Revised according to the latest Syllabus

ENGINEERING MATHEMATICS

Volume-IIIA for 3rd Semester (For CSE & IT)

B. K. PAL
K. DAS

GLORIOUS 105 YEARS OVER

Publishing Books on MATHEMATICS since 1932

U.N. DHUR & SONS PRIVATE LTD.
KOLKATA 700 073

5.3 TREES AND SPANNING TREE

5.3.1. Introduction

Application of graph is not possible without the knowledge of tree. In this chapter we define tree and discuss several theorems on it. Later we introduce spanning tree which is an another important notion in graph theory. This appears in numerous instances.

5.3.2. Trees and Related terms :

Tree. A connected graph without any cycle is called a tree.

The graph shown is Fig. 5.3.1 is a tree.

Remark. (1) We suppose tree contains at least one vertex

(2) Study of trees with infinite number of vertices is beyond the scope of this book.

(3) A tree is always a simple graph because a loop or a pair of parallel edges form a cycle.

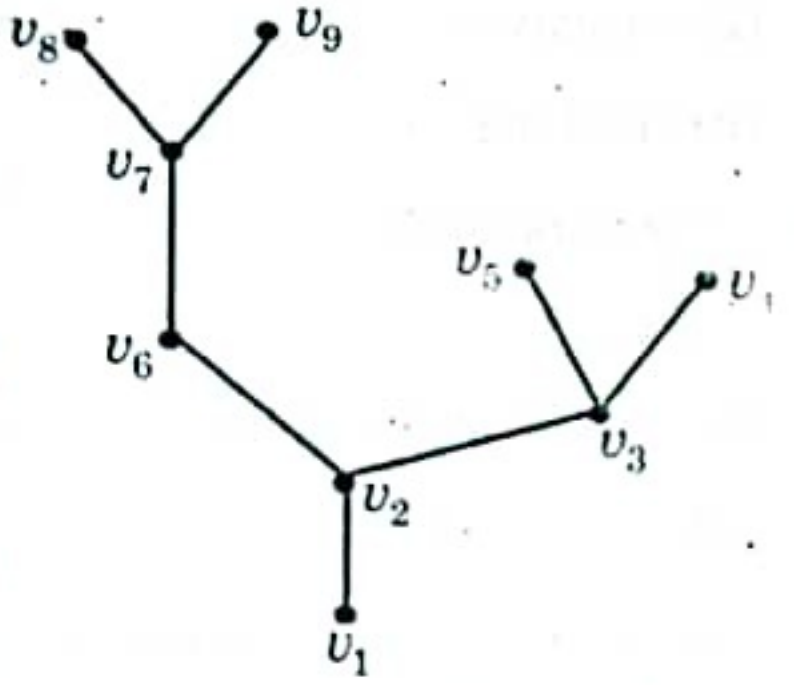


Fig.5.3.1

(4) To pose a practical example we can say that a river with its tributaries and subtributaries can be represented by tree.

Forest. A collection of some trees is called a forest.

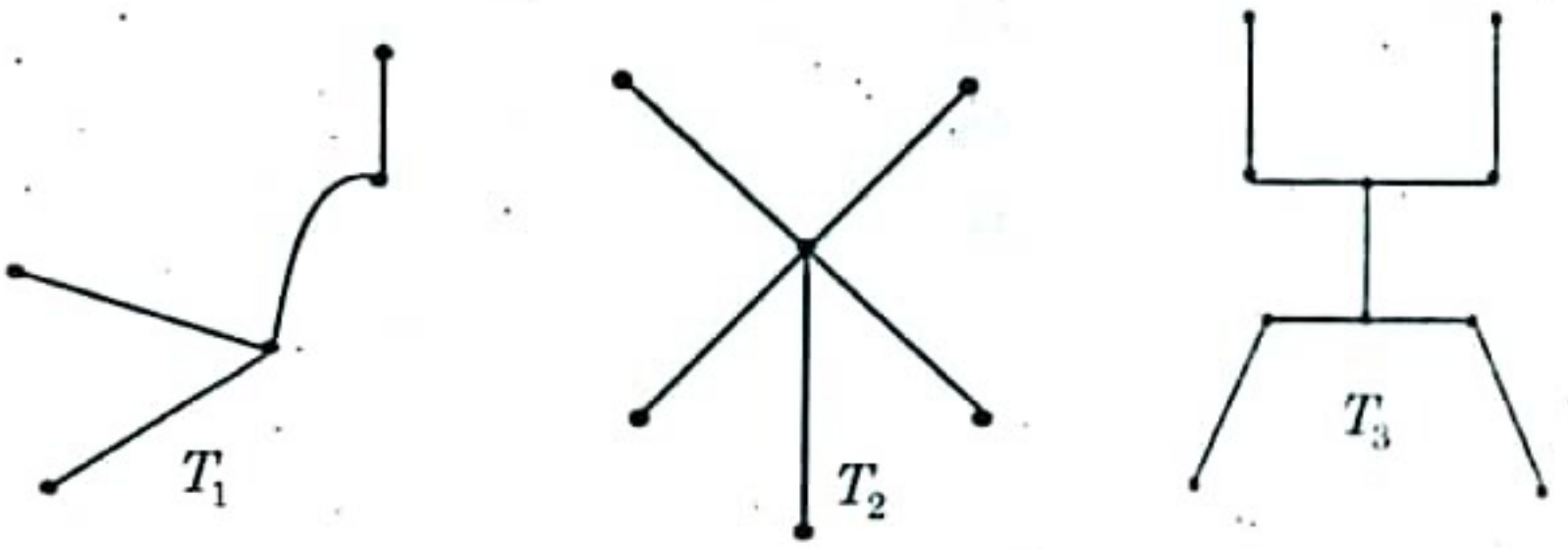


Fig. 5.3.1 (a)

In Fig 5.3.1 (a) the graph T is a collection of three trees T_1, T_2 and T_3. So the graph T is a forest.

Eccentricity of a Vertex.

Let v be a vertex of a graph G. The eccentricity of v, $E(v)$ is the distance from v to the vertex farthest from v in G. Mathematically $E(v) = \max_{v_j \in G} d(v, v_j)$.

In Fig.5.3.2 ; $d(A,C)=1, d(A,B)=2$, $d(A,D)=2, d(A,E)=3$, $D(A,F)=3$. So E or F is farthest from A. So, $E(A)=3$.

Similarly $E(B)=3, E(C)=2, E(D)=2$, $E(E)=3$ and $E(F)=3$.

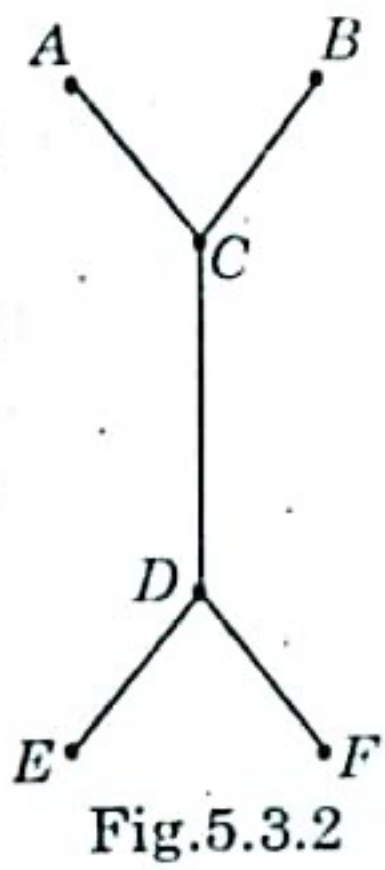


Fig.5.3.2

Note. *Eccentricity becomes larger as vertex lying nearer the extreme of a graph.*

Centre of a Graph.

The vertex with least eccentricity in a graph G is called the centre of G.

In Fig.5.3.2 the vertices C and D are two centres of the graph.

Note. (1) *From the above example it is evident that a graph may have many or unique centre.*

(2) *The centre of graph having only one vertex is nothing but that vertex itself. The eccentricity of that vertex is 0.*

Illustrative Example : Let the graph shown in Fig.4.2.3 shows the 14 branches of a bank. The vertices represent the branches and the edges represent the communication link between its two end branches. Since the graph is connected we know that all the branches can be communicated by any branch, either directly or through some other branches. The eccentricity of each vertex shows how close a branch to the farthest branch of the group of fourteen.

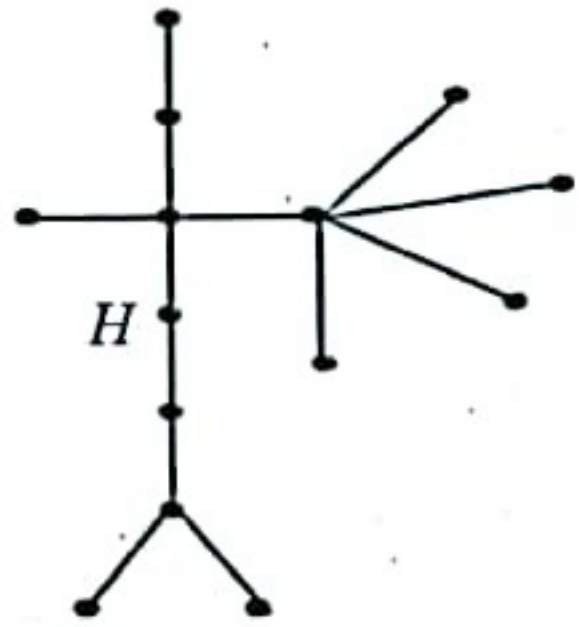


Fig.5.3.3

If it is decided to promote one of the branches to the head-office where closeness of communication is the only criterion then we must think about the branch H because H is the centre of the graph (so far communication link is concerned).

Minimally Connected graph. A connected graph is said to be minimally connected if the graph becomes disconnected when one edge is removed.

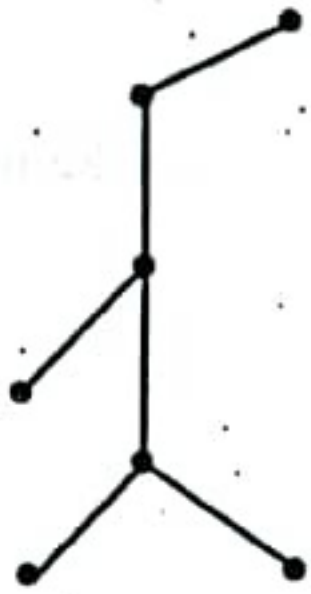


Fig.5.3.4(a)

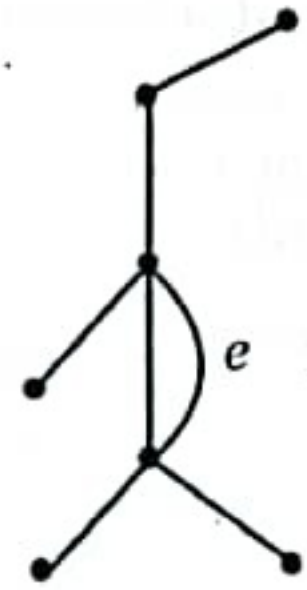


Fig.5.3.4 (b)

The connected graph shown in Fig.5.3.4 (a) is minimally connected whereas the graph shown in Fig.5.3.4(b) is not so because if the edge e is removed the graph is still connected.

Note. *A minimally connected graph can not have a cycle, i.e. a minimally connected graph is a tree.*

Binary Trees.

A tree in which there is exactly one vertex of degree two and each of the other vertices is of degree one or three is called a binary tree.

The graph shown in Fig.5.2.5 is a binary tree. V is the only vertex having degree 2.

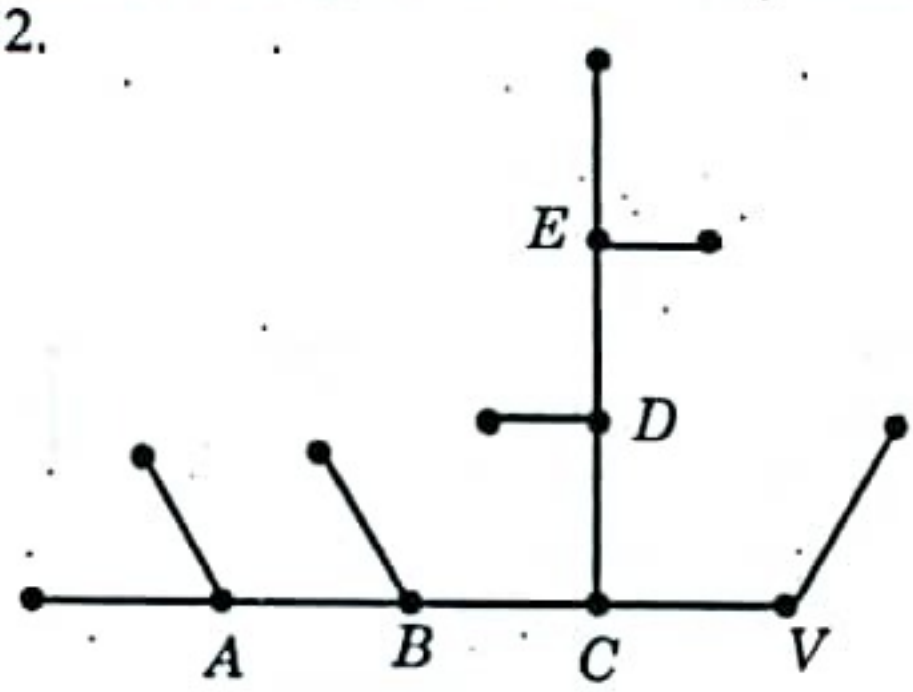


Fig.5.3.5

Note. (1) *The vertex of degree 2 in a binary tree is called* ***Root*** *of the tree.*

(2) *We suppose a binary tree has three or more vertices.*

(3) *One of the most useful applications of binary trees is in search procedures.*

Internal Vertex of a Tree.

The vertex of a tree is called internal vertex which is not pendant vertex. In Fig.5.3.5 the vertices A, B, C, D, E and V are all internal vertices.

5.3.3 Theorems on Trees.

Theorem 1. Every pair of vertices in a tree is connected by one and only one path.

Proof. Let T be a tree ; A, B be an arbitrary pair of vertices. Since T is a connected graph so A and B are connected by a path. Let, if possible, A and B be connected by two distinct paths. These two paths togetherly form a cycle and then T can not be a tree. So there is only one path connecting A and B.

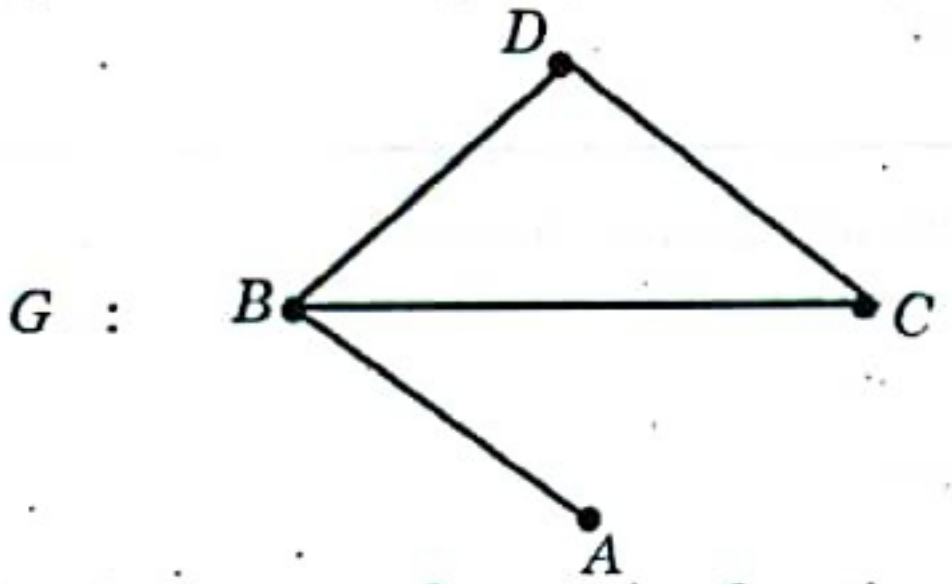
Theorem 2 (Converse of Theorem 1)

If there is one and only one path between every pair of vertices in a graph G then G is a tree.

Proof. Since there is a path between every pair of vertices so the graph G is connected. Let, if possible, G posses a cycle. So there exists at least one pair of vertices A, B such that there are two distinct paths between A and B. This contradicts the hypothesis. So G does not have any cycle. So, G is a tree.

Illustrations.

Ex. 1. *Find the longest and shortest path found in the graph*



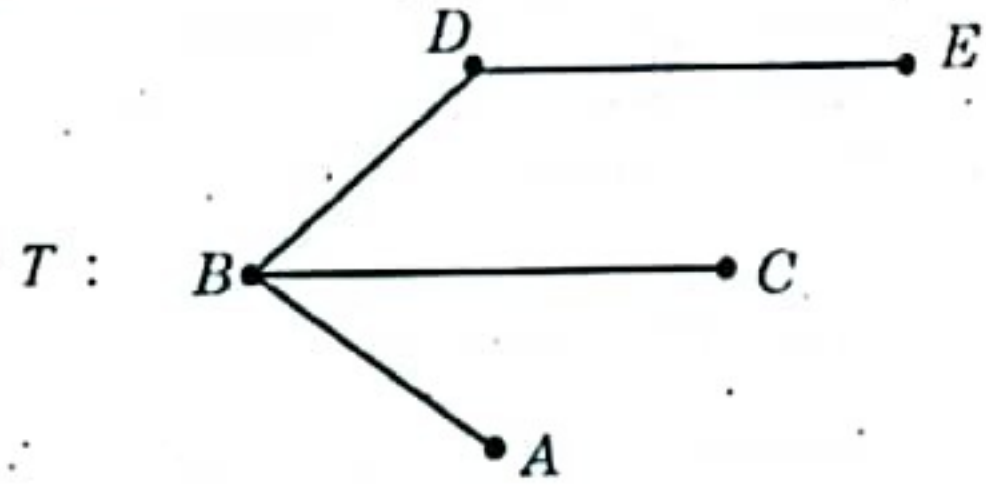
We get several lengths between for a pair of vertices. However we compute them in the following table :

Pair of vertices :	(A, B)	(A, D)	(A, D)	(A, C)	(A, C)	(B, C)	(B, C)	(B, D)
length between the pair :	1	2	3	2	3	1	2	1

Pair of vertices :	(B, D)	(C, D)	(C, D)
length between the pair :	2	1	2

From the table we see the greatest length is 3 and smallest length is 1. So there exist several longest paths and several shortest. Among these $A - B - D$ is a longest path and $C - D$ is a shortest path.

Ex. 2. *Find the longest and shortest path in the tree T.*



Here we see the length is unique for a pair of vertices. We compute them in the following table :

Pair of vertices :	(A, B)	(A, D)	(A, E)	(A, C)	(B, C)	(B, D)	(B, E)
Length between the pair :	1	2	3	2	1	1	2
Pair of vertices :	(C, D)	(C, E)	(D, E)				
Length between the pair :	2	3	1				

We see length of (C, E) is one of the maximum . So, $C - B - D - E$ is one of two longest paths. $A - B$ is a shortest path in the given tree.

Note. *In a graph which is not a tree there are generally many paths between two vertices.But in a tree there is only one path between any two vertices. So the determination of length between two vertices and consequently finding of shortest and longest path becomes more easier in a tree.*

Theorem 3. A connected graph is a tree if and only if addition of an edge between any two vertices in the graph creates exactly one cycle.

Proof. G be the connected graph. Let G be a tree and A, B be two arbitrary vertices in G.

Let an edge e be added between A and B. Since G is a tree, by Theorem 1, we have an unique path P in G joining A, B. Since $e \notin G \therefore e \notin P$. So, P together with e creates a unique cycle.

Conversly, let addition of an edge between any two vertices in the graph creates a unique cycle. Let A, B be any two arbitrary vertices in G and an edge e be added between A and B. Then this makes a unique cycle. So A and B must already be connected by a unique path. So by Theorem 2, G is a tree.

Theorem 4. The degree of each of the origin and terminus of the longest path (the path of maximum length) in a tree with at least two vertices is one.

Proof. Let T be a tree and $P : v_0, e_0, v_1 e_1 ; \cdots v_{m-1}, e_{m-1}, v_m$ be a longest path in T where v_i are vertices and e_i are edges. Since the tree has at least two vertices so $v_0 \neq v_m$. We shall show $\deg(v_0) = \deg(v_m) = 1$.

Let, if possible, $\deg(v_0) \neq 1$. Since degree of vertices of a tree with at least two vertices cannot be 0, so $\deg(v_0) > 1$.

So there must be another edge $e \neq e_0$ joining v_0 to a vertex v of T. If this $v = v_i$ for some i then the path $v_0, e_0 ; v_1, e_1 ; \cdots v_i e, v_0$ forms a cycle. This is impossible since T cannot have any cycle. If v is not equal to any v_i of the path P then $v, e ; v_0, e_0 ; v_1, e_1 ; \cdots v_{m-1}, e_{m-1}, v_m$ becomes a path of length $m+1$. This is again a contradiction since the longest path in T has length m. Thus $\deg(v_0) = 1$.

Similarly we can show $\deg(v_m) = 1$ also.

Theorem 5. Any tree with two or more vertices contains at least two pendant vertices.

Proof. Any two vertices in a tree is connected by one and only one path. Since the tree is supposed to be a finite graph (having finite number of vertices) so there exists a longest path

$$P : v_0, e_0 ; v_1, e_1 ; \cdots v_{m-1}, e_{m-1} ; v_m$$

in the tree. Then from the previous theorem we see v_0 and v_m are pendant and they are distinct. This completes the proof.

Theoem 6. A tree with n number of vertices has $n-1$ number of edges.
[*W.B.U.T. 2013*]

Proof. Let T be the tree. The result will be proved by method of induction on n. Clearly the result is true for $n = 1, 2$.

We assume the result is true for k number of vertices whenever $k < n$. In T let a be an edge with end vertices A and B. Since two vertices in a

tree are connected by only one path so there is no other path between A and B ; a is the only path joining A and B. So, $T-a$ i.e. the graph obtained from T by deleting the edge a, becomes a disconnected graph. Now the graph $T-a$ has exactly two components, say T_1 and T_2. Let T_1 and T_2 contains n_1 and n_2 number of vertices. So, $n = n_1 + n_2$. If the component T_1 contains a cycle then T would have a cycle, which is not possible. So T_1 is a tree. Similarly T_2 is a tree also. So by our hypothesis T_1 has $n_1 - 1$ and T_2 has $n_2 - 1$ number of edges. Thus $T-a$ consists of $(n_1-1)+(n_2-1) = n_1+n_2-2 = n-2$ number of edges. Hence T has $n-2+1 = n-1$ edges.

