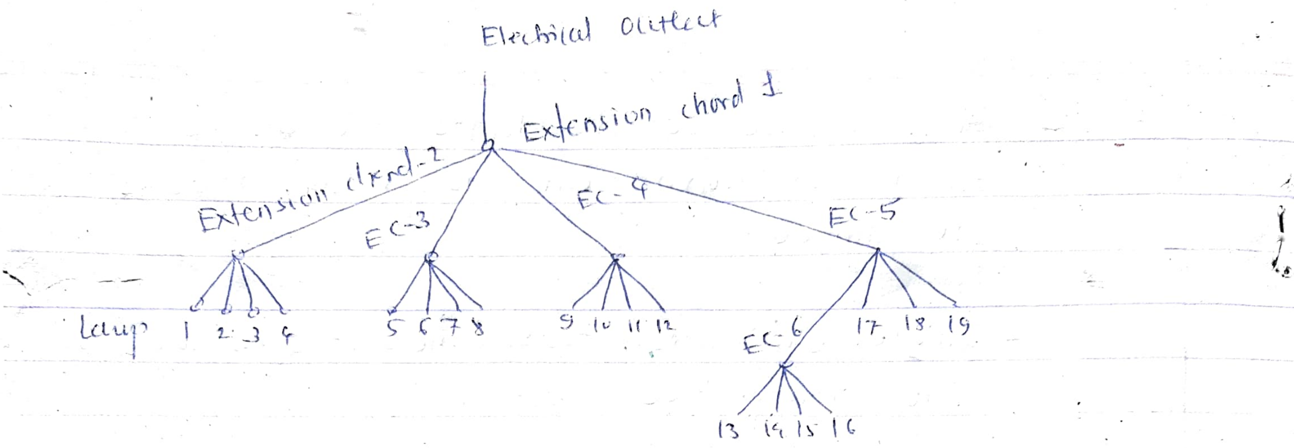
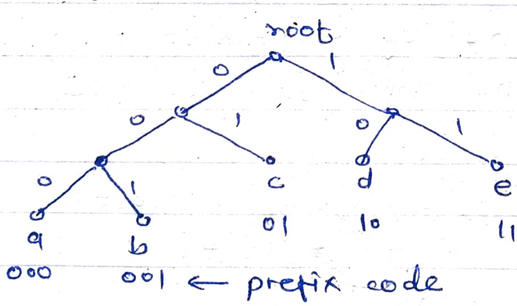


* <u>Prefix Codes</u> & <u>Binary Search Trees</u>

A set of sequence is said to be a prefix code if no sequence in the set is a prefix of another sequence in the set

Ex. {01, 10, 11, 000, 001} – Prefix code but
{1, 00, 01, 000, 0001} – Not prefix code
because the sequence 00 is a prefix of sequence 0001.

To obtain prefix code, full binary tree is used as shown in fig. where 0 is assigned to left branch & 1 is assigned to right branch.



<u>Optimal Tree</u>

Let T be any full binary tree and let $w_1, w_2, w_3 \dots w_t$ be the weights of the terminal vertices (leaves).

Then, weight (W) of the binary tree is

$$\boxed{W(T) = \sum_{i=1}^{t} w_i l_i}$$

$l_i \to$ length of path from root

DSGT-4

H. E. Suryavanshi

9

The Shirpur Education Society's

R. C. PATEL INSTITUTE OF TECHNOLOGY

Nimzari Naka, Shirpur, Dist - Dhule (MS)

Ph No. : (02563) 259600,259801 Telefax : (02563) 259801

Website : http://www.rcpit.ac.in

High Caliber Technical Education in an Environment that Promotes Excellence

TEST (I / II) / PRELIMINARY EXAMINATION

Name of Candidate : ______________________

(IN BLOCK LETTERS) (Surname) (First Name) (Middle Name)

Year : FE / SE / TE / BE Branch : ________ Division : ________ Roll No. : ________

Semester : I / II Name of Subject : ______________________

Total Supplements : 1 + ______ = ______

Signature of Student

Signature of Supervisor

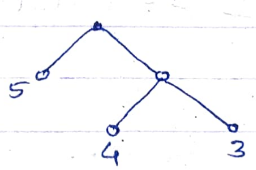
Question Number	1	2	3	4	5	6	7	8	9	10	Total
Marks Obtained											
Marks out of											

Signature of Examiner :

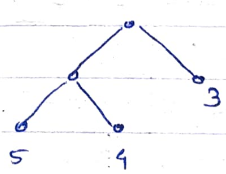
Signature of Moderator :

(Start From here only)

The full binary tree is called an optimal tree if its weight is minimum.



