN'S Algorithm to make a tree with minimum weighted path length.

Use to make a 2-tree out of given weights so that the path length of the tree is minimum.

ALGO

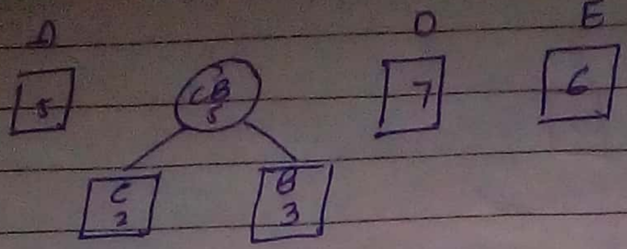
Starting with only weights keep on combining 2 min. weights into a sub-tree untill a single tree is found.

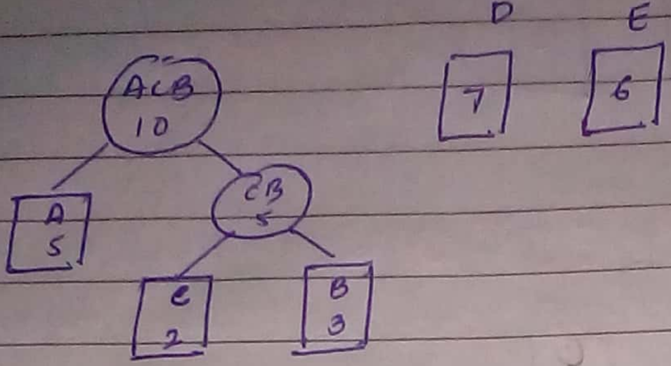
e.g Apply Huffman's algorithm on following weights.