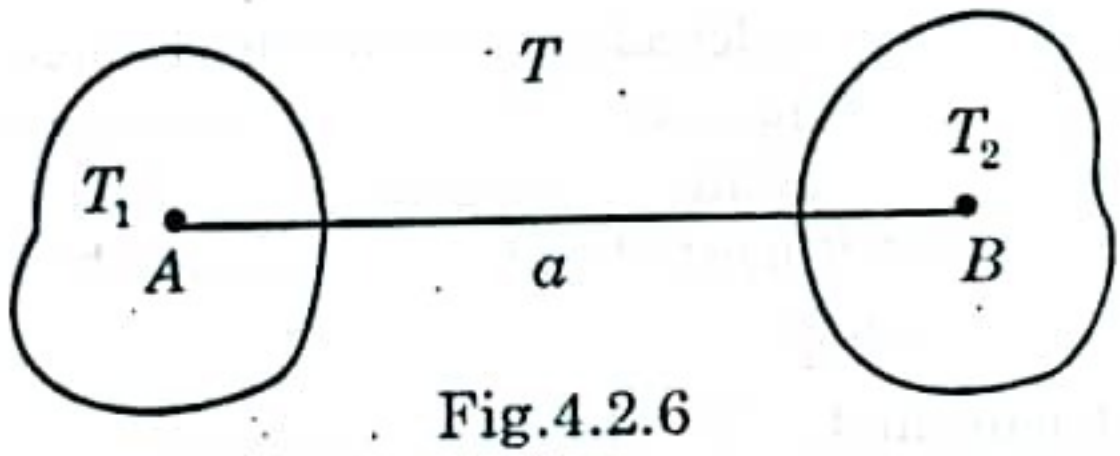


Fig.4.2.6

Theorem 7. (Converse of the above theorem)

A connected graph with n vertices and $n-1$ edges is a tree.

Proof. Let G be a connected graph with n vertices and $n-1$ edges. Let, if possible, G be not a tree. Then G contains a cycle. Let, e be an edge of this cycle. Then the subgraph $G-e$ (the subgraph obtained from G by deleting the edge e) is still connected.

$G-e$ has $n-2$ edges and n vertices. This is not possible since we know a connected graph with n vertices has at least $n-1$ edges.

Theorem 8. A graph is a tree if and only if it is minimally connected.

[*W.B.U.T. 2013*]

Proof. Let T be a tree having n vertices. So, by Theorem 7. T has $n-1$ edges.

If one edge is removed from T then it has $n-2$ edges. Then T becomes disconnected since a connected graph with n vertices must have at least $n-1$ number of edges (Theorem 6 in Art 3.1.5). Thus T is a minimally connected graph.

Conversely, let T be a minimally connceted graph with n number of vertices. Since T is connected graph so, Number of Edges of $T \geq n-1$. Let, if possible, T be not a tree. Then T contains a cycle. T becomes still connected if one edge of this cycle is removed from T. This contradicts our hypothesis that T is a minimally connected graph. Hence T is a tree.

Illustrations.

Ex. 1. *Find the minimum number of line segments to interconnect 100 distinct points.*

From the previous theorem the answer is $100-1=99$.

Ex. 2. *We need only* $n-1$ *pieces of wire to short electrically n pins together. The resulting structure is a tree.*

Theorem 9. A graph with n number of vertices, $n-1$ number of edges and without any cycle is connected.

Proof. Let G be such a graph. Let, if possible, G be disconnected. Then G has two or more connected components. Without loss of generality suppose G_1 and G_2 be two such components. Since G_1 and G_2 are subgraphs of G so they also do not contain any cycle. Let v_j and v_k be two vertices in the components G_1 and G_2 respectively. Add an edge e between v_j and v_k (shown in Fig 4.2.7). Since there is no path between v_j and v_k in G so adding e would not create a cycle. Thus the graph together with G and e(mathematically speaking $G \cup e$) becomes a cycleless connected graph i.e. a tree. We see this tree is having n number of vertices and $(n-1)+1=n$ number of edges. This contradicts the fact that a tree having n number of vertices must have $n-1$ number of edges (See Theorem 7). Hence T is a connected graph.

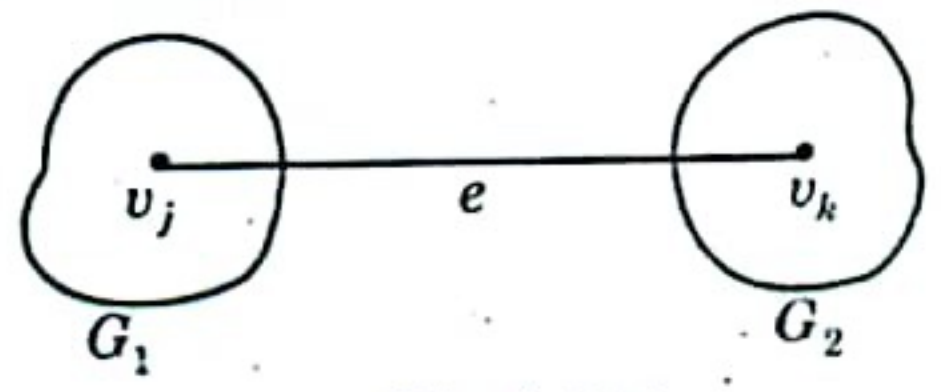


Fig.4.2.7

Theorem 10. Every tree has either one or two centres.

Proof. Left to the reader as exercise (See the steps of finding the centre in subsequent Illust. Ex.)

Note. *If a tree has two centres then the two centres must be adjacent.*

Theorem 11. A tree with two or more vertices is a bipartite graph

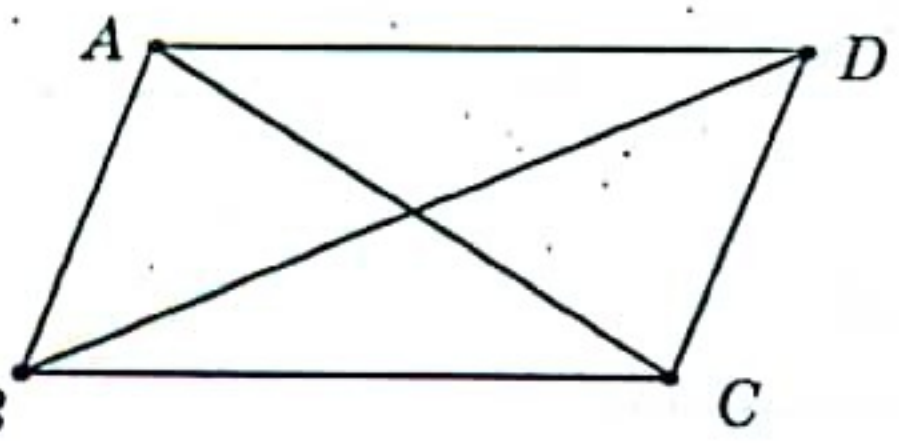
Proof. Let $V=\{v_1, v_2, \ldots . v_n\}, n \geq 2$, be the vertex set of a tree T. Two subsets V_1 and V_2 are being formed from V in the following way :

The vertex $v_1 \in V_1$. If v_2 is adjacent to v_1 then $v_2 \in V_2$ otherwise it $\in V_1$. If v_3 is adjacent to at least one vertex in V_1 then $v_3 \in V_2$ otherwise it $\in V_1$. In this way, in general, if v_r is adjacent to at least one vertex in V_1 then $v_r \in V_2$ otherwise $v_r \in V_1$. Thus each of the vertices of $v_1, v_2, \ldots . v_n$ is classified into the two classes V_1 and V_2. Obviously $V_1 \cap V_2 = \phi$. Since v_1 is not isolated and T has at least two vertices so there exists at least one vertex which is adjacent to v_1. $\therefore V_2 \neq \phi$. Thus V is partitioned into two non-null disjoint subsets V_1 and V_2.

Now let e be an arbitrary edge of the graph T. Its two ends be v_i and v_j. If $v_i \in V_1$ then $v_j \notin V_1$ because v_j is adjacent to v_i. Thus e is incident to one vertex in V_1 and one vertex in V_2. So T is bipartite.

Remark : The converse of the above theorem is not true. For example the following graph is a Bipartite graph, with the partitions $\{A, B\}$ and $\{D, C\}$ of the vertex set, which is not a tree.

G :



Theorem 12. If a Tree is a complete bipartite Graph $K_{m,n}$, then either m or n is 1.

Proof : Let T be a tree which is a complete bipartite graph $K_{m,n}$. Then number of vertices of T is $m+n$. Hence it has $m+n-1$ number of edges. Again $K_{m,n}$ has mn number of edges. Since $T \equiv K_{m,n}$ so $mn = m+n-1$.

$$a, (m-1)(n-1) = 0 \Rightarrow \text{either } m = 1 \text{ or } n = 1$$

Corollary. $K_{1,n}$ or $K_{n,1}$ are the only complete bipartite graph which are Tree also.

5.3.4. Theorems on Binary Trees.

Theorem 1 The number of vertices in a binary tree is always odd.

[W.B.U.T. 2014, 2010, 2007]

Proof. Let a binary tree has n number of vertices. Since a binary tree has one vertex of degree 2 and the other vertices are of degree 1 or 3 so

there are $n-1$ number of odd vertices in the tree. Since we know the number of odd vertices in a graph is even, $n-1$ is an even integer. So n is an odd integer, that is the number of vertices is odd.

Theorem 2. The number of pendant vertices in a binary tree is $\frac{n+1}{2}$ where n is the number of vertices in the tree.

[W.B.U.T. 2015]

Proof. Let x = number of pendant vertices in the binary tree T, i.e., x = number of vertices of degree one. T has only one vertex of degree 2 and each of the other vertices is of degree three. So, sum of the degrees of all vertices of

$$T = x \times 1 + 1 \times 2 + (n - x - 1) \times 3 = 3n - 2x - 1 .$$

We know in a graph, sum of all degrees $= 2 \times$ number of edges.

Since number of edges in a tree is $n-1$, $3n - 2x - 1 = 2(n-1)$

or, $x = \frac{n+1}{2}$.

Theorem 3. The number of internal vertices in a binary tree is one less than the number of pendant vertices. *[WBUT 2012]*

Proof. A non-pendant vertex in a binary tree is an internal vertex. Let the binary tree contains $x+y$ vertices, where x = Number of pendant vertices, y = Number of non-pendant vertices, i.e. internal vertices in the binary tree. $\therefore$ the total number of vertices, $n = x + y$.

From Theorem 2 $x = \frac{n+1}{2}$

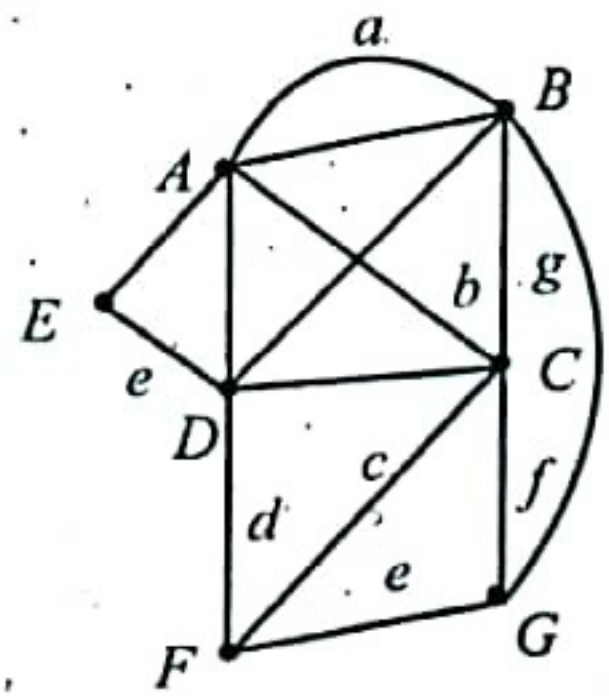
or, $x = \frac{x+y+1}{2}$ or, $2x = x + y + 1$ or, $y = x - 1$

$\therefore$ No. of internal vertices = No. of pendant vertices -1.

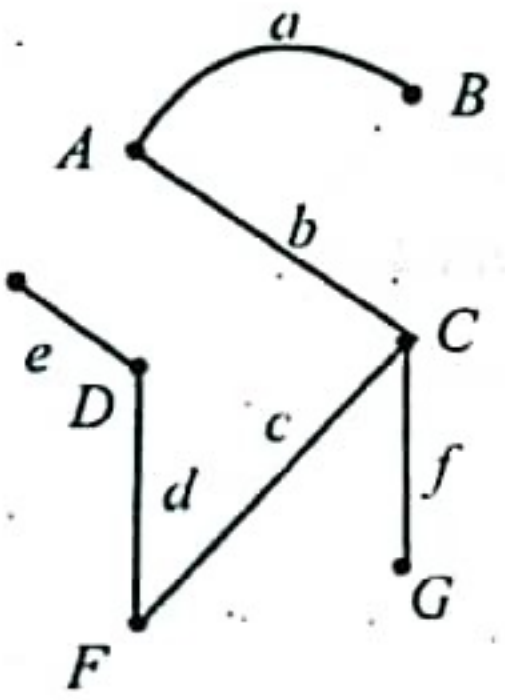
5.3.5 Spanning Tree and Co-Tree.

A tree T is called a spanning tree of a connected graph G if T is a subgraph of G and if T contains all the vertices of G.

The graph T in Fig.5.3.8 (b) is obtained from the graph shown in Fig.5.3.8 (a) just by deleting some particular edges. It is seen that all the vertices of G are included in the tree T. So T is a spanning tree of G.



G : Fig.5.3.8 (a)



T : Fig.5.3.8 (b)

Note. (1) *Sometimes a spanning tree of G is called a skeleton of G.*

(2) *Spanning tree is the largest tree (with maximum number of edges) among all trees in G.*

(3) *Because of (2) we call a spanning tree a maximal tree within G.*

(4) *Spanning tree is defined only for connected graph because a tree is always connected and from a disconnected graph we cannot find a connected subgraph with the same vertex-set.*

(5) *Each component of a disconnceted graph would have a spanning tree. All such spanning trees form spanning forest of G.*

(6) *In practical purpose spanning tree has more importance than that of an ordinary tree in a graph.*

Branch of a Tree.

An edge of a Tree is called a branch of the tree. In Fig.5.3.8 (b) the edges *f, b, e,* etc. are branches of the tree.

Chord (or Tie) of a Tree.

Let T be a tree contained in a graph G. An edge of G that is not in T is called a chord of T in G. In Fig.5.3.8 (a) g is a chord of the spanning tree T in G. It has many other chords.

Note. *Branches and chords are defined only with respect to a tree in a graph G.*

Co-Tree (*or, Chord-set or Tie-set*)

The complement of a spanning tree T in a connected graph G is called Co-tree of T. It is a denoted by $\overline{T}$.

For example, the Co-Tree of the spanning tree T shown in Fig 4.2.8 (b) is

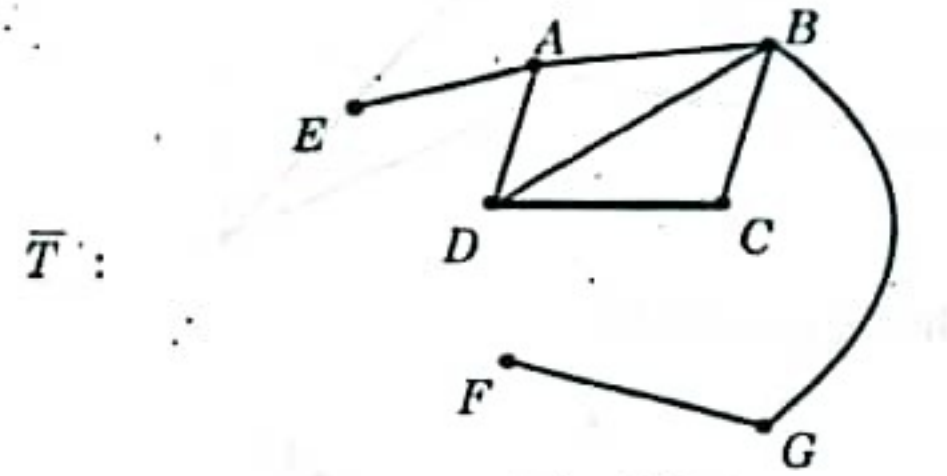


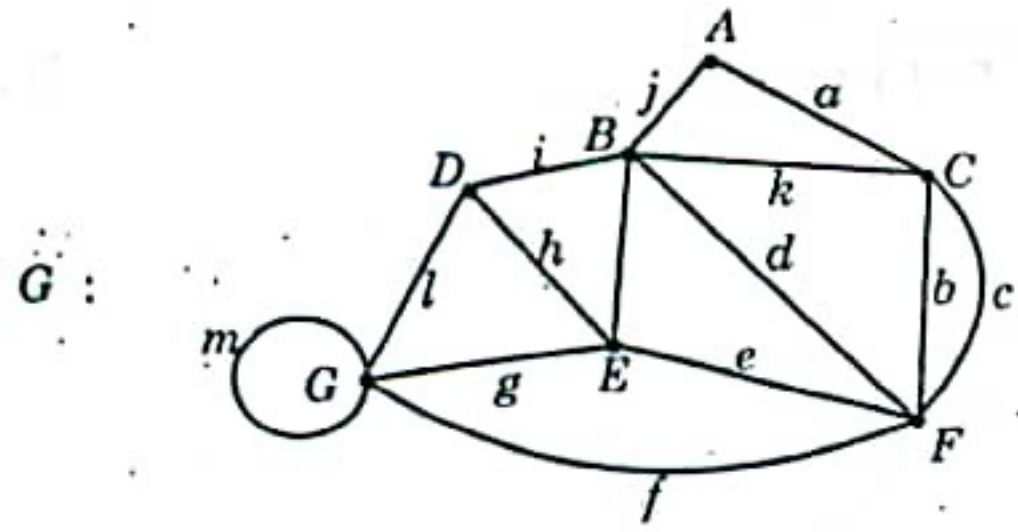
Fig.5.3.9

Note. (1) *Co-tree of T is nothing but the collection of chord of T in G.*

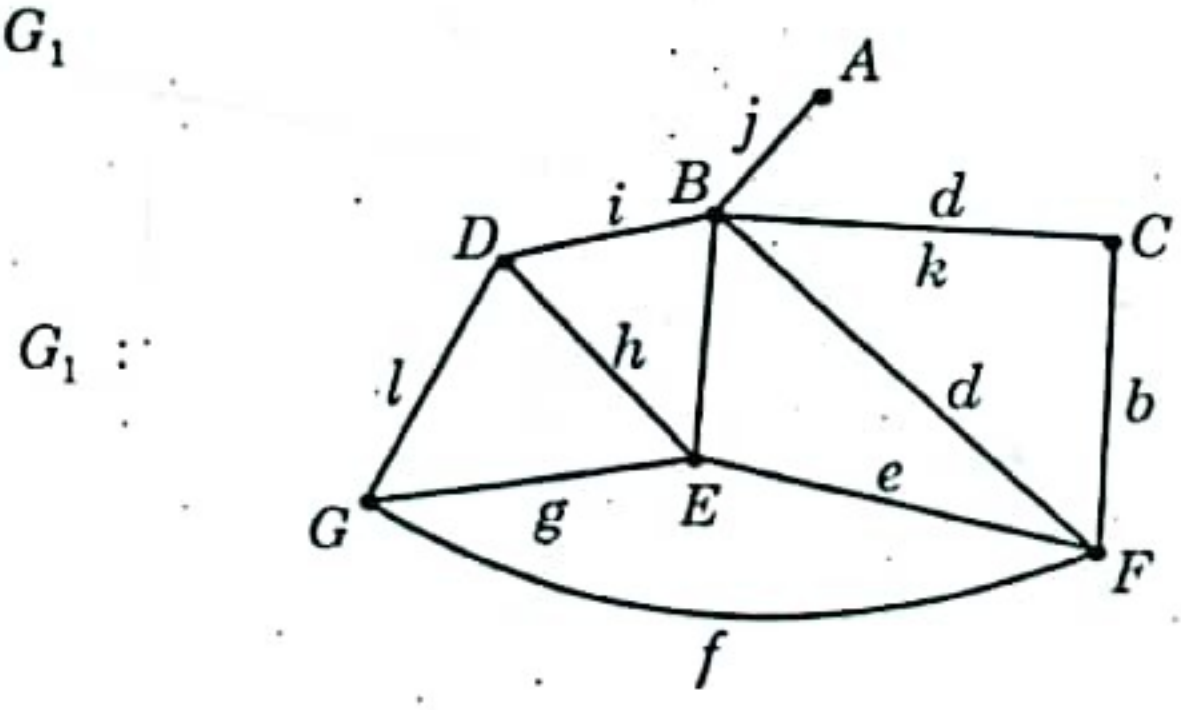
(2) *It is evident that* $T \cup \overline{T} = G$.

Finding a Spanning Tree of a Connected Graph.