(T_1)



(T_2)

Weight of $T_1 = w(T_1) = (5\times1)+(4\times2)+(3\times2)$

$= 5+8+6$

$\therefore\ w(T_1) = 19$

Weight of $T_2 = w(T_2) = (5\times2)+(4\times2)+(3\times1)$

$= 10+8+3$

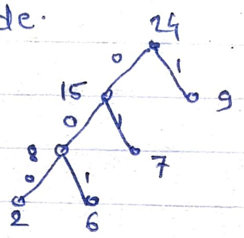
$w(T_2) = 21$

$\therefore\ w(T_1) \le w(T_2)$ $\therefore$ T_1 is optimal tree

5

* **Optimal prefix codes**

A binary prefix code obtained from a optimal tree is called optimal prefix code.



optimal prefix codes {000, 001, 01, 1}

* **Huffman Algorithm to find optimal tree**

Let $w_1, w_2, \ldots w_t$ be the weights of leaves & it is required to construct optimal binary tree.

Algorithm

Step ① Arrange the weights in increasing order.

Step ② Select two leaves with minimum weight w_1 & w_2.
- Add new node i.e parent node with weight $(w_1 + w_2)$

Step ③ Repeate step no. ② untill no weight remains.

Step ④ Tree obtained in this way is optimal tree for given weights & stop.

* **Binary Search Tree**

Used to solve searching problems.

Let $K_1, K_2, K_3 \ldots K_n$ be the n-items in ordered list which are known as keys.
& $K_1 < K_2 < K_3 \ldots < K_n$

Given item 'x', the problem is to determine whether 'x' is equal to one of the keys or 'x' falls between ~~x_i~~ k_i & k_{i+1}

Ex. x is less than k_1, OR
x is equal to k_1, OR
x is greater than k_1 but less than k_2
and so on.

This problem can be solved by __Binary Search Tree__

n - branch nodes $(k_1, k_2, k_3 \cdots k_n)$
$(n+1)$ - leaf nodes $(k_0, k_1, \cdots k_n)$

It is convenient to take middle of key (k_i) from $k_1, k_2, \cdots k_n$ as root.

For branch node with label k_i,
- left sub-tree contains vertices with label $< k_i$ &
- right subtree contains label $\geq k_i$

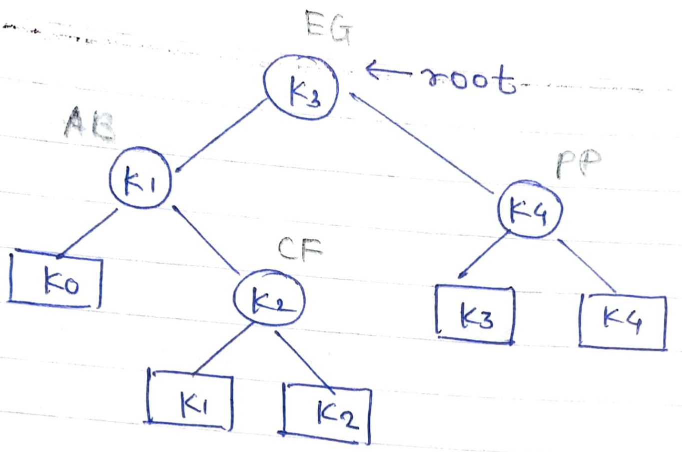
__Circle__ - denotes branch nodes &
__Square__ - denotes leaf nodes

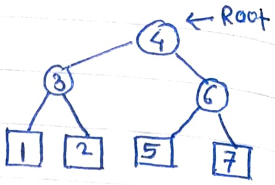
__Search Procedure__

- Compare 'x' with root k_i
  - If 'x' is equal to k_i, stop
  - If $x < k_i$, → compare 'x' with left son of k_i
  - If $x > k_i$, → compare 'x' with right son of k_i

Such comparision continues for succesive branch nodes until either x mathces a key or leaf is reached.