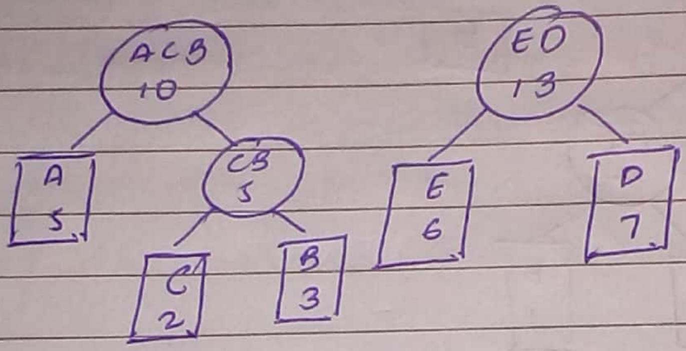
A	B	C	D	E
5	3	2	7	6

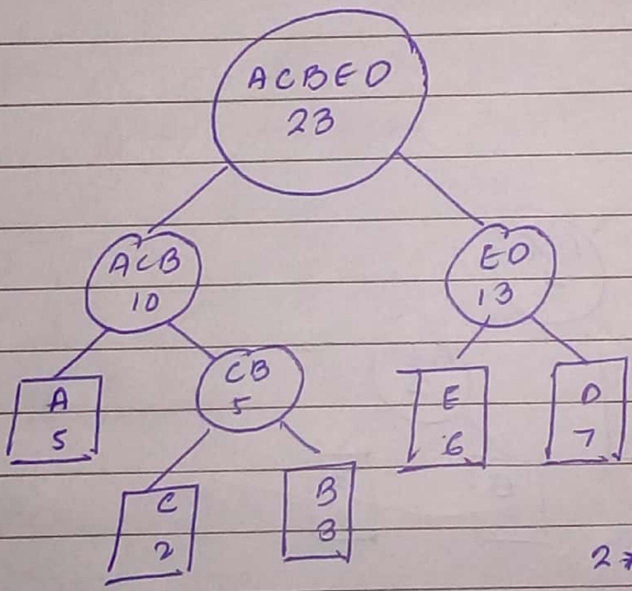
Date: / / /

Scanned by CamScanner









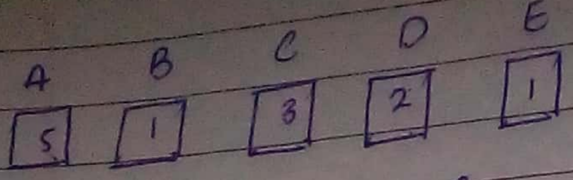
$2*5 + 3*2 + 3*3 + 2*6$
$+ 2*7$

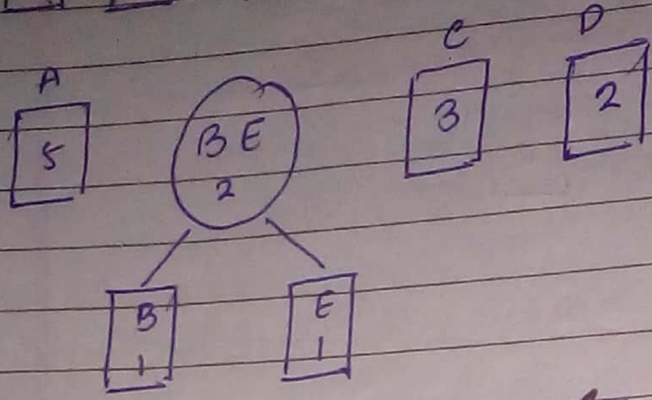
PRACTICAL Application of Huffman's Algorithm

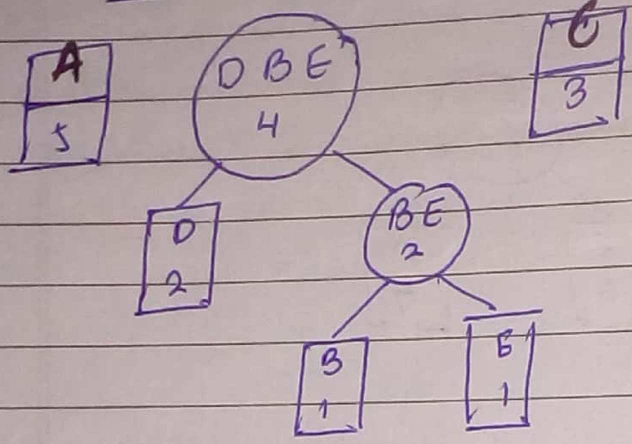
- for encoding signals

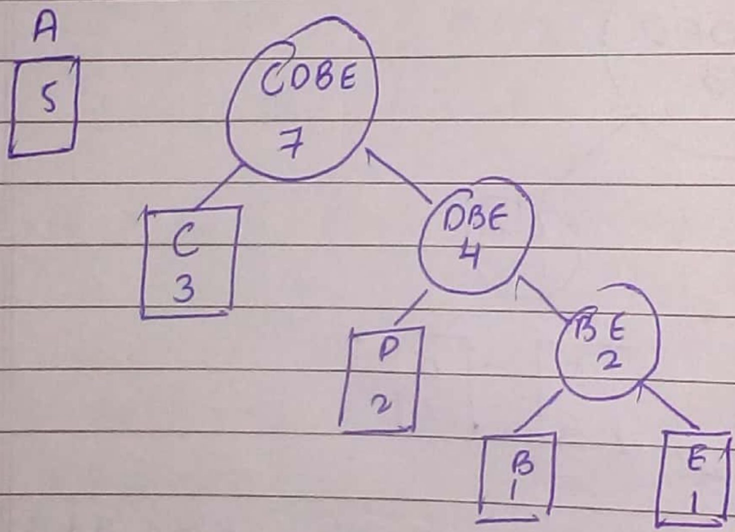
suppose we want to transmit following stream of signals