Find a spanning tree from the following graph G :

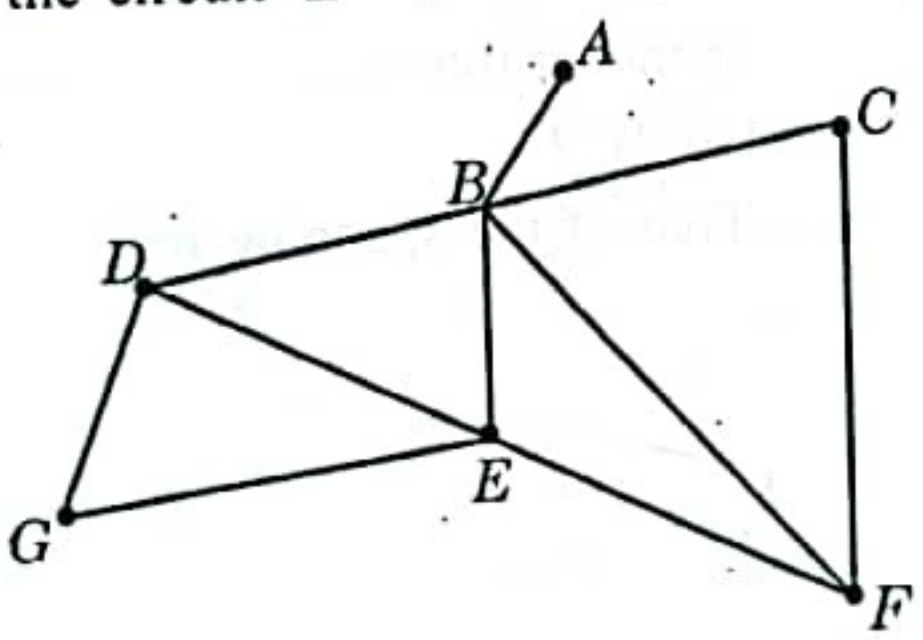


If G has no cycle then G is its own spanning tree. Next remove all selfloops and parallel edges from G (if exist). Now consider a cycle $A-B-C-A$ of G. We delete an edge viz. a from this cycle and get a subgraph G_1



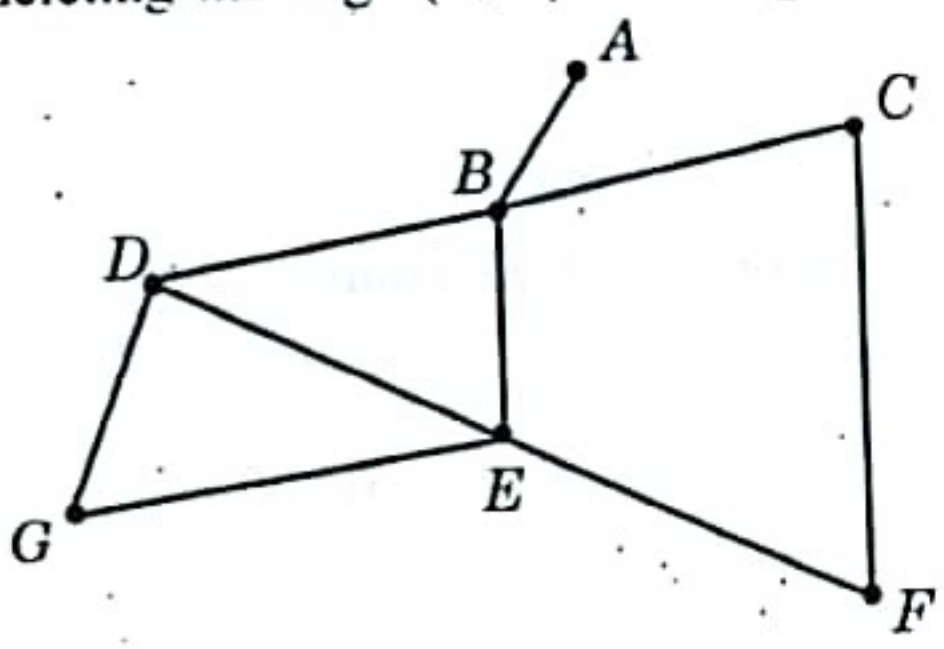
Next consider the circuit $E-G-F-E$. Deleting one edge f from this cycle we get

G_2 :



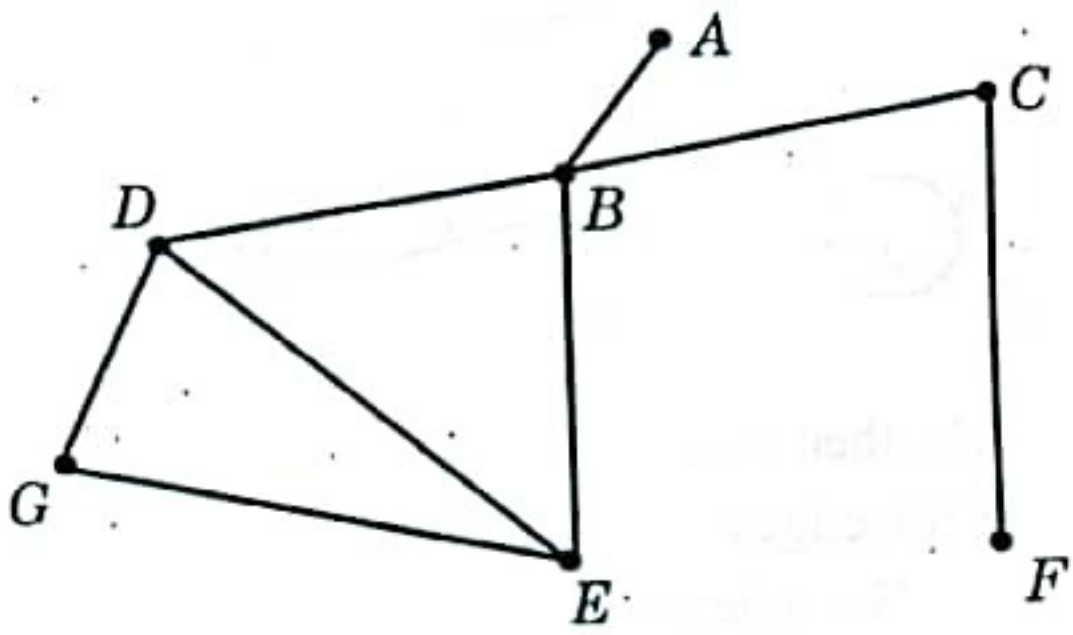
Similarly deleting the edge (B, F) from G_2 we get

G_3 :

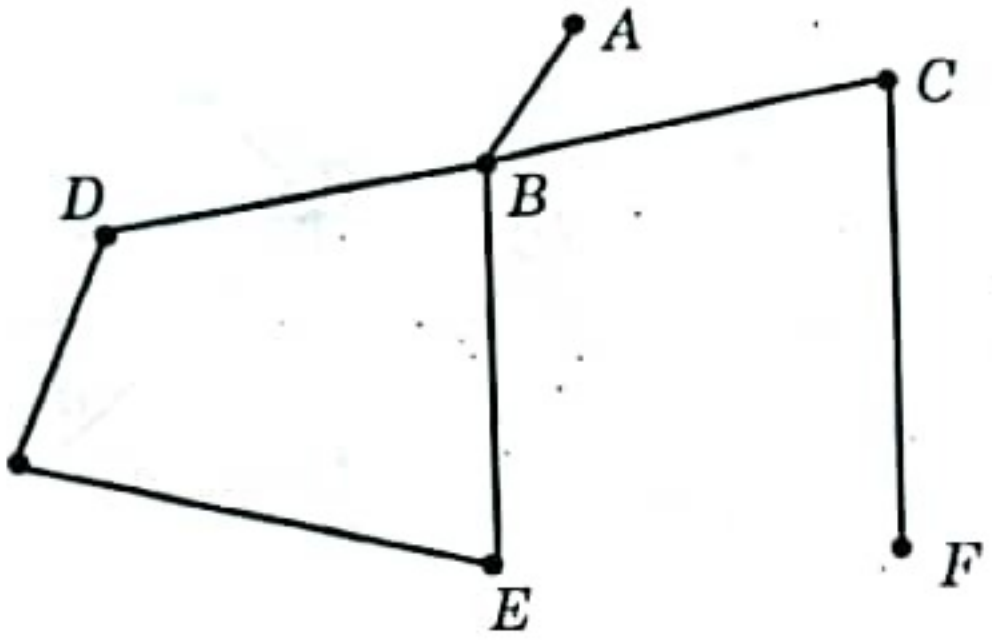


Deleting the edge (E, F) from the cycle $B-C-F-E-B$, we get

G_4 :

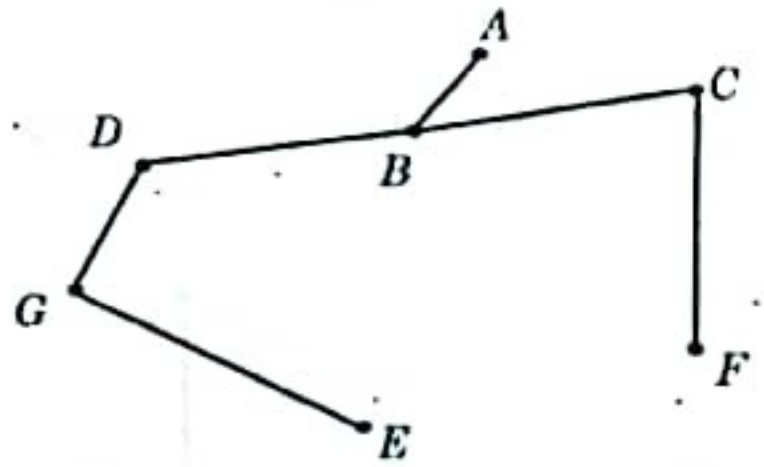


Next G_5 :



Finally we find the cycle $B-D-G-E-B$. From this deleting the edge BE we get the spanning tree

G_6 :



Note. *Though we can find a spanning tree by the above procedure it is not so systemetic. For a graph having so many edges and vertices it is impossible to use the above method. To overcome this we introduce BFS and DFS method of finding a spanning tree. These are algorithm base process. Because of this, these can be used in computer for finding spanning tree.*

5.3.6 Breadth First Search (*BFS*) Algorithm for construction of a spanning tree.

Discard all parallels and loops from the graph.

Choose any vertex v_k of the graph. In this algorithm label this vertex as 0. Then we proceed stage by stage by labelling a new vertex at every stage according to the following rule :

Find all unlabelled vertices in G which are adjacent to the vertices labelled i. Label those vertices as $i+1$ and get them joined (with i-labelled vertex) by edges so that no cycle is formed.

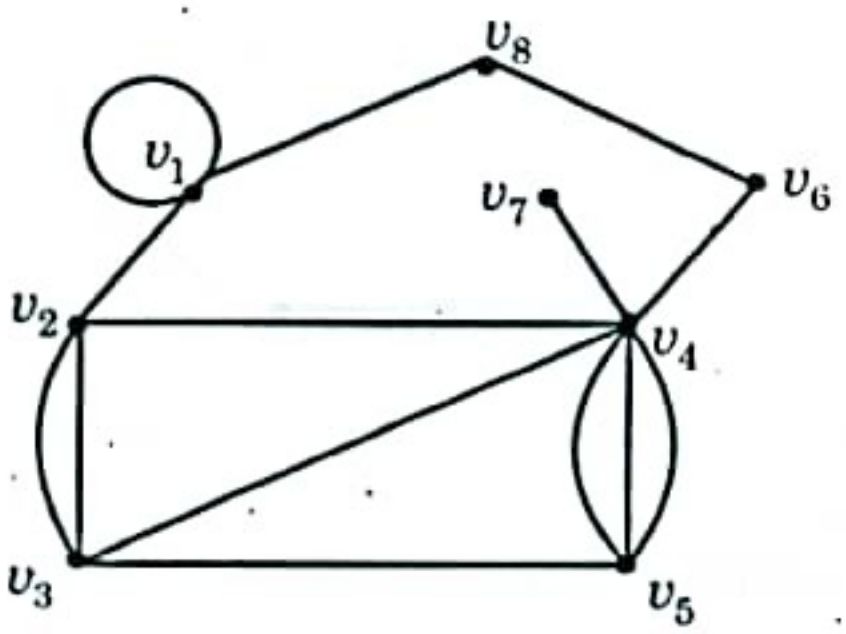
This stage to stage labelling and joining stops when all the vertices are labelled. All the vertices and the successive joining edges form the required spanning tree.

To be more precise see the following Illustrative Examples.

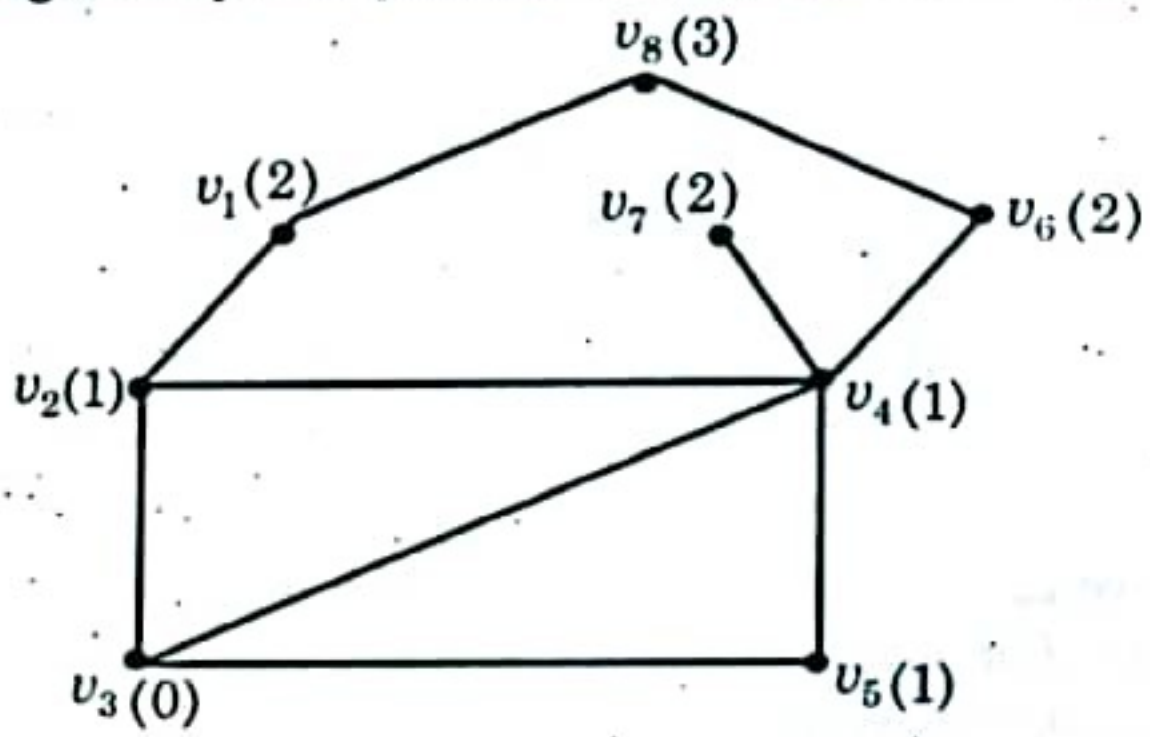
Illustrative Examples.

Find by BFS Algorithm a spanning tree in the following graph :

[WBUT 2012]



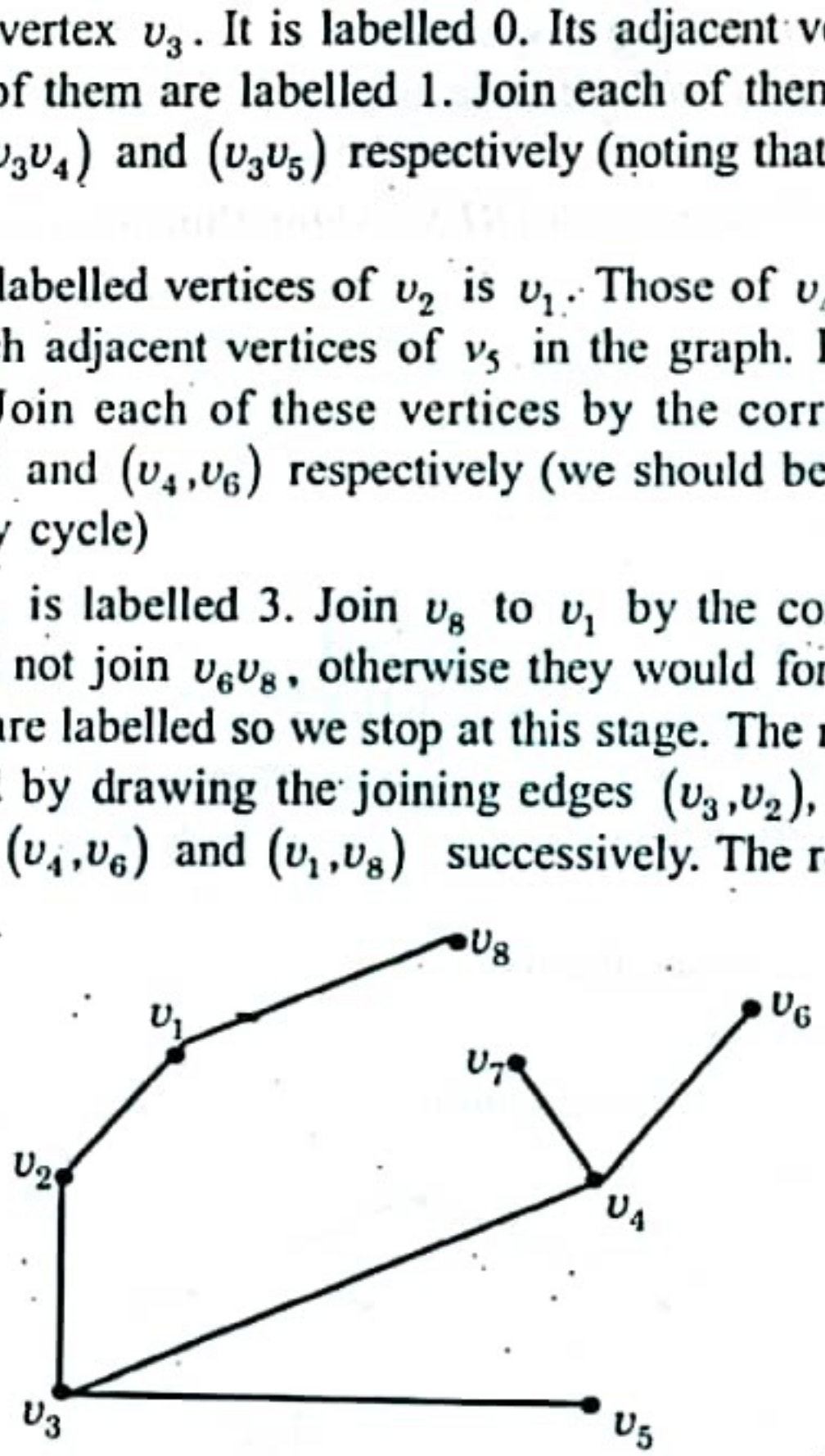
Discarding all loops and parallels we have the following graph :



Choose the vertex v_3. It is labelled 0. Its adjacent vertices are v_2, v_1 and v_5. Each of them are labelled 1. Join each of them with v_3 by the edges (v_3v_2), (v_3v_4) and (v_3v_5) respectively (noting that they do not form a cycle).

Adjacent unlabelled vertices of v_2 is v_1. Those of v_4 are v_7 and v_6. There is no such adjacent vertices of v_5 in the graph. Each of v_1, v_7, v_6 is labelled 2. Join each of these vertices by the corresponding edges $(v_2,v_1), (v_4,v_7)$ and (v_4,v_6) respectively (we should be careful that they do not form any cycle)

Similarly v_8 is labelled 3. Join v_8 to v_1 by the corresponding edge (v_1,v_8). We do not join v_6v_8, otherwise they would form a cycle. Since all the vertices are labelled so we stop at this stage. The required spanning tree is obtained by drawing the joining edges (v_3,v_2), (v_3,v_4), (v_3,v_5) $(v_2,v_1), (v_4,v_7), (v_4,v_6)$ and (v_1,v_8) successively. The required Spanning Tree is

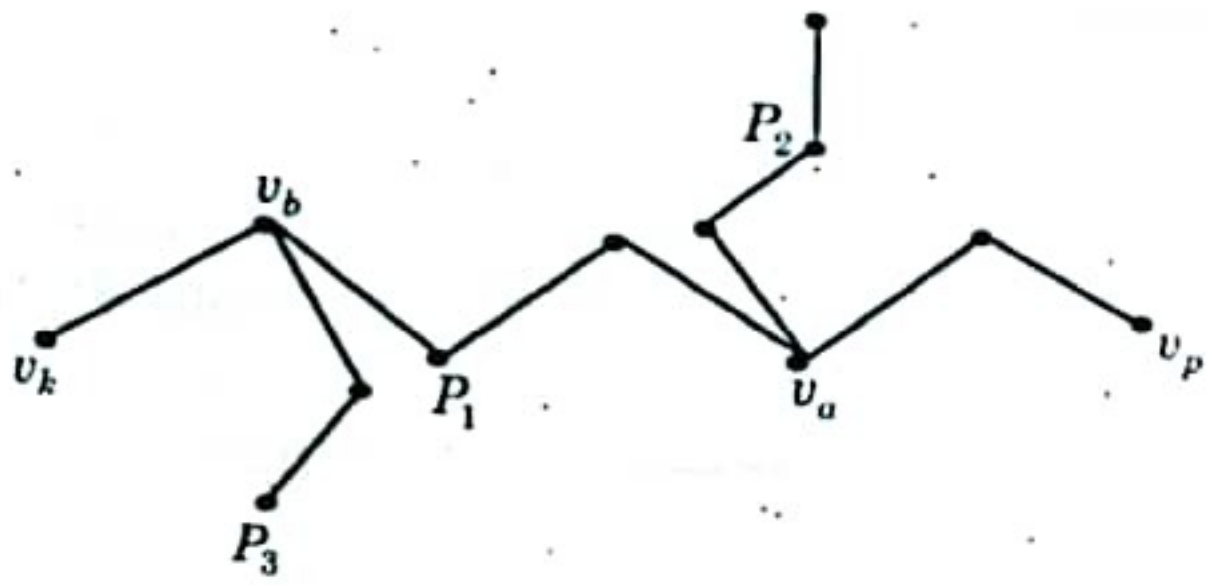


5.3.7. Depth First Search Algorithm for construction of a spanning Tree.

Discard all parallels and loops from the graph. Choose any vertex v_k of the graph. Make a path starting from v_k as long as possible by

successively adding edges and vertices. Let this path be $P_1 : v_k - v_p$.

Now backtrack (along P_1) from v_p and let v_a be the first vertex starting from which we can make a path (as long as possible) containing no vertices of P_1

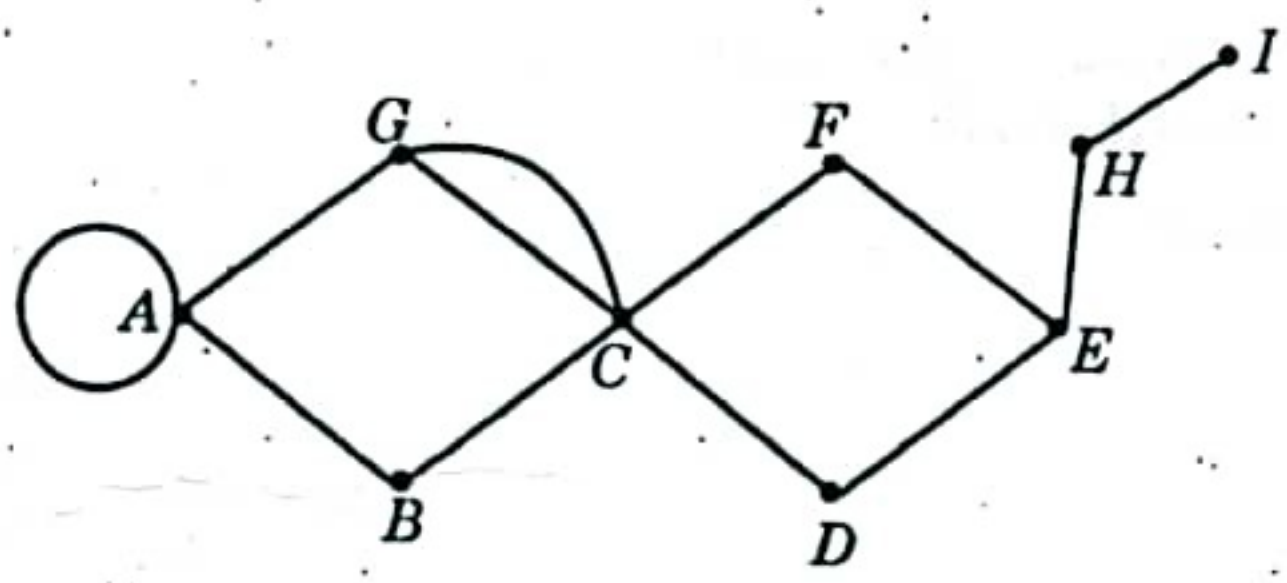


Let v_b be the next vertex reached along this tracking from which we can make another path (as long as possible) containing no vertices of P_1 and P_2.Continuing this process we get paths P_3, P_4,⋯. We stop at that stage when each of the vertices of the graph is included in some of the paths P_1, P_2, P_3,⋯.These paths together form the required spanning tree.