Ex.





Let AB, CF, EG, PP be the keys k_1, k_2, k_3 & k_4
& item to be search is BB

① Compare BB with k_3 (i.e EG)

② Since BB < EG
Compare BB with k_1 (i.e AB)

③ Since BB > AB
Compare BB with k_2 (i.e CF)

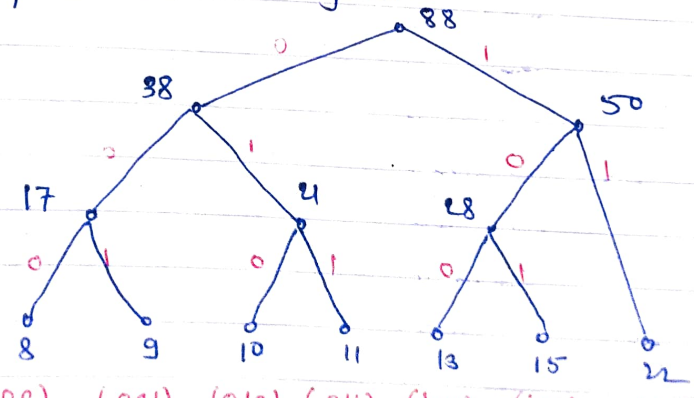
④ Since BB < CF
the leaf labeled k_1 is reached

It means item 'BB' is larger than AB (i.e k_1) but less than CF (i.e k_2)

Ex. ① Construct optimal tree for the weights 8, 9, 10, 11, 13, 15, 22.
Find the weight of the optimal tree.

⇒

By Huffman Algorithm.

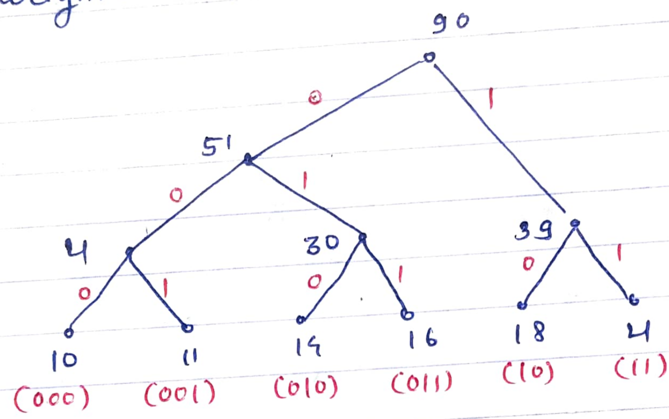


optimal code → (000) (001) (010) (011) (100) (101) (11)

$$\text{Weight of tree} = (8\times3) + (9\times3) + (10\times3) + (11\times3) + (13\times3) + (15\times3) + (22\times2)$$
$$= 242$$

Ex. ② Construct optimal binary prefix code for the weights 10, 11, 14, 16, 18, 21.

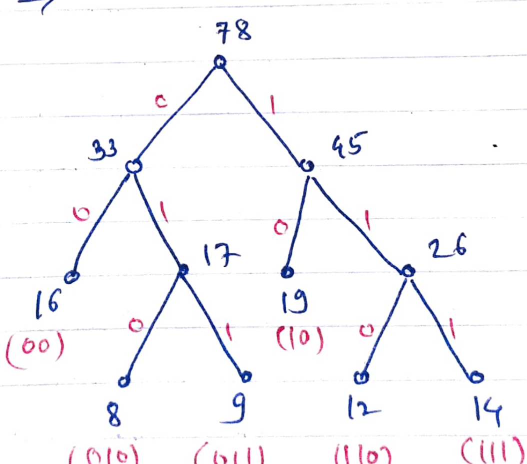
⇒



weight		optimal binary prefix code
10	→	000
11	→	001
14	→	010
16	→	011
18	→	10
21	→	11

Ex. ③ Construct optimal binary prefix code for the weights - 8, 9, 12, 14, 16, 19