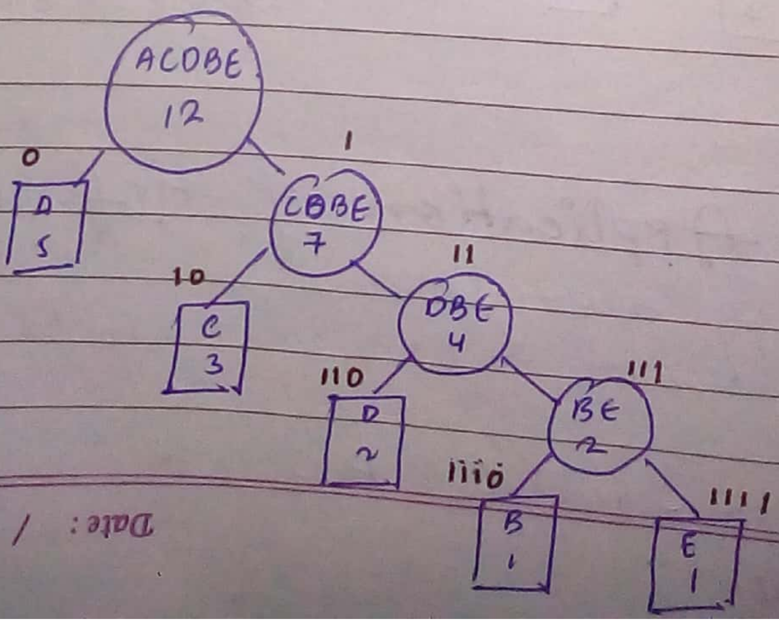
A B A A C C D D E A A C


A 5
B 1
C 3
D 2
E 1


A 5
BE 2
B 1
E 1
C 3
D 2


A 5
DBE 4
D 2
BE 2
B 1
E 1
C 3

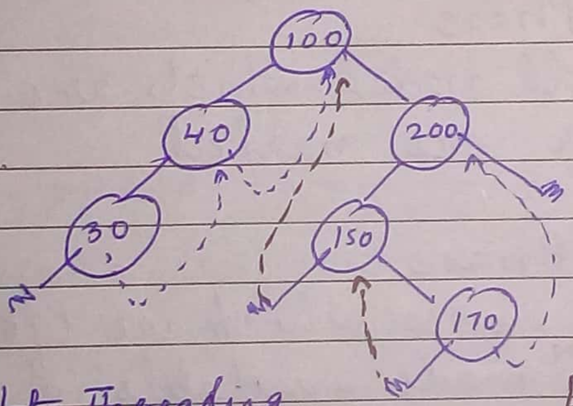

A 5
CDBE 7
C 3
DBE 4
D 2
BE 2
B 1
E 1


ACDBE 12
0
A 5
1
CDBE 7
10
C 3
11
DBE 4
110
D 2
111
BE 2
1110
B 1
1111
E 1

A	0	1 * 5 = 5
B	1110	4 * 1 = 4
C	10	2 * 3 = 6
D	110	3 * 2 = 6
E	1111	4 * 1 = 4
		25

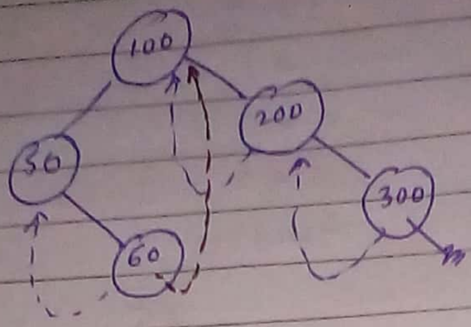
THREADED BST

In case of the nodes of a BST, left and right fields are pointing to NULL. These fields may be used for storing the address of inorder predecessor and inorder successor. If we do so then we need not to make referrance to stack for going to inorder successor or predecessor.



Right Threading Left Threading

If we store address of inorder successor in a field which was earliar pointing to NULL, this is called Right Threading. Similarly available left fields may be used to store add. of inorder predecessor. This will be left Threading



GARBAGE COLLECTION

Routine to detect and reclaim memory which is no more being pointed by an external pointer. So this memory is reclaim by Garbage Collection Routine of o.s and is added to available pool of memory. Garbage collection consists of 2 phases.

Phase 1 : Marking Phase

In this phase all nodes which are being pointed by external pointer are marked

Phase 2 : Collection Phase

In this phase all nodes which are left unmarked are collected reclaim and made available in availablelist of memory.

SPARSE ARRAY

If in an array majority of elements are 0 then we may go for its sparse representation. In sparse representation we will have representation only for non-zero elements, so will be saving memory. Sparse representation of non-zero elements of a 2-D array can be done by using triplets

Scanned by CamScanner

eg $\begin{matrix} 0 & 0 & 0 & 50 \\ 20 & 0 & 0 & 0 \\ 0 & 0 & 90 & 0 \end{matrix}$

[0, 3, 50], [1, 0, 20], [2, 2, 90]

DEQUEUE

A queue where insertion and deletion can be done from both sides.

1. Input Restricted Dqueue
   Insertion - One End
   Deletion - Both Ends

2. Output Restricted Dqueue
   Insertion - Both Ends
   Deletion - One End

PRIORITY QUEUE

Elements are sort as per priority it means highest priority elements are served first. It can be done on two ways.

1. Fixing Priority While Inputting Elements
   Whenever we insert an element, we sort it as per priority, it means it is inserted in a pos^n acc. to its priority. So in this system highest priority element will always be at the front, so inserting will be slower while serving is faster.

2. Whenever a new element comes irrespective of its priority it is always inserted at rear.

Whenever we are serving the element, highest priority element will be searched in queue and that will be served. It means now

Abstract Data Type

Is how a data type, is defined in its abstract form, in its mathematical model. Consists of two components.

1. How data is arranged in its abstract form.
2. What all operation may be performed on that data type