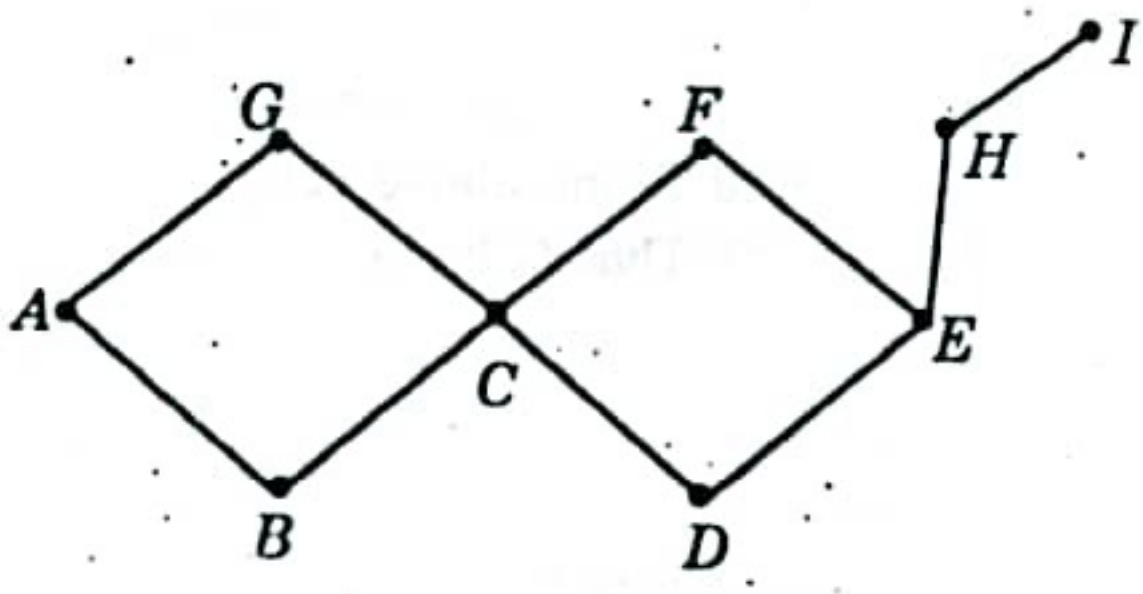
To be more precise see the following Illustrative Example.

Illustrative Example.

Ex. *Find by DFS Algorithm a spanning tree in the following graph.*

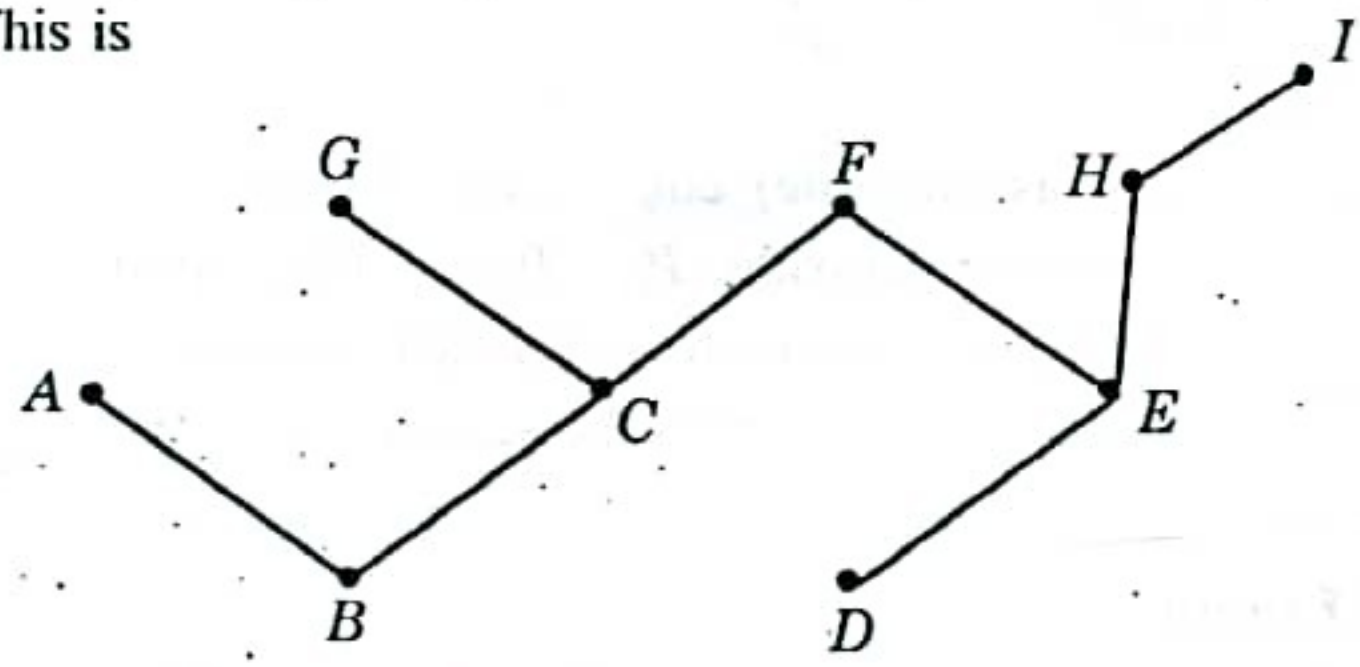


Discarding all loops and parallels we have the following graph.



Choose arbitrarily the vertex G. Make a path starting from G as long as possible by successively adding edges and vertices. This path be $P_1 : G, C, F, E, H, I$. Now backtrack from I to H. We get no path starting from H. Next we backtrack H to E. We get a path $P_2 : E, D$ noting that P_2 does not make any cycle with the path P_1. Again from E we re-track to F and then to C. There exists a path from C, say $P_3 : C, B, A$ such that all of its vertices are not included in the previous two paths. Since the three paths P_1, P_2 and P_3 contain all the vertices of the graph, we stop at this stage.

The required spanning tree is formed by these three paths P_1, P_2 and P_3. This is



Note. (1) *This method is also known as backtracking method.*

(2) *That all the vertices of the graph would be included in this method can be proved as a theorem.*

5.3.8. Theorems on Spanning Tree.

Theorem 1. A graph G has a spanning tree if and only if G is connected.

[*W.B.U.Tech 2010, 2005*]

Proof. Let G be a connected graph. If G has no cycle then it is its own spanning tree. If G contains a cycle then delete an edge from that cycle. According to a previous Theorem the graph is still connected. If this graph has no cycle then it becomes a spanning tree of G ; otherwise repeat the operation till an edge from the last cycle is removed — giving a connected graph without any cycle. This left out graph contains all vertices of G, because no vertices were removed in the above deletion process. So this left out tree is a spanning tree of G. Thus G has a spanning tree.

Conversely, let G be a graph having a spanning tree, say T. Let v_1 and v_2 be two arbitrary vertices of G. Since T contains all vertices of G therefore v_1 and $v_2 \in T$. Since T is a tree so T is connected and so v_1 and v_2 are connected by a path. So, G is connected.

Theorem 2. Every tree in a connected graph can be extended to a spanning tree of the graph.

OR

If T be a tree of a connected graph G then there exists a spanning tree S of G such that $T \subseteq S$.

Proof. G is the connected graph ; T is a tree of G. If G has no cycle then it is its own spanning tree and since $T \subseteq G$ the result is proved.

If G contains a cycle then there exists an edge of the cycle which is not an edge of T since T is tree. Delete this edge from that cycle. According to a theorem the graph is still connected and contains T. If this graph has no cycle then it becomes a spanning tree of G containing T ; otherwise we repeat the operation till an edge from the last circuit, which is not an edge of T, is removed. This left out graph contains all vertices of G because no vertices of G are removed in the above process of deletion of edges. So this left out tree say S is a spanning tree of G and $T \subseteq S$.

Theorem 3. Let T be a spanning tree in a connected graph G. G has n vertices and e edges ; then T has $n-1$ branches and $e-n+1$ chords.

Proof. This follows from the definition of branch and chord and from the fact that a tree with n vertices contains $n-1$ edges.

Note. *If chords of a spanning tree are removed from a graph then we get the spanning tree. So at least* $e-n+1$ *number of edges to be removed to get a graph free from any cycle.*

Theorem 4. The total number of spanning tree in a complete graph with n vertices is n^{n-2}.

Proof. Beyond the scope of the book.

5.3.9. Illustrative Examples.

Ex. 1. *Find the minimum number of cables to be removed from an electric network with 1000 connecting cables joining 500 nodes so that all cycles are removed though connection among the nodes are unhampered.*

Here the No. of vertices, $n = 500$; the No. of edges, $e = 1000$ So at least $e - n + 1 = 1000 - 500 + 1 = 501$ number of cables to be removed to get the network without any cycle.

Ex.2. *Prove that a pendant edge in a connected graph is contained in every spanning tree of it.*

Let e be a pendant edge of the connected graph G. Let v be one end of e whose degree is 1. Let S be a spanning tree. Let, if possible, S does not contain e. Since e is the only edge containing v so $v \in S$. This contradicts the fact that a spanning tree contains all vertices of the graph. Hence proved.

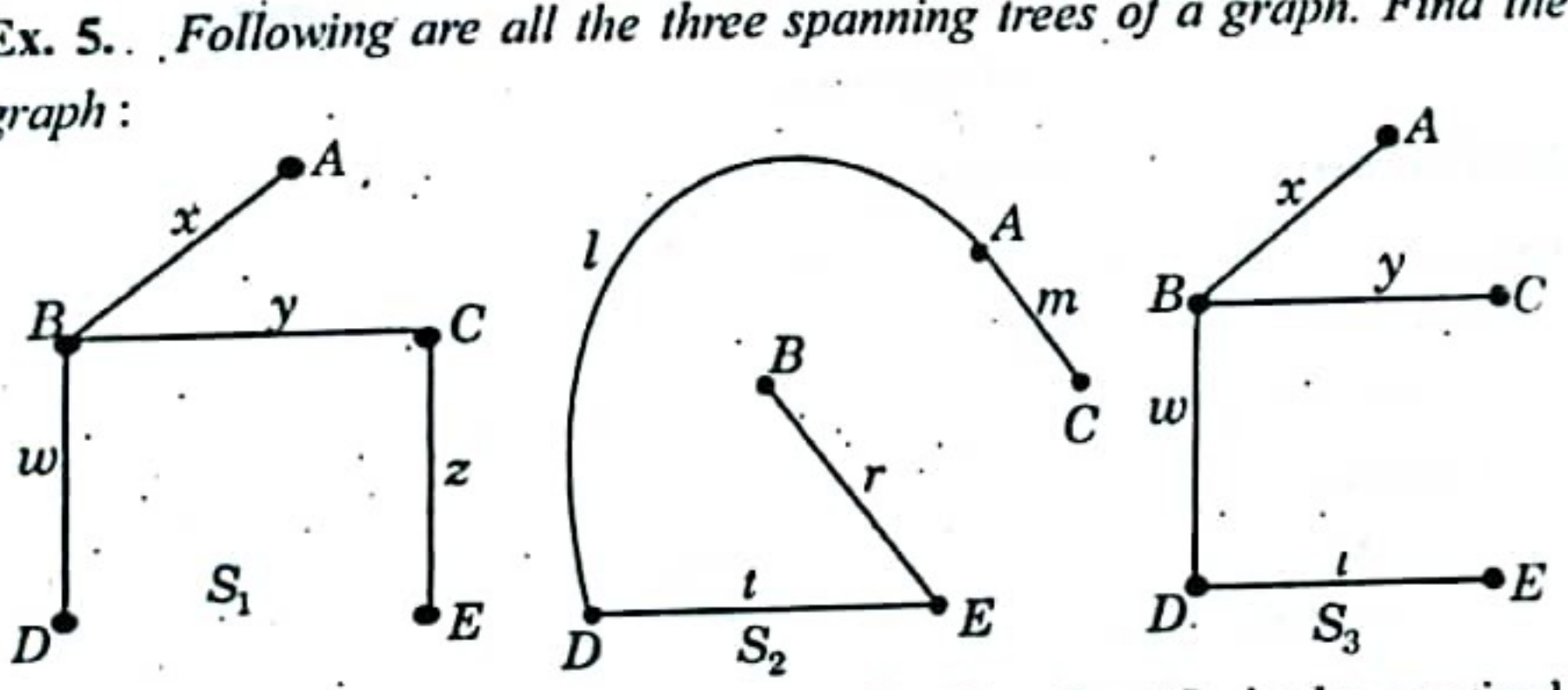
Ex. 3. *Prove that any cycle in a graph must have at least one edge in common with a co-tree (chord set) of the graph.*

Let G be a graph and C be an arbitrary cycle of G. Let T be a tree in G. (Every non-null graph must have at least one tree). $\overline{T}$ is co-tree of T. Since a tree does not contain a cycle so $C \not\subset T$, i.e., there exist at least one edge say e of C which is not an edge of T. So e is a chord of T, i.e., $e \in \overline{T}$. Thus e is an edge in common with the co-tree of T.

Ex. 4. *Union of all the spanning trees of a graph, without any self loop, is the graph itself.*

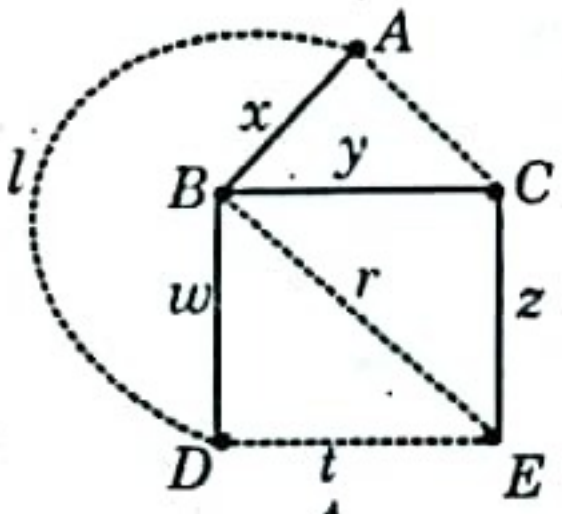
Let S_i represents the spanning tree of a graph G. We shall prove that $G = \bigcup_i S_i$. Since $S_i \subseteq G$ for all i, so $\bigcup_i S_i \subseteq G$. On the other hand every vertex of G belongs to S_i ($\because$ S_i contains all vertices of G). Let e be an arbitrary edge of G and u, v are two end vertices of e where $u \neq v$. As e is not a self loop, so e is a tree. By an earlier theorem, there exists a spanning tree (of G) which contains e. Thus $e \subseteq S_i$ for some i. So $e \in \bigcup_i S_i$. Thus $G \subseteq \bigcup_i S_i$. Hence $G = \bigcup_i S_i$.

Ex. 5. *Following are all the three spanning trees of a graph. Find the graph :*



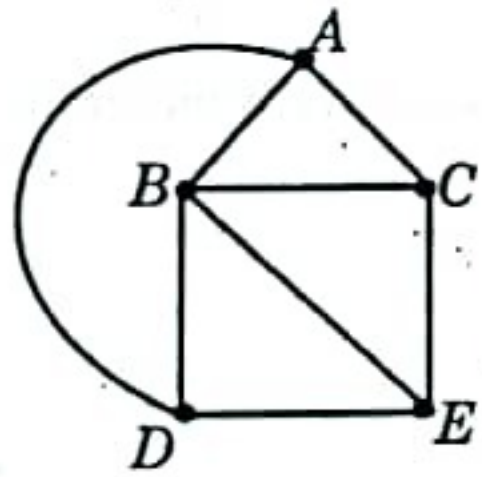
From the previous example we see $G = S_1 \cup S_2 \cup S_3$ is the required graph.

Now, $S_1 \cup S_2 =$



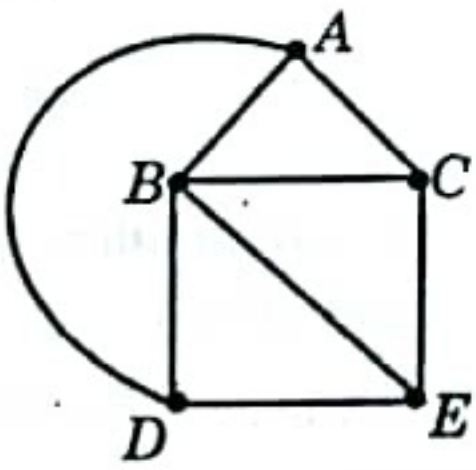
(Super impose of S_2 on S_1 is shown by the dotted edges)

$=$



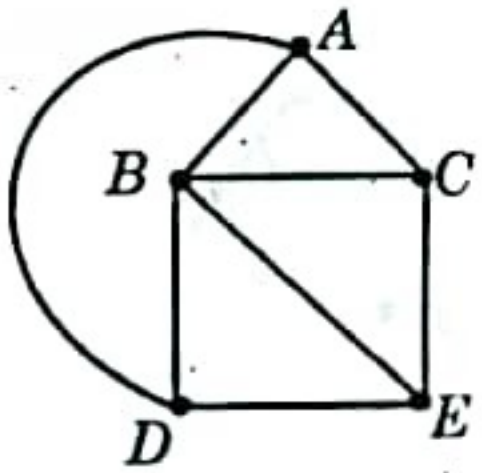
(here the dotted lines are converted to plain ones)

Then $S_1 \cup S_2 \cup S_3 =$



(there are no dotted to be be super imposed)

So the required graph is $G =$



5.3.10. Finding all Spanning Trees in a Graph.

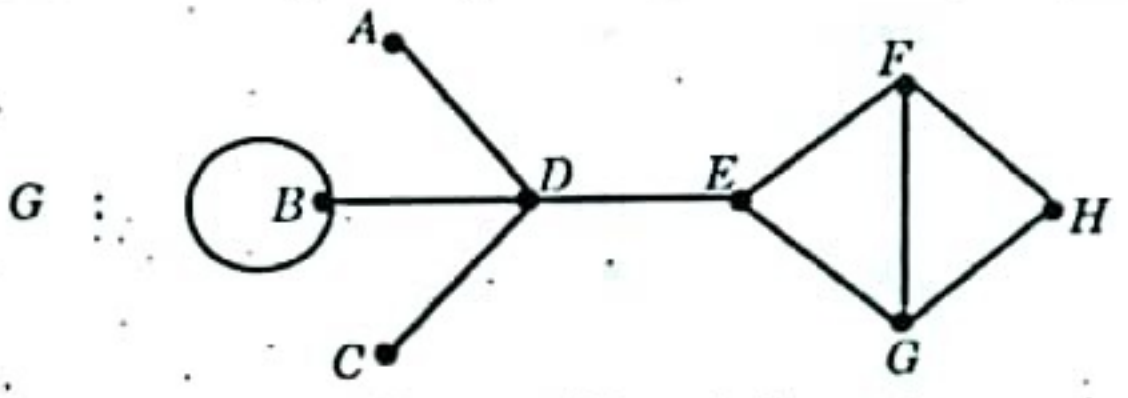
As we have stated earlier finding of all spanning trees in a graph is must for practical purpose. In Art 5.3.6 and 5,3.7 we have shown the procedure of finding one spanning tree in a graph. In this section we discuss the procedure of finding all the spanning trees starting from one spanning tree which is already found.

Theorem. Starting from any spanning tree of a graph G, every spanning tree of G can be found by successive cyclic exchanges.

Proof. Beyond the scope of the book. But we show the process of cyclic exchange in the following illustration .(Shown in the next page)

Illustration of Cyclic Exchanges.

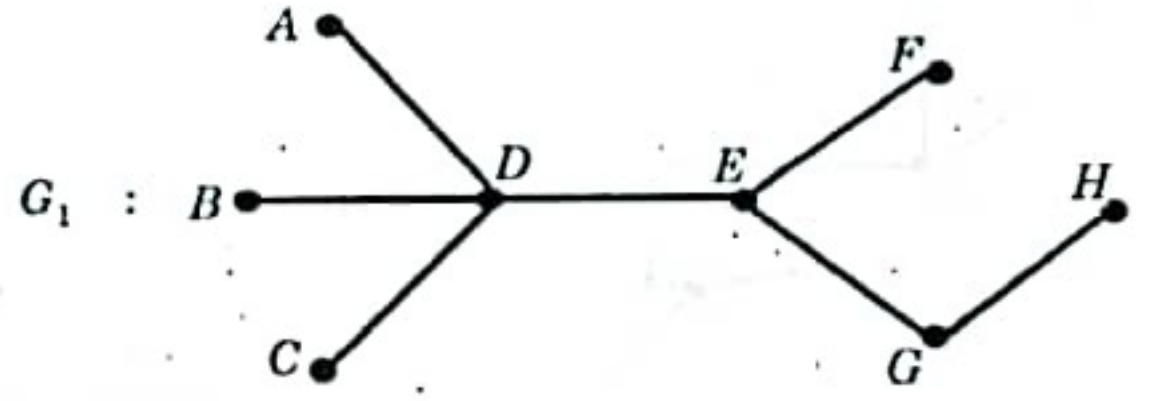
Ex. 1. *Find all the spanning trees of the connected graph,*



[W.B.U.Tech 2007]

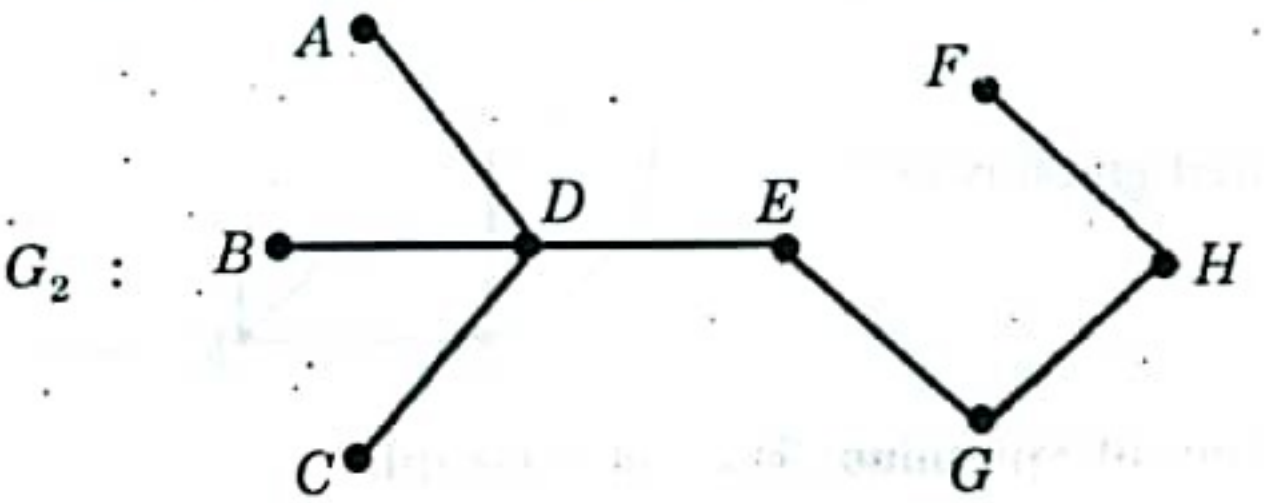
First remove all self loops (if any) from the graph.