⇒

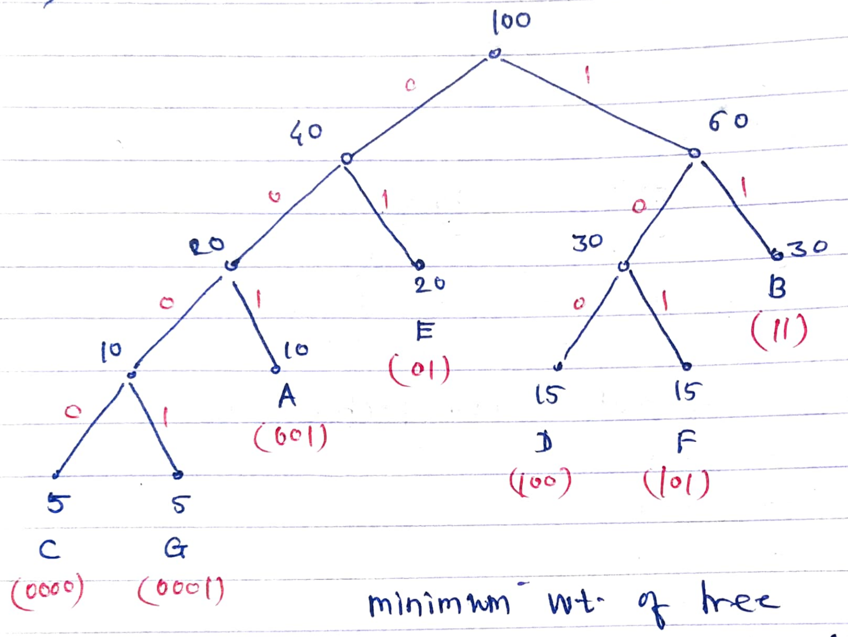


weight		prefix Code
8	→	010
9	→	011
12	→	110
14	→	111
16	→	00
19	→	10

Ex. (4) Suppose data items A, B, C, D, E, F & G occur with the following probability distribution:

Data item	A	B	C	D	E	F	G
Probability	10	30	5	15	20	15	5

⇒ Construct Huffman code. What is the min. wt. path length?



minimum wt. of tree

$= (5\times4) + (5\times4) + (10\times3)$
$+ (20\times2) + (15\times3) + (15\times3) + (30\times2)$

∴ minimum wt. = 260

The minimum weight path length for vertices

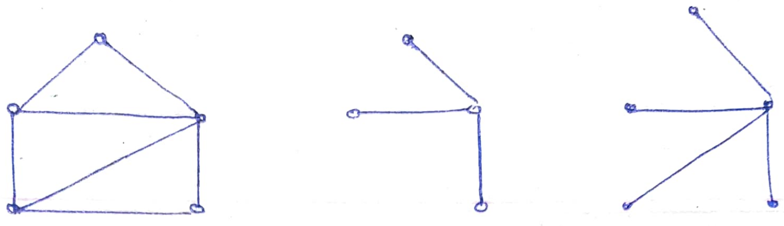
$A \to 3$	$D \to 3$	$G \to 4$
$B \to 2$	$E \to 2$	
$C \to 4$	$F \to 3$	

* Spanning Trees

A tree of a graph is a subgraph of the graph which is a tree.

A spanning tree of a connected graph is a spanning subgraph of the graph which is a tree.

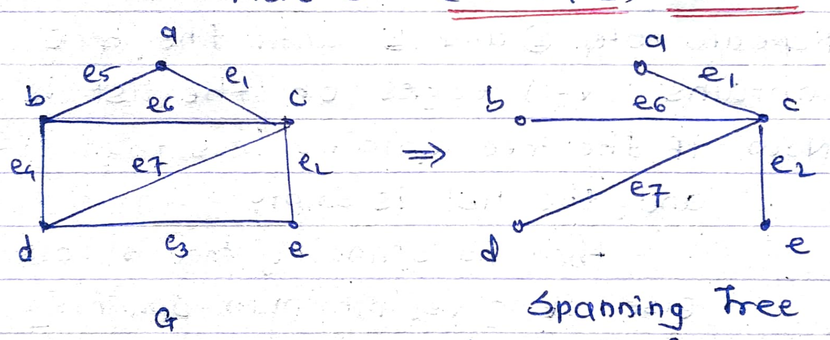
Ex. Fig. (b) & (c) Spanning tree of the graph in Fig. (a). Fig. (c) shows only tree of graph in fig (a)



(a) (b) (c) Spanning Tree

- A branch of a tree is an edge of the graph that is in the tree.
- A chord or link of a tree is an edge of the graph that is not in a tree.
  ↳ The set of the chords of a tree is referred to as the complement of the tree.
- For a connected graph with 'e' edges and 'v' vertices, there are (v−1) branches in any spanning tree.