At a glance we see the graph

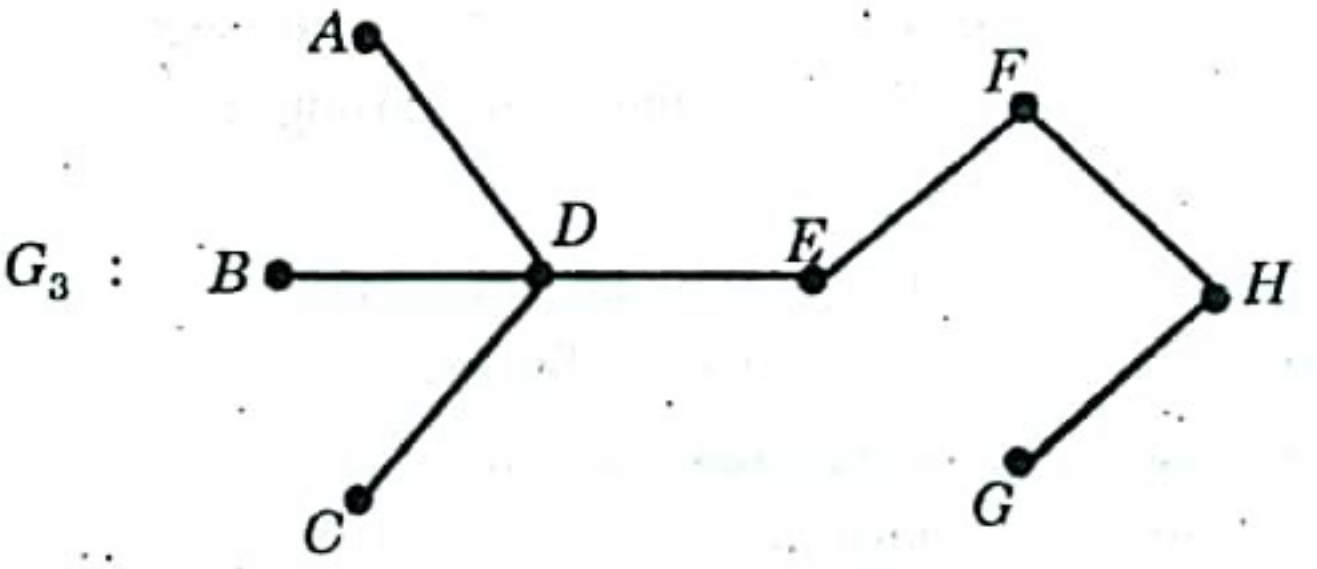


is a spanning tree of G. We shall find all other spanning tree starting from G_1. G_1 has two chord (F, H) and (F, G).

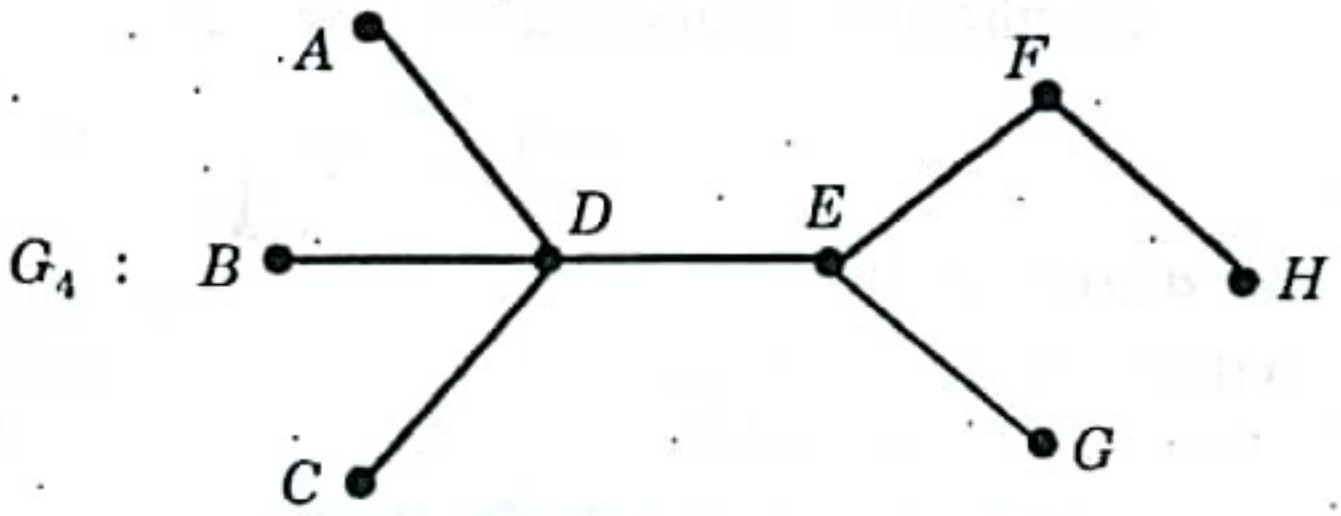
Add the chord (F, H) to G_1. A cycle line $EFHGE$ is formed. Remove the branch EF from this cycle and we get a spanning tree



Remove the branch EG from the cycle $EFHGE$ and we get a spanning tree

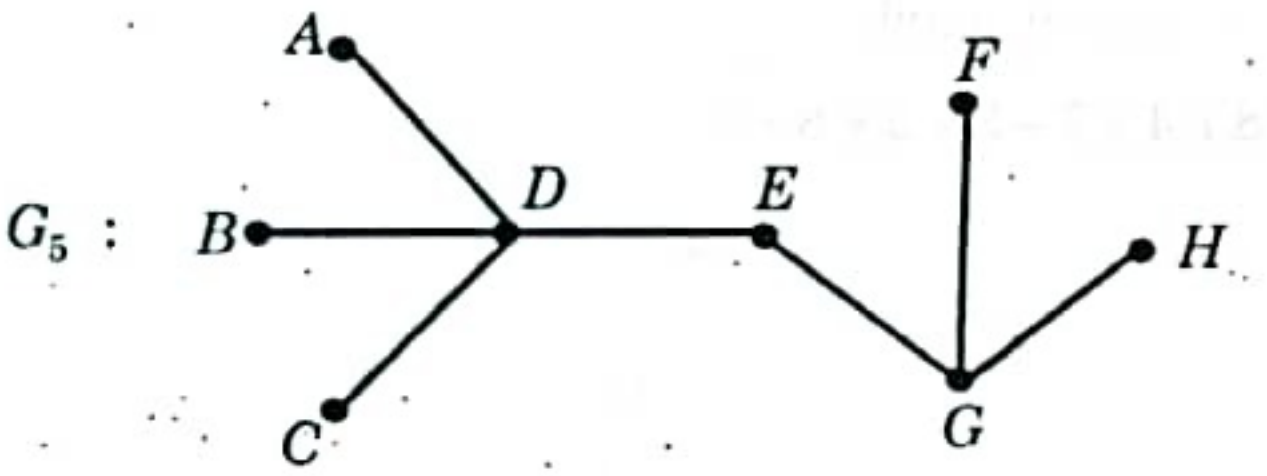


Remove the branch GH from the same cycle and we get a spanning tree

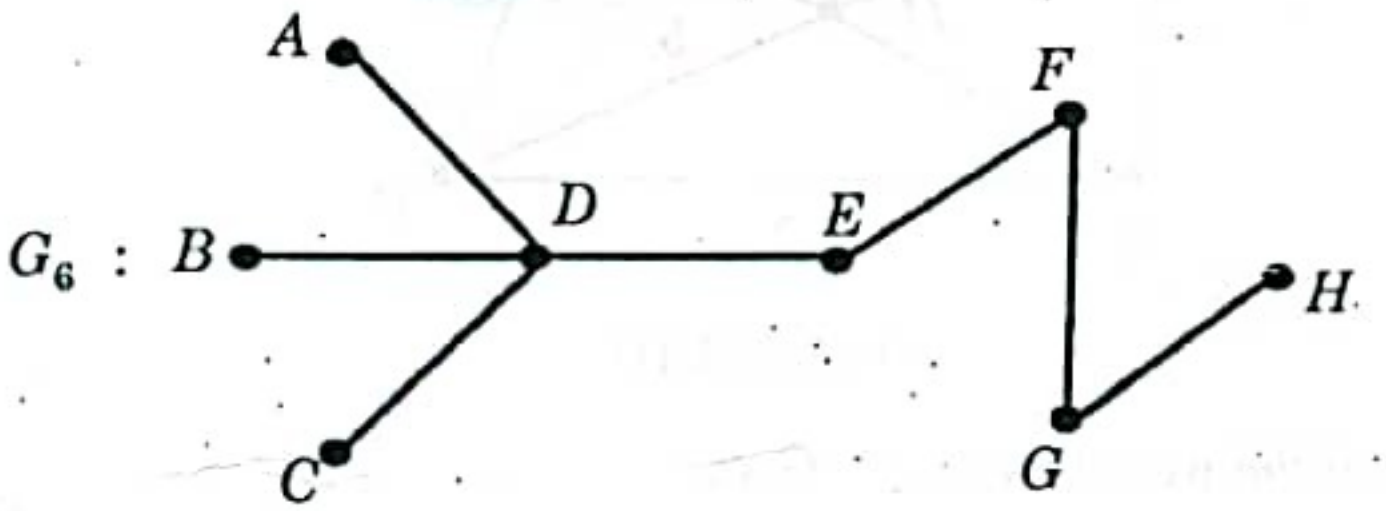


Next add the chord (F, G) to G_1. A cycle $FEGF$ is formed.

Remove the branch EF from this cycle and we get the spanning tree



Remove the branch EG from the cycle $FEGF$ and get the spanning tree



Thus all the spanning trees of G are found which are G_1, G_2, G_3, G_4, G_5 and G_6. These are shown above.

5.3.11. Weight of an edge and Weighted Graph.

Sometimes a real number is associated with each edge of a graph ; this number represents weight of the corresponding edge.

A graph each of whose edge bears a weight is called weighted graph.

Sum of the weights of all edges of a graph is called the weight of the graph.

Weight of a Tree.

The sum of the weigths of all edges of a tree is called the weight of the tree.

Illustration.

Ex. 1. *We show a graph in Fig.5.3.10.*

Here the vertices $A, B, C, \cdots$ represent the cities of a country. The edge say $(A\,C)$ represents the road from the city A to city C and so on. Let the cost of transporation from A to C be Rs. 10,000. Then we suppose the edge $(A\,C)$ bears a weight 10. This is inserted beside the side AC. Similarly other edges bear the corresponding transpotation cost as their weight. Here the weight of graph

$$= 10 + 8 + 4 + 7 + 3 + 5 + 8 + 9 + 3 = 57\,.$$

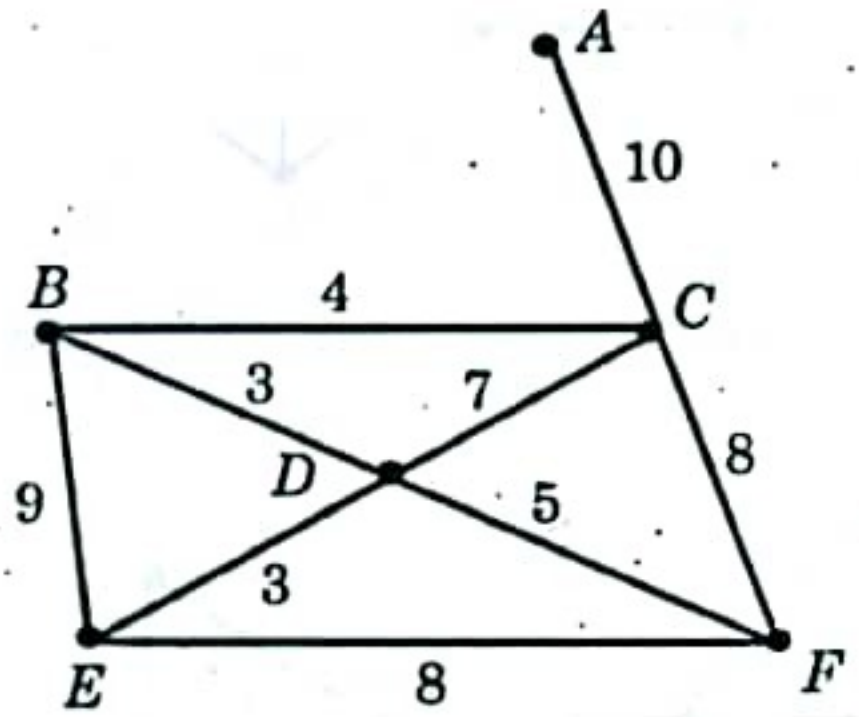
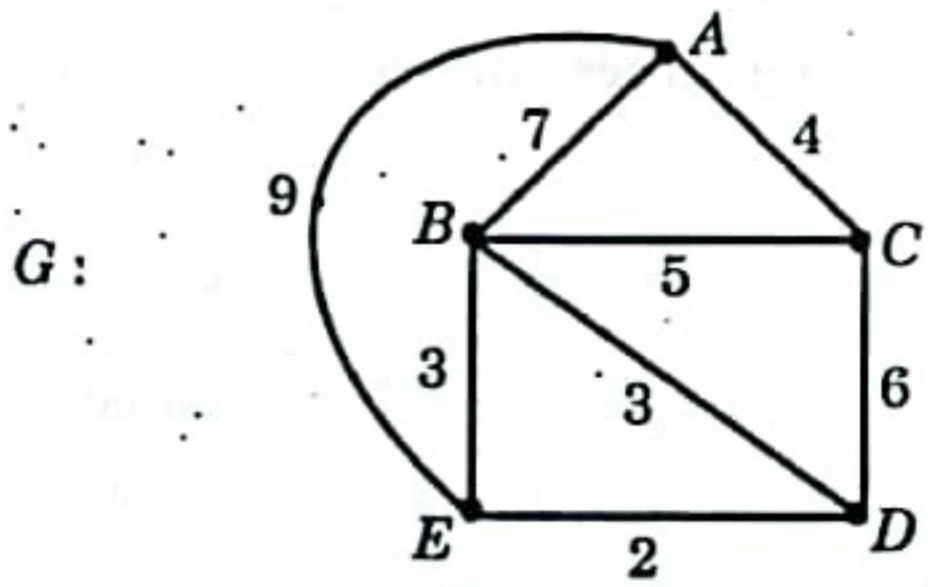


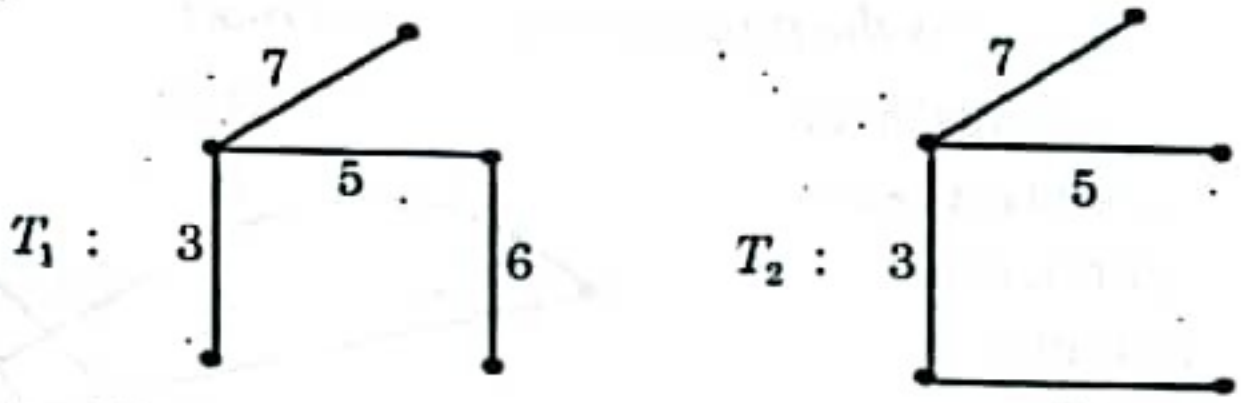
Fig.5.3.10

Ex. 2. *From the weighted graph G shown below*



pick up two spanning trees and find their weights.

We show two of several spanning trees below (actually we know how to find all the spanning tree. See a previous Illustration)



Weight of $T_1 = 7+5+3+6 = 21$

Weight of $T_2 = 7+5+3+2 = 17$.

Note. (1) *Different spanning tree of a graph has different weights.*

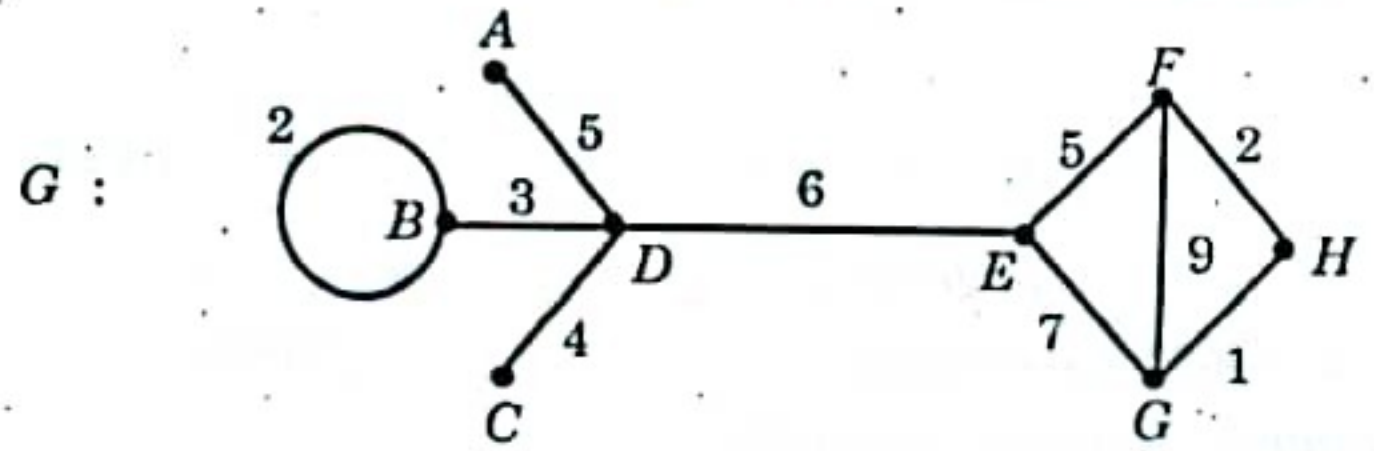
(2) *The spanning tree having smallest weight has practical significance.*

5.3.12. Minimal Spanning Tree (or, Shortest Spanning Tree).

Let G be a weigthed graph. A spanning tree with the smallest weight in G is called a minimal spanning tree.

Illustrations.

Ex. 1. *Find the minimal spanning tree of the weighted graph G where*

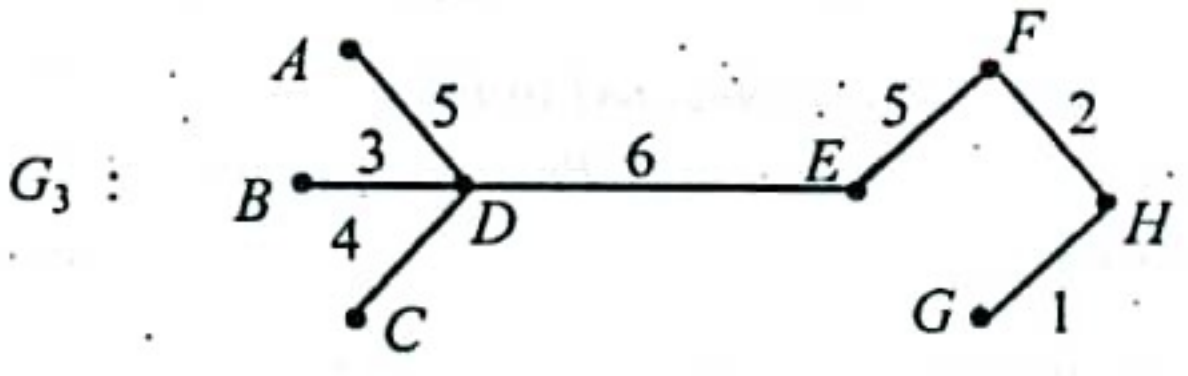


We found all the spanning trees of this graph in Art 4.2.10

The weights of these spanning trees are :

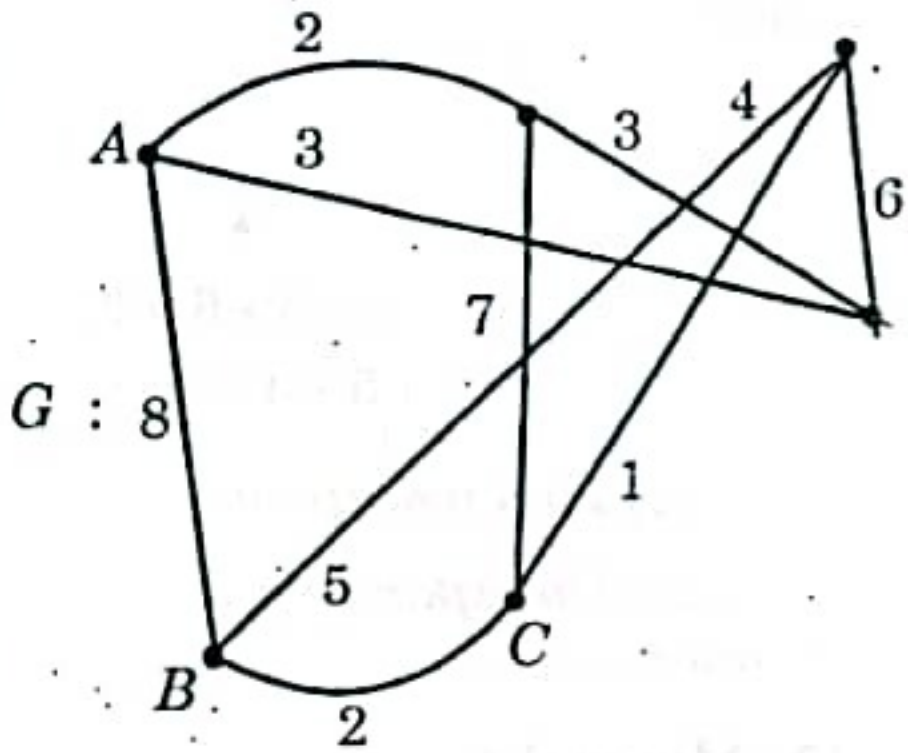
Spanning Tree : G_1	G_2	G_3	G_4	G_5	G_6	
Weight	: 31(5+3+4+6+5+7+1)	28	26	32	35	33

So, the minimal spanning tree is G_3 and



Ex.2. *The graph G represents the proposed network of roads among 6 cities.*

Each vertex represents a city and each edge represents a road to be built up for connecting two cities. The weight of an edge is the expenditure (in crore of rupees) to built the road e.g. the expenditure of road connecting A and B is Rs. 8 crore. If there is no edge between two vertices we understand no direct road can be built connecting the two corresponding cities. For instant, no direct road can be built between A and C. Now the problem is to find the least expensive road net work that connects all the cities together. This is nothing but finding a shortest spanning tree in the above graph G.



Note. (1) *Among all spanning trees in a graph the minimal spanning tree has greater practical significant (See Ex. 2 above)*

(2) *There may have two or more minimal spanning trees.*

(3) *If every edge of a graph G has equal weight then every spanning tree would have equal weight ; so every spanning tree may be treated as minimal spanning tree of G.*

(4) *Minimal spanning tree is sometimes called shorted-distance-spanning-tree.*

(5) *Mathematically the process used in the above example is not suitable for finding shortest spanning tree because there is a great difficulty of finding all the spanning trees in a graph. There are some systemetic procedures (e.g. Kruskal's algorithm, Prim's algorithm etc.) to find such minimal spanning tree. We are just going to discuss these.*

Ex. 3. *Let e be an edge of a connected weighted graph G whose weight is smaller than that of other edges in G. Prove that every shortest spanning tree of G contains the edge e.*

Let; if possible, there exist a shortest spanning tree, say S which does not contain e. So e is a chord of S. So, $S \cup e$ is a fundamental circuit.

Then there exists a branch of S, say f such that $(S \cup e) - f$ is a spanning tree (This is a part of cyclic exchange). Now weight of $(S \cup e - f) =$ wt of S + wt of the edge e - wt of the edge f = wt of S -{wt of edge f – wt of the edge e} < wt of S, since wt of e < wt of edge f. This is a contradiction since S is a shortest spanning tree. Hence proved.

5.3.13. Kruskal's Algorithm of finding Minimal (or, shortest) Spanning Tree.

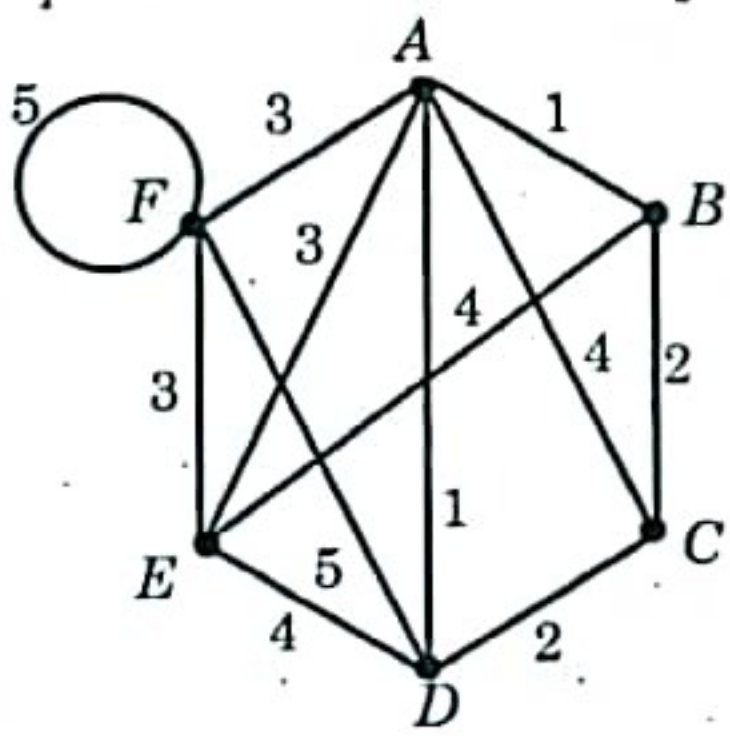
Let G be a given weighted graph with n vertices. We have to find a shortest spanning tree in G. We make a list of all edges (which do not form a loop) of G in order of non-decreasing weight. From this list we select an edge having smallest weight. From the remaining edges select an edge with smallest weight, that makes no cycle with the previously selected edges. Continue this successive steps of selection until $n-1$ edges are selected ($\because$ a tree with n vertices contains $n-1$ edges). These edges constitute a minimal spanning tree in G.

Note : *The validity of this algorithm follows from a well-known theorem (not given here).*

Illustrative Example (on Kruskal's Algorithm)

Ex. 1. *Find by Kruskal's Algorithm a minimal (i.e. shortest) spanning tree from the following graph G.* [*WBUT 2010, 2005*]

G :

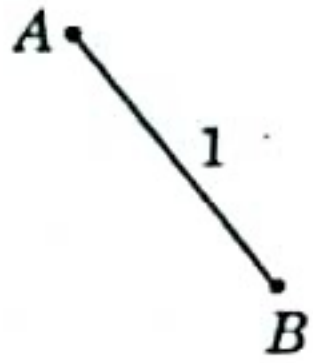


G contains 6 vertices. So the shortest spanning tree contain 5 edges.

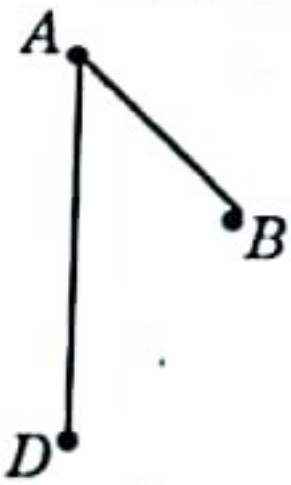
Step 1. List of edges (which do not form loop) arranged in order of non decreasing weight :

edges :	(AB)	(AD)	(BC)	(CD)	(AF)	(AE)	(FE)	(AC)	(EB)	(ED)	(FD)
Weight :	1	1	2	2	3	3	3	4	4	4	5

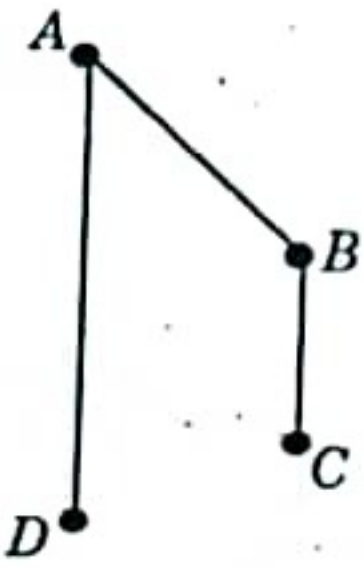
Step 2 : We select the smallest edge (AB) shown as



Step 3 : Select the next smallest edge (AD). We accept it, since it does not form any cycle with (AB). This is shown as

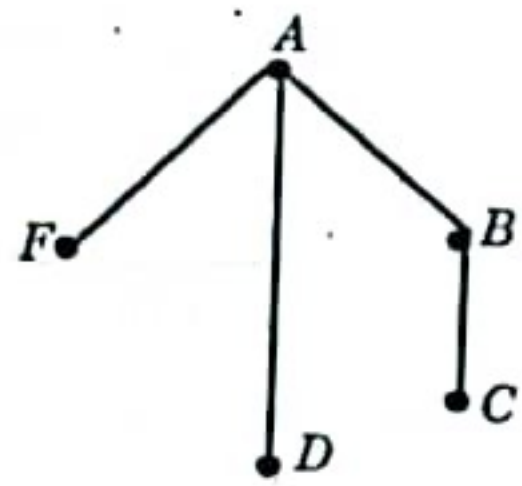


Step 4 : Select the next smallest edge (BC). It does not form any cycle with BAD. So we accept it. This is shown as

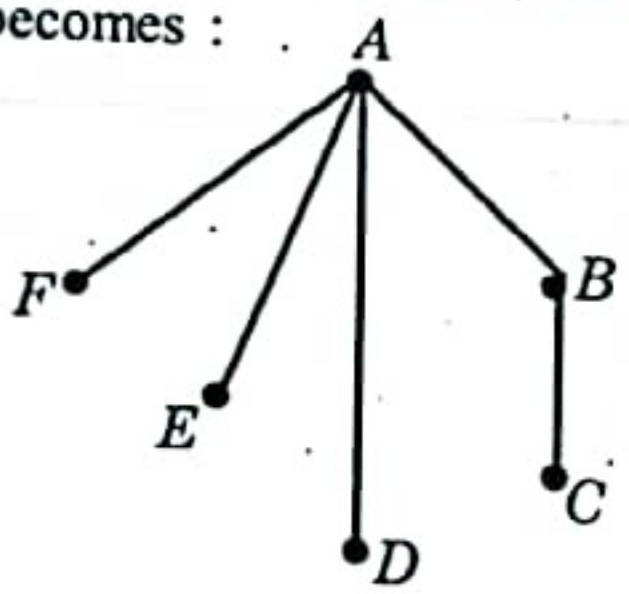


Step 5 : Select the next smallest edge (CD). It forms a cycle with $CBAD$. So we do not accept it.

Step 6 : Select the next smallest edge (AF). It does not form any circuit with $CBAD$. We accept it. The new tree is

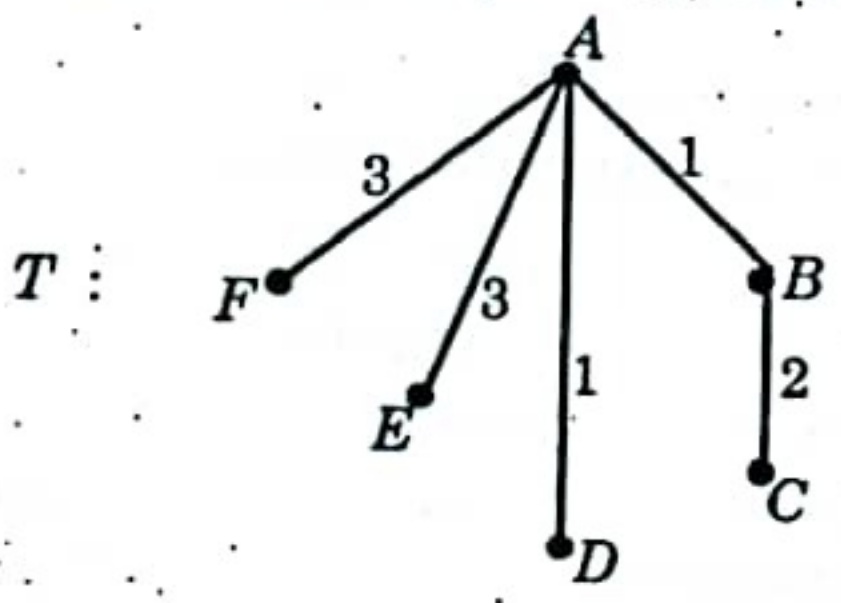


Step 7 : Select the next smallest edge (AE). We accept it (due to similar reason). The new tree becomes :



Since this tree contains $5\,(6-1)$ edges, this is our required smallest spanning tree. We stop at this step.

Thus the required minimal spanning tree is



Weight of this spanning tree $= 1+2+1+3+3 = 10$.

Note. (1) *There exist other alternative shortest spanning tree. If we had chosen (FE) in Step7 in place of (AE) we could have get another shortest spanning tree.*

(2) *Obviously every minimal spanning tree obtained by different alternative selection has same weight.*

5.3.14. Prim's Algorithm of finding Minimal (or, shortest) Spanning Tree.

Let G be a given weighted graph with n vertices. We have to find a shortest spanning tree in G. Remove all self loops if exist. If there exist parallel edges between two vertices then delete them except that having minimum weight. We label the vertices by $v_1, v_2, \cdots, v_n$. Tabulate the weights of the edges of G in a $n \times n$ table. We suppose that weight of the edge joining v_i to v_i is 0 and if there is no edge connecting v_i and v_j then the weight of the edge $(v_i\ v_j)$ is ∞. Let w_{ij} = weight of the edge from v_i to v_j.

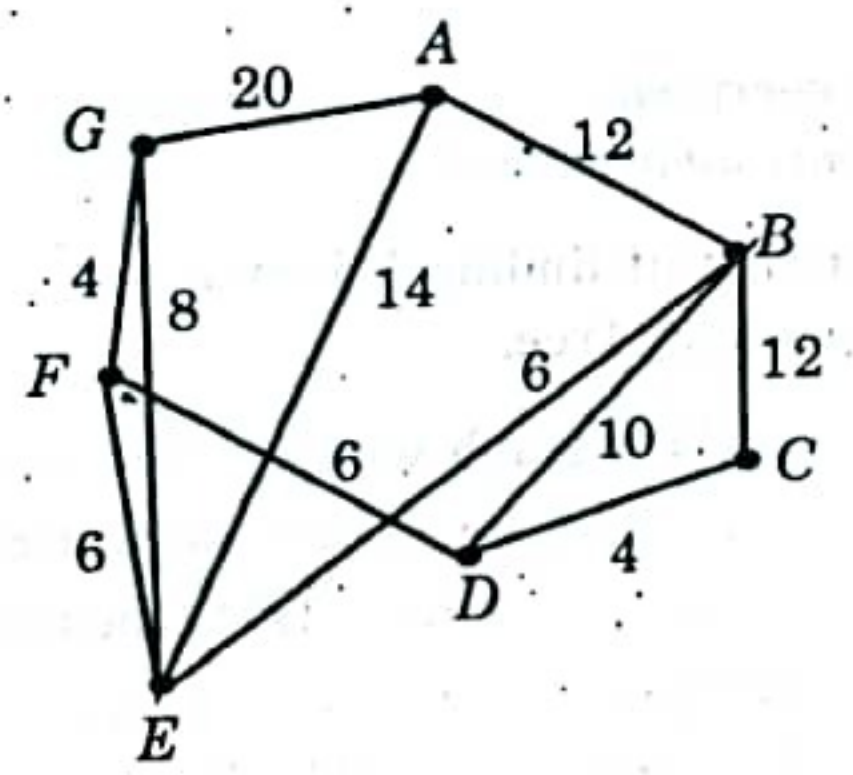
The table is :

	v_1	v_2	v_3	...	...	...	v_{n-1}	v_n
v_1	0	w_{12}	w_{13}	...	...	...	w_{1n-1}	w_{1n}
v_2	w_{21}	0	w_{23}	...	...	...	w_{2n-1}	w_{2n}
.			...	...	...			
.								
.								
.								
v_n	w_{n1}	w_{n2}	w_{n3}	...	...	...	w_{nn-1}	0

Now, start from vertex v_1. Connect it to its nearest neighbour, i.e., to the vertex $(\neq v_1)$ which has smallest entry in 1st row of the table. Let v_k be this vertex. Thus a tree v_1, v_k is formed. Next connect this tree to the vertex $(\neq v_1, v_k)$ that has smallest entry in 1st row and k-th row. Let v_j be this new vertex. Then a tree v_1, v_k, v_j is formed. Continue this successive steps of selection of vertices until $n-1$ edges are selected ($\because$ a tree with n vertices contains $n-1$ edges). These edges constitute a shortest spaning tree.

Illustrative Example. (On Prim's Algorithm)

Find by Prim's Algorithm a minimal spanning tree from the following graph [*W.B.U.T. 2015*]

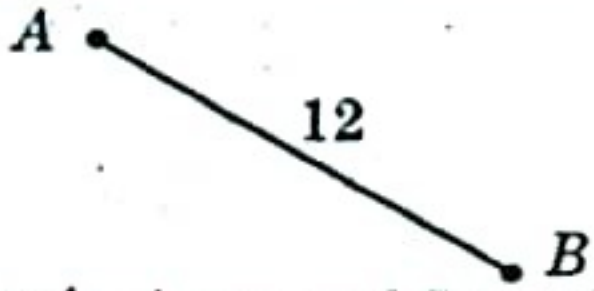


The given graph contains 7 vertices. So any spanning tree contains $7-1=6$ edges. The weight of each edge of the graph is tabulated in the following 7×7 table. Supposing the weight of the edge joining a vertex

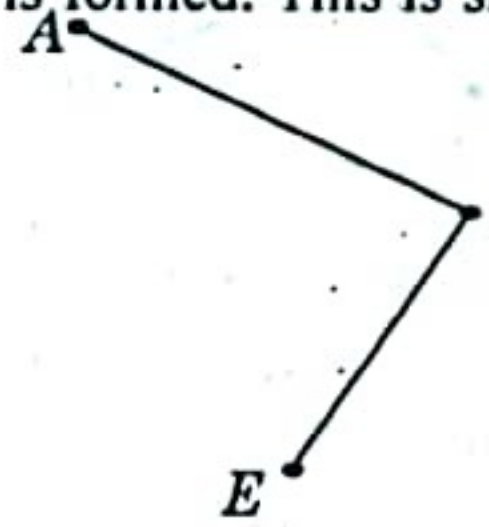
to itself is 0 and the weight between two vertices which are not directly connected by any edge is ∞.

	A	*B*	*C*	*D*	*E*	*F*	*G*
A	0	12	∞	∞	14	∞	20
B	12	0	12	10	6	∞	∞
C	∞	12	0	4	∞	∞	∞
D	∞	10	4	0	∞	6	∞
E	14	6	∞	∞	0	6	8
F	∞	∞	∞	6	6	0	4
G	20	∞	∞	∞	8	4	0

Step 1 : Start from vertex *A*. Smallest entry in *A Row* i.e. 1st row of the table is 12. This corresponds the vertex *B*. (Though the smallest entry was 0 we do not select it because in that case the corresponding vertex becomes A again.) Thus the tree *AB* is formed. This is shown as

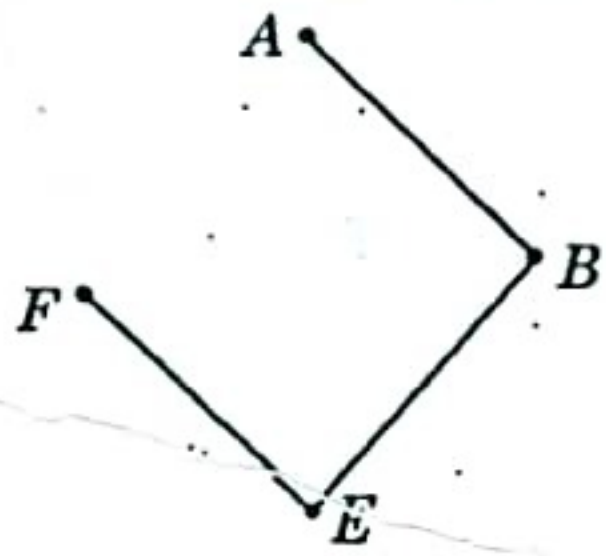


Step 2. Smallest entry in *A*-row, and *B*-row i.e. in 1st and 2nd row is 6. (We do not select 0 due to same reason.) This corresponds other distinct vertex *E*. The tree *ABE* is formed. This is shown as

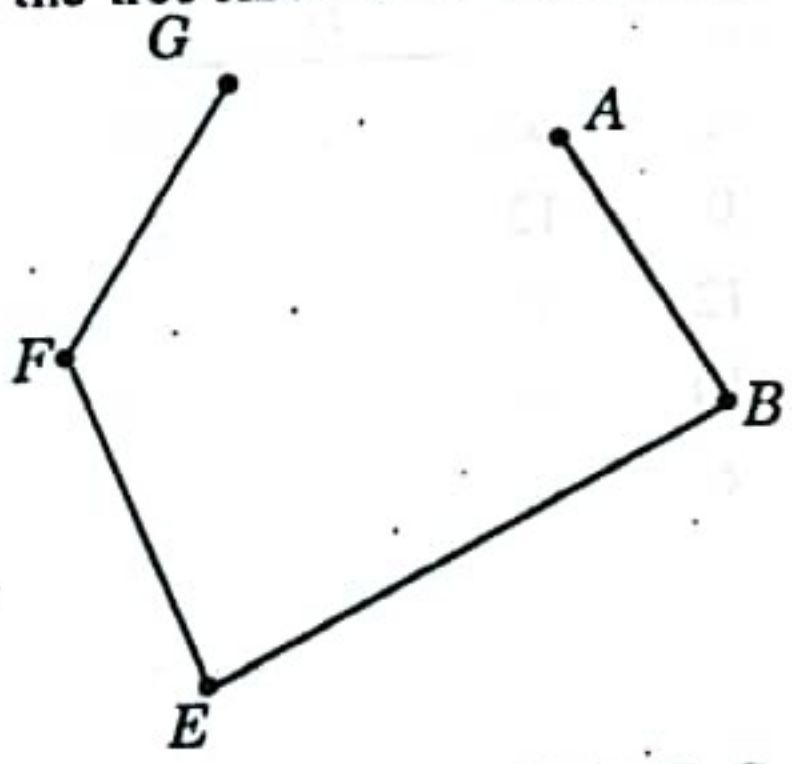


Step 3. Smallest entry in *A*-row, *B*-row and *E*-row is 6 which corresponds the vertex *F*.

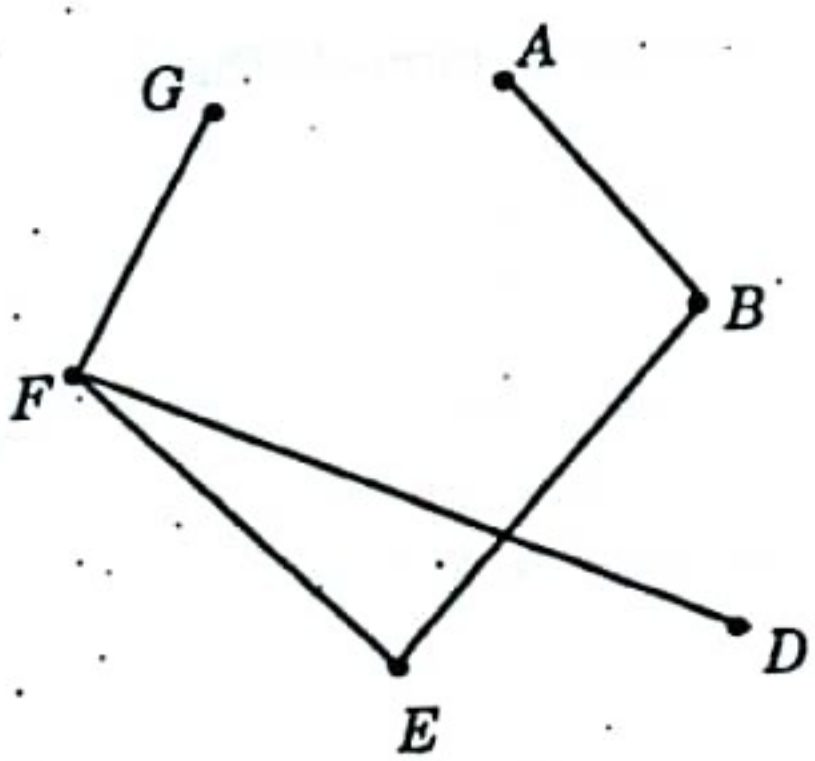
We do not select the other '6' because in that case the vertex *B* would be again picked up. The tree *ABEF* is formed :



Step 4: The minimum entry in A, B, E, F -rows is 4 which corresponds G. Thus the tree $ABEFG$ is formed as follow



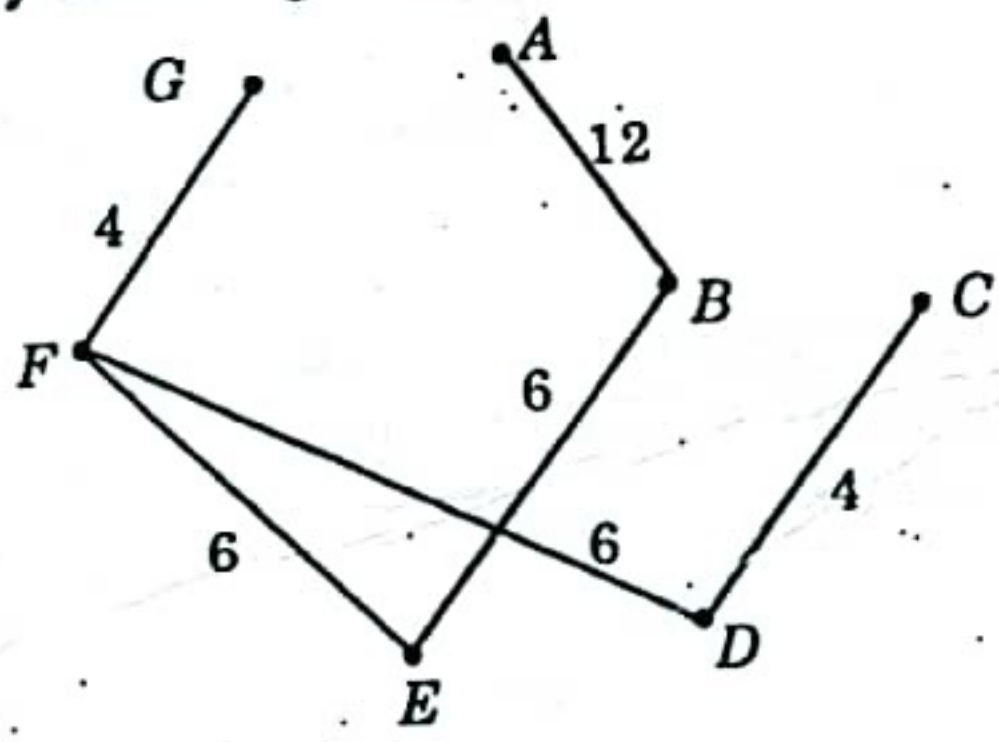
Step 5 : The minimum entry in A, B, E, F, G row is 6 (in F-row) corresponding the new vertex D. Thus the tree.



is formed

Step 6 : The minimum entry in A, B, E, F, G and D row is 4 (lying in D-row). This corresponds the new vertex C. So the above tree is extended to other by connecting C to D.