  ∴ There are (e−v+1) chords.



Spanning Tree

branches = $\{e_1, e_2, e_6, e_7\}$

chords = $\{e_3, e_4, e_5\}$

* Minimum Spanning Tree

For weighted graph, the weight of a spanning tree is defined to be the sum of the weights of the branches of the tree.

A minimum spanning tree is one with minimum weight.

Algorithms to find out minimum spanning tree

1. Kruskal's Algorithm
2. Prim's Algorithm

* Kruskal's Algorithm

This technique was proposed by Joseph Kruskal in 1956.

Algorithm.

① List all the edges of the graph G in increasing order of weights.

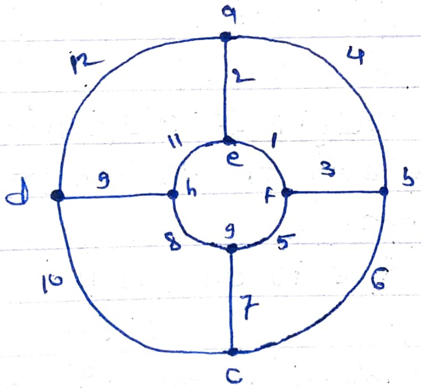
② Select edge with minimum weight from the list and add it to the spanning tree if the inclusion of the edge does not make a circuit.

- If the selected edge makes the circuit, then remove it from the list.

③ Repeate step ① and ② until the tree contains (v-1) edges or the list is empty.

④ Now, if the tree contains less than (v-1) edges and the list is empty then no spanning tree is possible Else it gives minimum spanning tree.

Ex. ① Determine the minimum spanning tree for the graph G shown in Fig. using kruskal's algorithm.



⇒

Edge	Weight	Selection of Edge
e-f	1	Yes
e-a	2	Yes
f-b	3	Yes
a-b	4	NO
f-g	5	Yes
b-c	6	Yes
g-c	7	NO
g-h	8	Yes
h-d	9	Yes
d-c	10	NO
e-h	11	NO
a-d	12	NO

DSGT-4 H.E. Somyavanshi 10

The Shirpur Education Society's

R. C. PATEL INSTITUTE OF TECHNOLOGY

Nimzari Naka, Shirpur, Dist - Dhule (MS)

Ph No. : (02563) 259600,259801 Telefax : (02563) 259801

Website : http://www.rcpit.ac.in

High Caliber Technical Education in an Environment that Promotes Excellence

TEST (I / II) / PRELIMINARY EXAMINATION

Name of Candidate : ______________________

(IN BLOCK LETTERS) (Surname) (First Name) (Middle Name)

Year : FE / SE / TE / BE Branch : ________ Division : ________ Roll No. : ________

Semester : I / II Name of Subject : ______________________

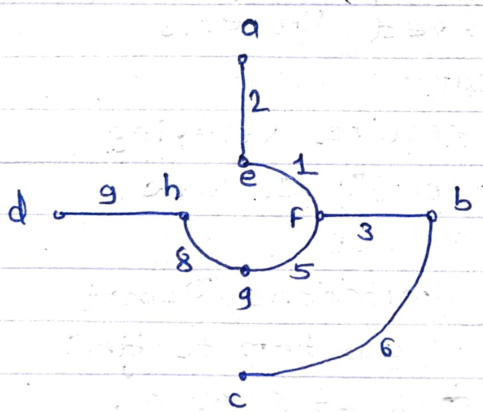
Total Supplements : 1 + ______ = ______

Signature of Student Signature of Supervisor

Question Number	1	2	3	4	5	6	7	8	9	10	Total
Marks Obtained											
Marks out of											

Signature of Examiner : Signature of Moderator :

(Start From here only)



minimum spanning tree

weight of minimum spanning tree

$= 1+2+3+5+6+8+9$

$= 34$

73