T :



We stop at this step because we get a tree with 6 edges. So the tree T is the required minimal spanning tree.

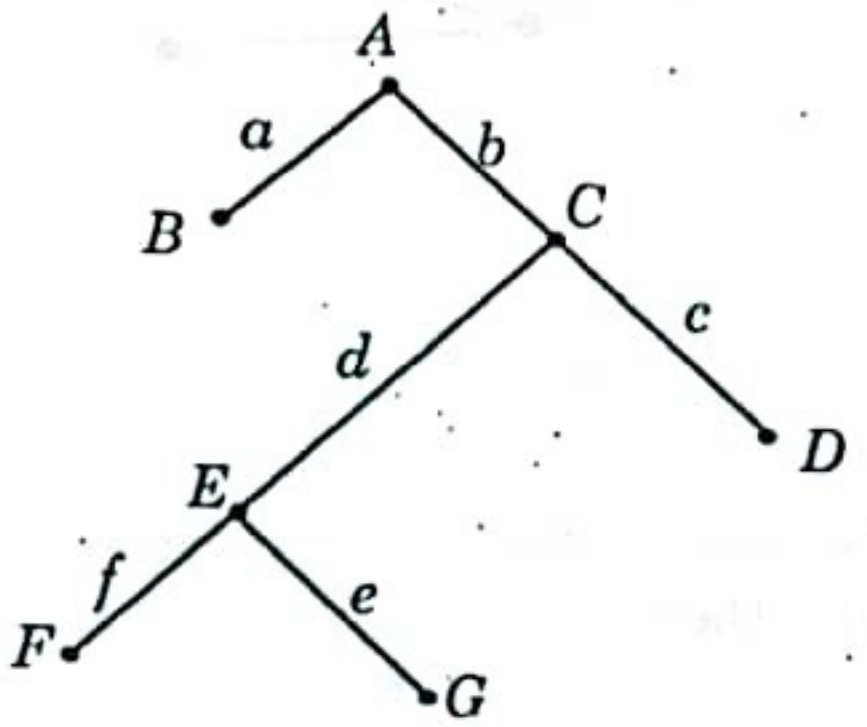
Weight of this spanning tree $= 4+6+6+12+6+4 = 38$.

Note. *Alternative shortest spanning tree may exists in the above example.*

5.3.15. Miscelleneous Examples.

Ex. 1. *Cite a situation in Business-Marketing that can be represented by a tree.*

A company which sells its products in Eastern India and Southern India is planning to launch a new product. First they introduce the product in a very small test market in Eastern India. If the product fails, it is stopped from production. Otherwise it is introduced in all of Eastern India. If the product is success in Eastern India it is introduced in all of Southern India; if not, it is introduced in a small test market in Southern India. Again, if it is successful, it is introduced to entire Southern India. We can use a tree which represents all the possible outcomes of the product introduction process. The tree is shown below :

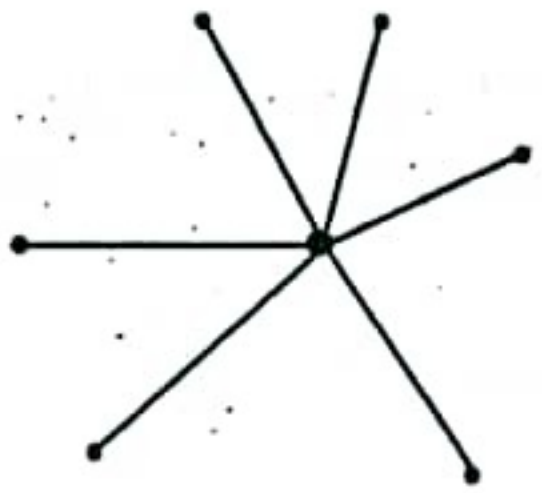


Where $A \equiv$ small test market in Eastern India, $B \equiv$ stopped.

The edge $a \equiv$ failure, $C \equiv$ whole Eastern India, edge $b \equiv$ success, $c \equiv$ success, $D \equiv$ entire Southern India, edge $d \equiv$ failure, $E \equiv$ small test market in Southen India, edge $e \equiv$ success, $G \equiv$ Entire Southen India, $f \equiv$ failure and $F \equiv$ stopped the product.

Ex. 2. *Draw two distinct (i.e. non-isomorphic) trees with 7 vertices.*

The two trees T_1 and T_2 shown below are the required trees :

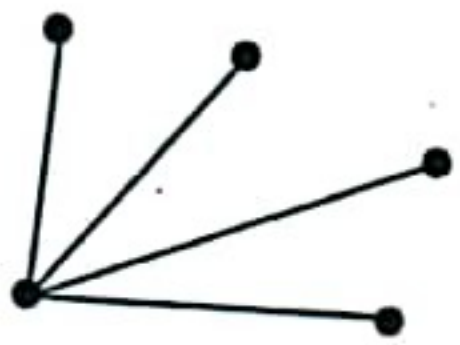
T_1 :  and T_2 : 

T_1 , T_2 are not isomorphic because T_1 has a vertex of degree 6 whereas T_2 has no vertex of degree 6.

Ex. 3. *Draw a tree with five vertices and total degree 8.*

We know total degree $= 2 \times$ No. of edges $\therefore \quad 8 = 2 \times$ No. of edges

$\therefore$ No. of edges $= 4 = 5 - 1$. So this type of trees do exist. One is

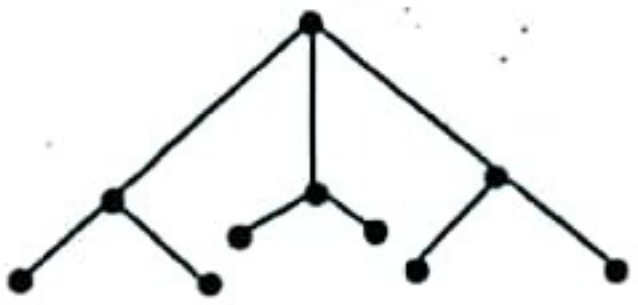


Ex. 4. *Draw a connected graph having six vertices, five edges and a circuit.*

A connected graph which has six vertices and $6 - 1 = 5$ edges is a tree (by Th.7 of Art 5.3:3). A tree can have no cycle. So it is impossible to draw such graph.

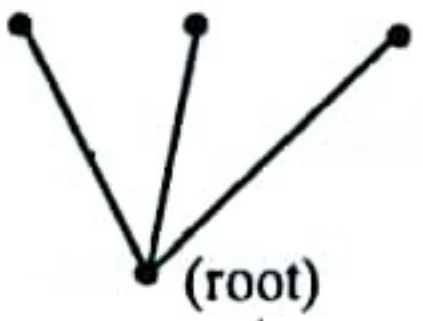
Ex. 5. *Draw a tree with four internal vertices and six terminal vertices*

This is possible. The required graph is

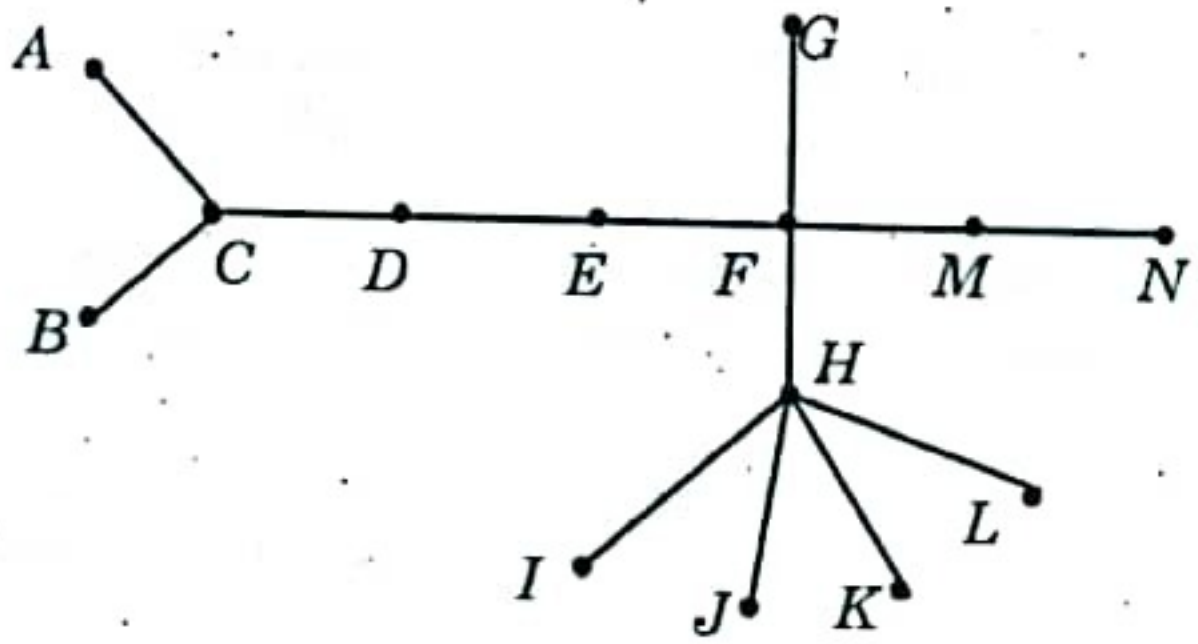


Ex. 6. *Draw a rooted tree having 4 vertices*

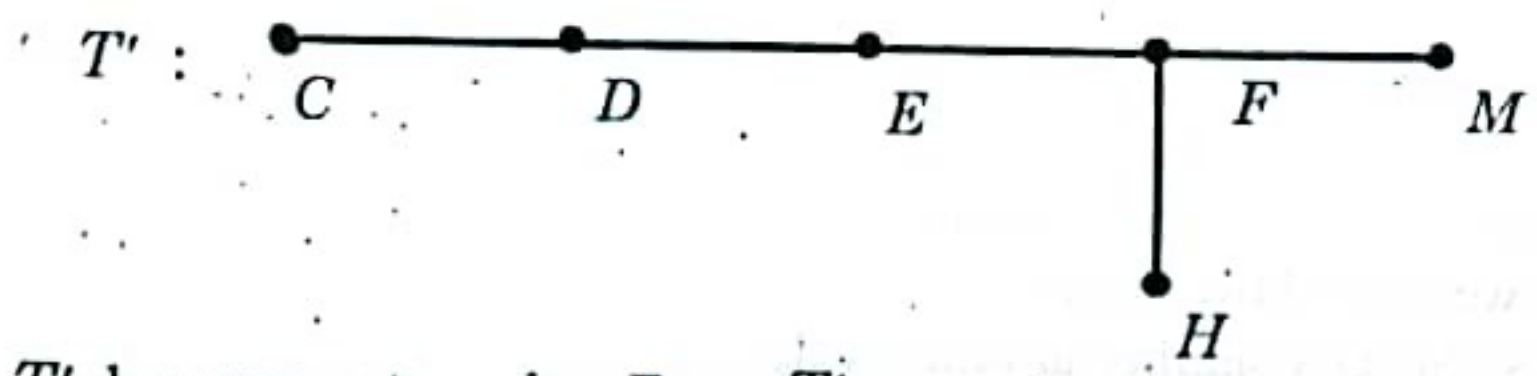
The graph is



Ex. 7. *Find the centre of the following tree*



Let T be the given tree. Remove all pendant vertices from T. The resulting graph T' becomes



T' becomes a tree also. From T' we again remove its pendant vertices and get another tree T''.

From T'' we get T''' by same procedure

T''' : E

E is the centre of the tree.

Ex. 8. *Let T be a tree. Prove that T is a path if and only if the maximum degree of the vertices of T is 2.*

Let T be a path. We know the degree of the origin and terminus vertices of a path is 1 and other intermediate vertices has degree 2. So the maximum degree of the vertices is 2.

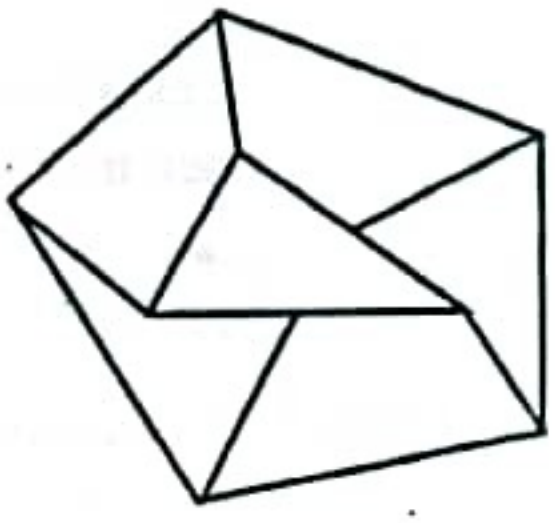
Conversely, suppose the maximum degree of the vertices of T is 2. We know a tree has at least two pendant vertices. Let u and v be the two such vertices of T. Since T is connected so there exists a path P connecting u and v. Let, if possible, there exists an edge say $e \in T - P$. Since T is connected so one end vertex of e say $v_i \in P$. Since $\deg(u) = \deg(v) = 1$ so $v_i \neq u, v$. Now number of incident edges to v_i in P is 2. Since e is also an edge incident to v_i so $\deg(v_i) = 2 + 1 = 3$. This contradicts our hypothesis. So $T - P = \phi$. $\therefore$ $T = P$ $\therefore$ T is a path.

Ex. 9. *Prove that a connected graph G with n vertices and e edges has a unique cycle if and only if* $n = e$.

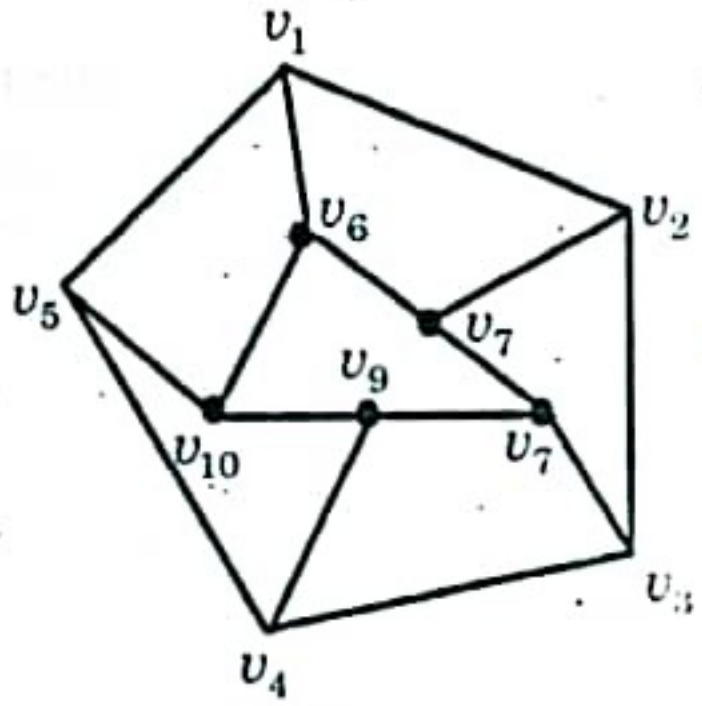
First let the connected graph G has a unique cycle, say C. Delete an edge say e from C ; (end vertices of e are not deleted). Then G becomes a connected graph without any cycle, i.e. $G - e$ becomes a tree with n vertices. So $G - e$ has $n-1$ number of edges. Since only one edge was deleted from G to obtain $G - e$ so G has $(n-1)+1 = n$ number of edges.
$\therefore \quad e = n$.

Conversely let the connected graph G is such that $e = n$. Since G is connected and number of edges $\neq n-1$ so G is not a tree, i.e., G has at least one cycle. Let, if possible, G has two cycle. If we delete the edges e_1 and e_2 from the two cycles respectively (keeping their end vertices intact) we get a subgraph $G - (e_1 \cup e_2)$ which is connected having no cycle. So $G - (e_1 \cup e_2)$ becomes a trees with n vertices. So $G - (e_1 \cup e_2)$ has $n-1$ number of edges. Since two edges were deleted from G so number of edges in G would be $(n-1)+2 = n+1 = e+1$. This is a contradiction. So G cannot have two cycles. Similarly we can show G cannot have more than two cycles. So G contains unique cycle.

Ex. 10. *Suppose we have a firm consisting of plots of land which are full of water (as shown in the figure). How many walls to be broken so that all the water can be drained out ?*



We first represent the problem by a graph as follow : The edges like $(v_1\ v_2)$ or $(v_6\ v_7)$ are walls etc.. If it is converted to a spanning tree then all water will be drained out. Here $n = 10,\ e = 15$



By Theorem 3 of Art 4.2.8, minimum number of edges to be removed to get a spanning tree is

$$e - n + 1 = 15 - 10 + 1 = 6.$$

So minimum 6 walls to be broken to get the water out.

Ex. 11. *The number of internal vertices in a binary tree having* $4p+1$ *vertices is always even. (where p is integer)*

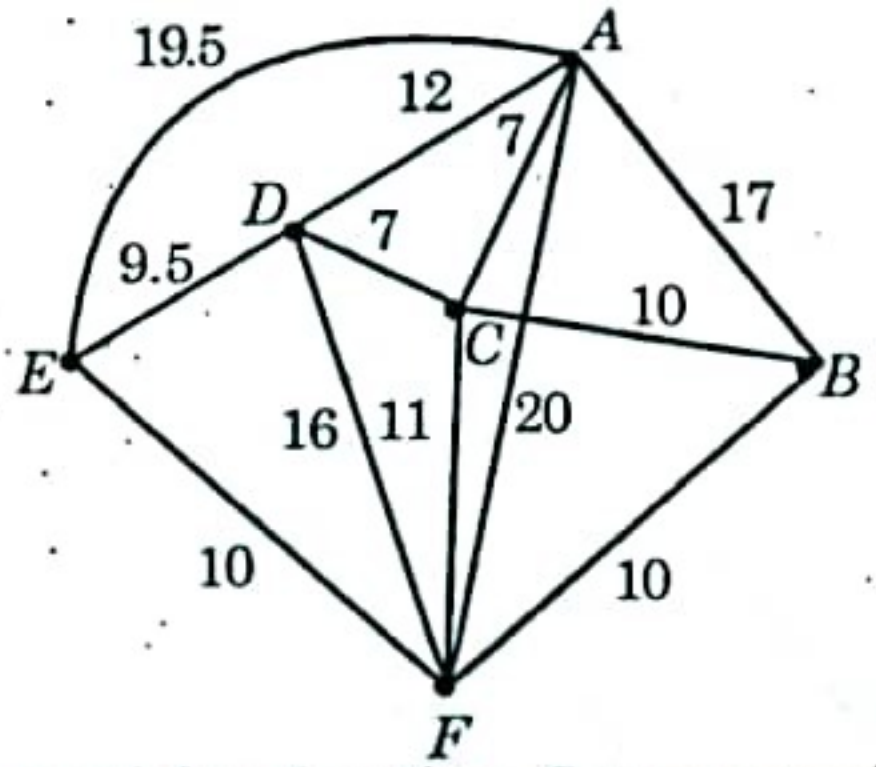
Let $n = 4p+1$. Number of pendant vertex of the binary tree is $\frac{n+1}{2}$ (from a previous Theorem).

The vertex of a tree which is not pendant is called internal vertex. So, number of internal vertices

$$= n - \frac{n+1}{2} = \frac{n-1}{2} = \frac{4p+1-1}{2} = 2p = \text{even}.$$

Hence proved.

Ex. 12. By Kruskal's Algorithm, find a minimal (or shortest) spanning tree and the corresponding weight of the spanning tree in the following graph :



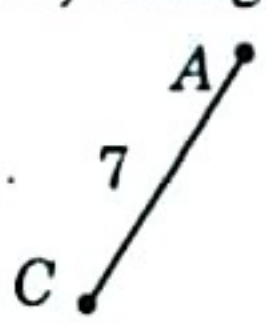
[*WBUT 2012*]

Soln. The graph has 6 vertices. So any spanning tree of it will contain $6-1=5$ edges.

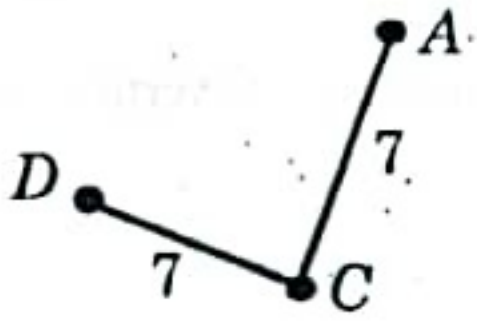
Step 1. List of edges arranged in non-decreasing weight order:

edges :	(AC)	(CD)	(DE)	(BC)	(BF)	(FE)	(CF)	(AD)	(DF)	(AB)	(AE)	(AF)
Weight:	7	7	9.5	10	10	10	11	12	16	17	19.5	20

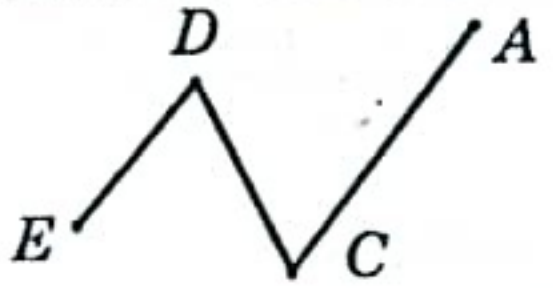
Step 2. We select the edge (AC) having smallest weight as



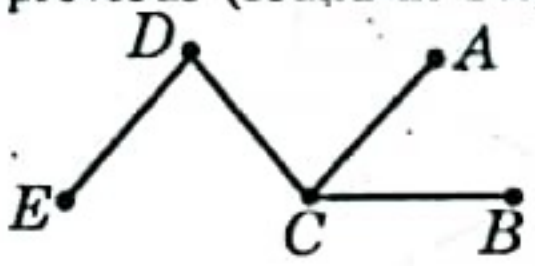
Step 3. Select the next smallest edge (CD). We accept it since it does not form any cycle with (AC). This is shown as



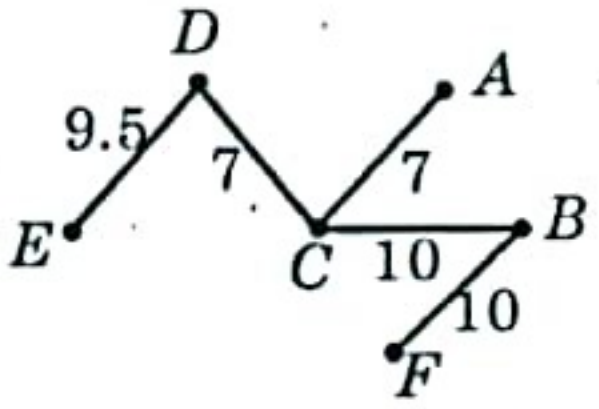
Step 4. Select the next smallest edge (DE). We accept it since it does not make any cycle with ACD. This is shown as



Step 5. Select next smallest edge *BC*. This is accepted since it does not make any cycle with the previous (found in Step 4). This is shown as



Step 6. Select the next smallest edge BF. We accept it since it does not make any cycle with the previous. This is shown as



We see this tree contains 5 edges. So this is spanning tree and by construction this is minimal spanning tree. Its weight is

$$7 + 7 + 9.5 + 10 + 10 = 43.5.$$

EXERCISE

[I] SHORT ANSWER QUESTIONS

1. Define tree. Draw a tree with four vertices.
2. Define a minimally connected graph. Draw a non-tree which is minimally connected
3. Define Binary Tree. Draw a binary tree with four edges
4. Prove that there exists a unique path joining two vertices of a tree.
5. If there exists one and only one path between any two vertices in a graph G then prove that G is a tree.
6. Prove that any tree with four vertices contains at least two pendant vertices.
7. Prove that a connected graph with 6 vertices and 5 edges is a tree.
8. If a graph is minimally connected prove that it is a tree.
9. If deletion of any edge of a graph G gives a disconnected subgraph then show that G is a tree.

10. If a tree has two centres then show that they are adjacent.

11. Prove that a tree can not have more than two centres.

12. Show that the number of vertices of a binary tree cannot be even.

13. If a binary tree has 23 vertices then prove that it has 12 pendant vertices

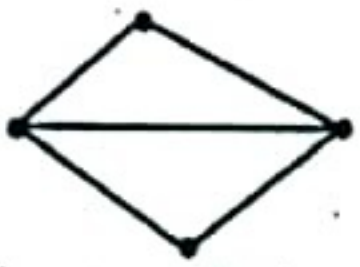
14. Show that the number of internal vertices of a binary tree with 27 vertices is 13.

15. Define spanning tree of a connected graph. Exhibit this for a graph.

16. Define chord of a tree. Draw a graph G with 5 vertices. Show a tree of G. Indicate all the chords of this tree.

17. Let G be a graph with n vertices and e edges. Find the number of chords of a spanning tree of G

18. Show that every connected graph has a spanning tree.

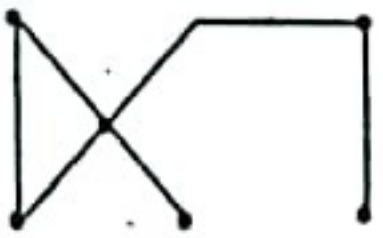
19. G be the graph  and T be tree 

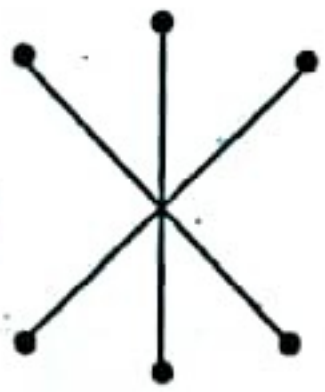
Extend this to a spanning tree. Is it possible for all trees of G.

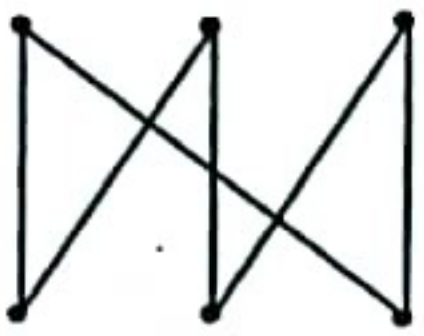
20. Which of the following graphs are tree ?

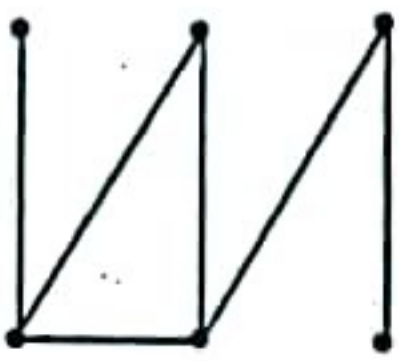
(i) 

(ii) 

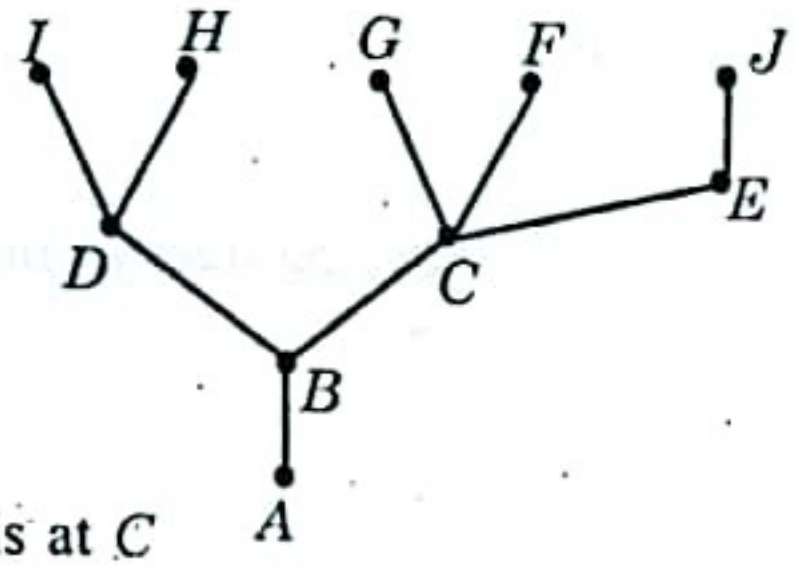
(iii) 

(iv) 

(v) 

(vi) 

21. You are given the following tree.



(i) Draw the subtree whose root is at C

(ii) Find the internal vertices

22. Draw all trees with exactly six vertices.

23. Draw all trees with five or fewer vertices.

24. Find the number of trees with seven vertices.

25. Draw three distinct trees (i.e. non-isomorphic trees) with 8 vertices.

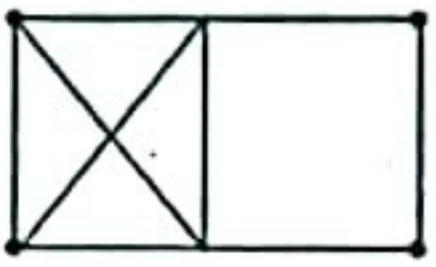
26. Let T be a tree with more than k edges. How many connected components are there in the subgraph of T obtained by deleting k edges of T ?

27. (a) Sketch all binary trees with six pendant vertices. Find the path length of each.

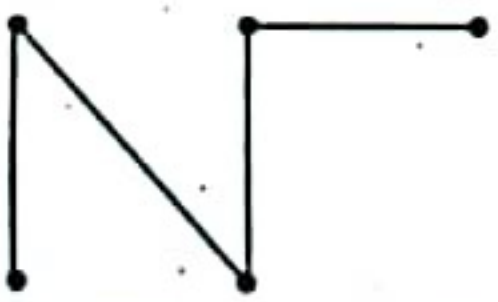
(b) How many binary trees are possible with three vertices ? Draw them.

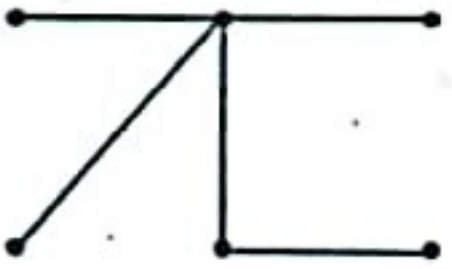
(c) Draw two different binary trees with five vertices having maximum number of pendant vertices.

(d) Differentiate between a general tree and a binary tree.

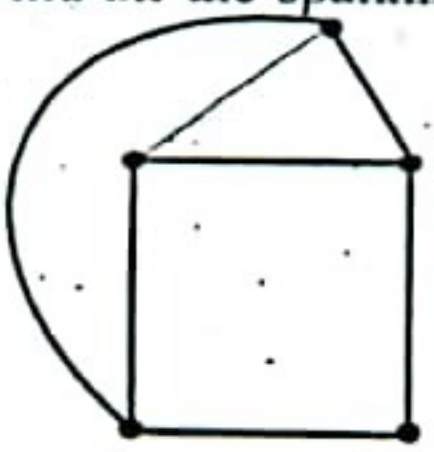
28. Let G :  be a connected graph.

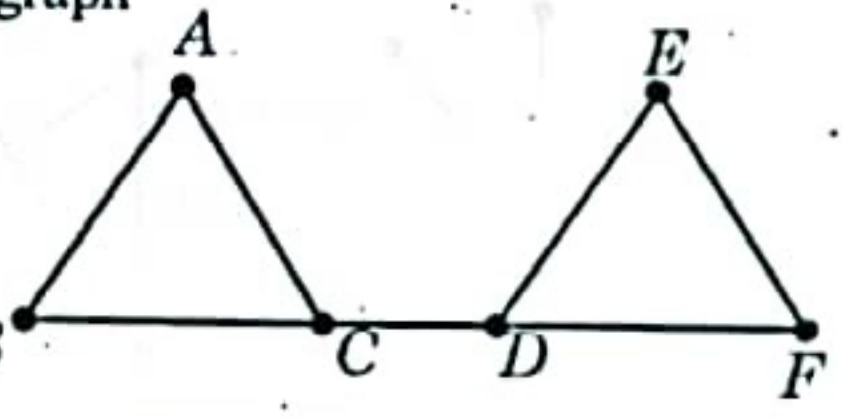
Which one of the followings is a spanning tree of G

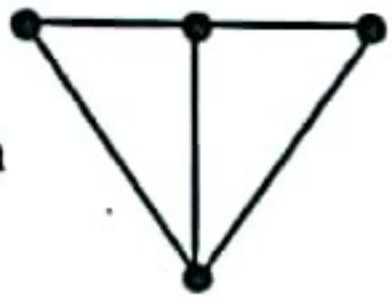
(i) 

(ii) 

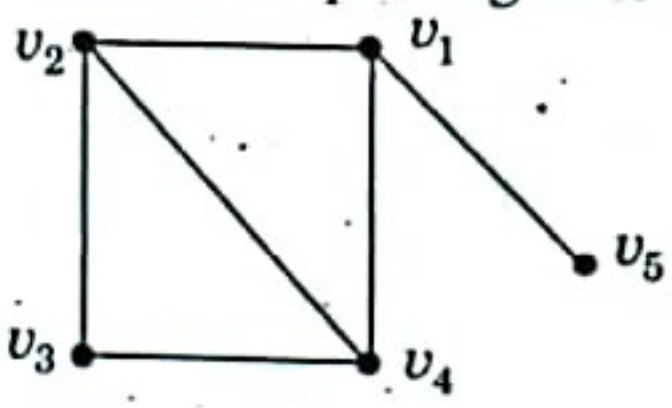
29. Find all the spanning trees of the graph

(i) 

(ii) 

30. Find all the spanning trees of the graph 

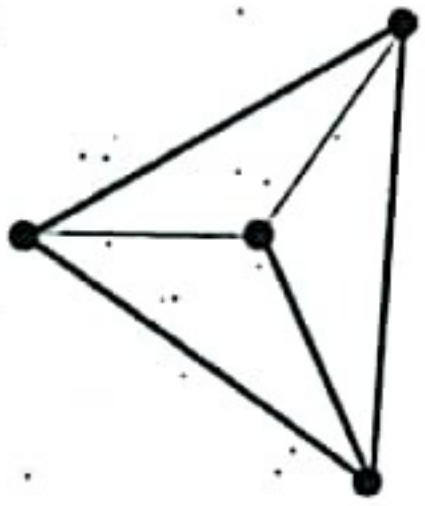
31. Sketch all spanning trees of the following two graphs :



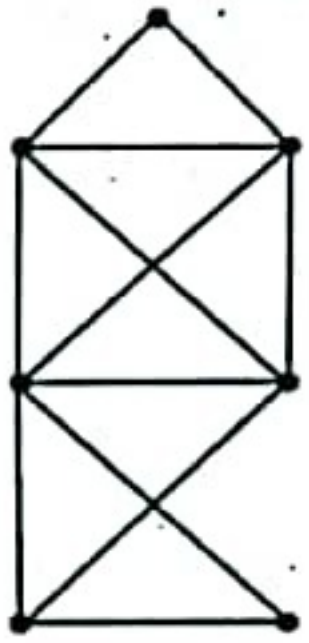
and 

32. Find a spanning tree for each of the following graphs by removing edges.

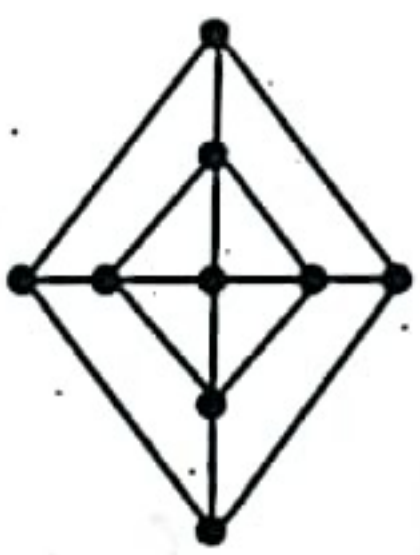
(i)



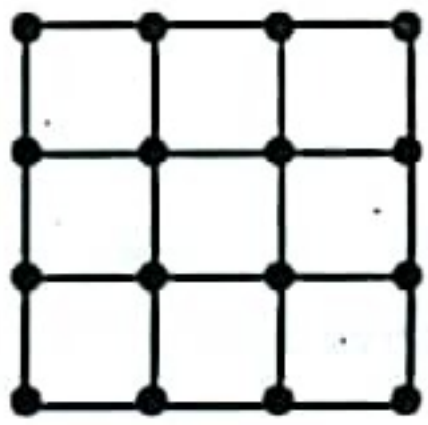
(ii)



(iii)

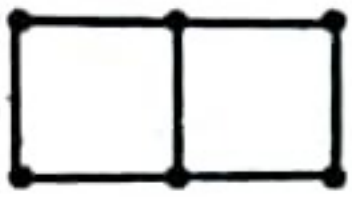


(iv)



(v)

33. Find the number of spanning trees of the graph G :



34. A graph with exactly one spanning tree is a tree.

[Hint : If C be a circuit contained in G then deletion of a branch of the sp. tree gives a sp. tree]

35. Draw a tree (i) with nine edges and nine vertices

(ii) with six vertices and sum of degrees of all vertices 14.

(iii) with six vertices having degree 1, 1, 1, 1, 3, 3

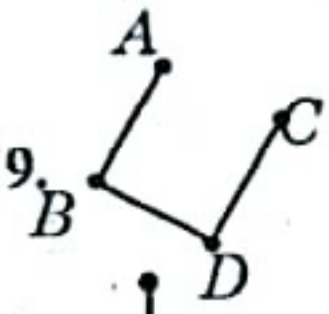
(iv) with all vertices of degree 2

36. Draw two rooted trees having four vertices

ANSWERS

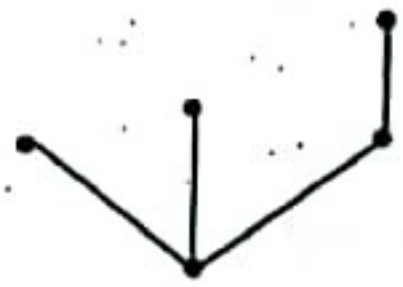
17. $e-n+1$

19.



20. (ii) (iii) (iv)

21. (i)



(ii) *A*, *B*, *C*, *D*, *E*

22.

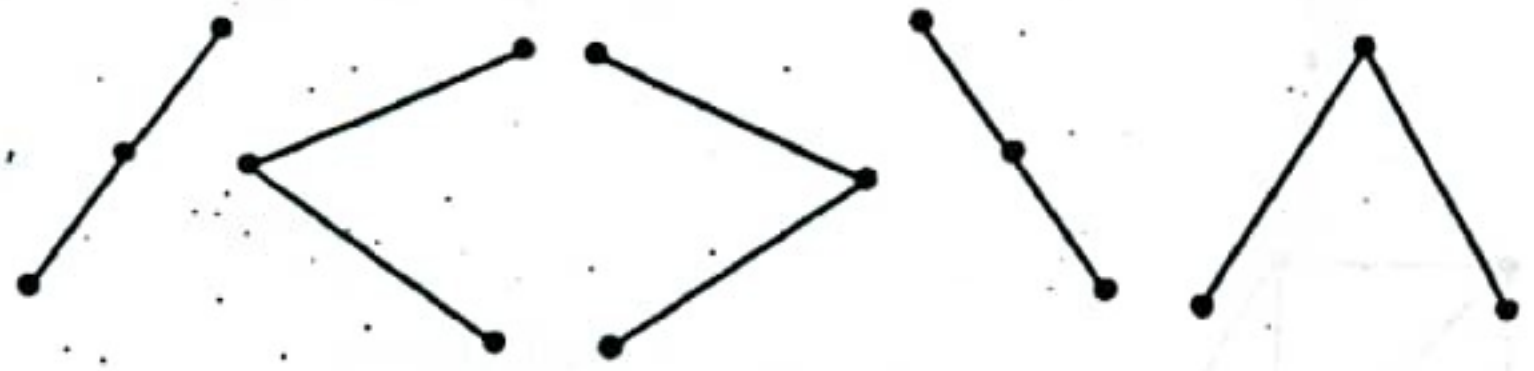
23. There are eight such trees including the null graph.

24. 10

28. No. of edges $-k$

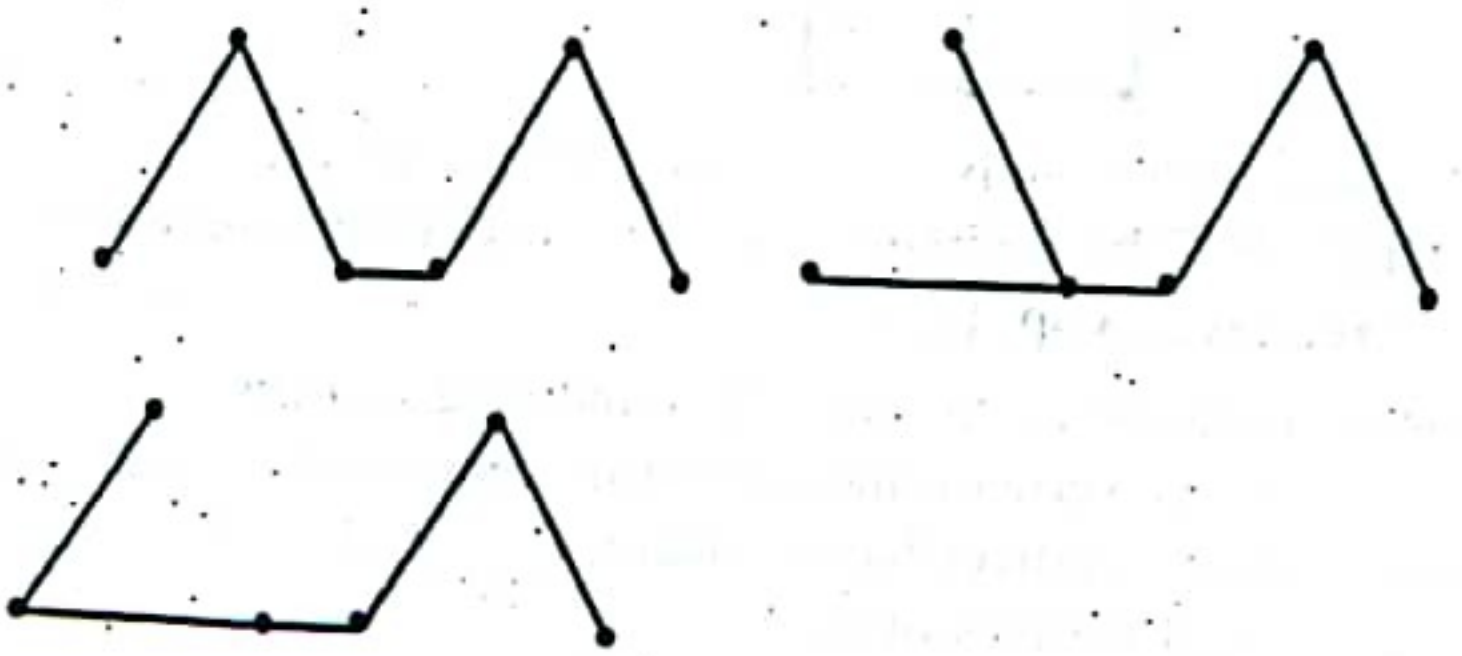
27. (a) Six such binary trees

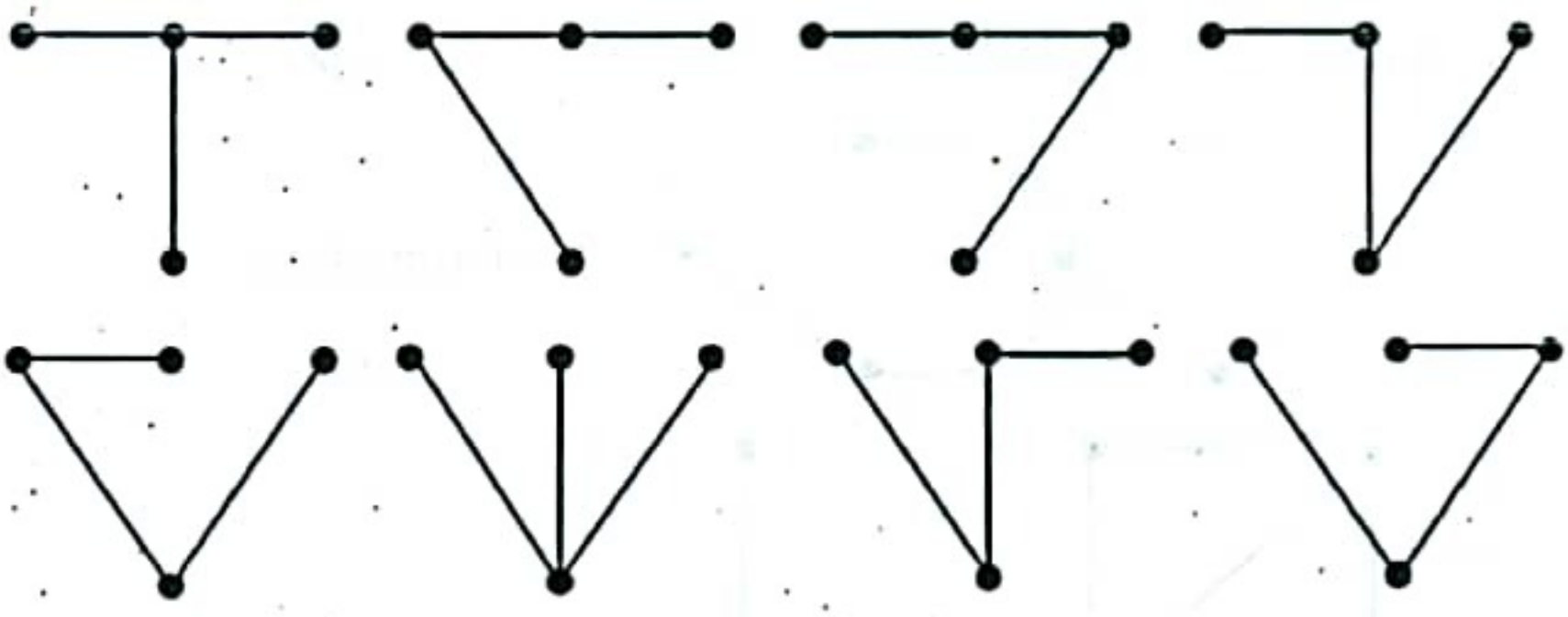
(b) Five. They are

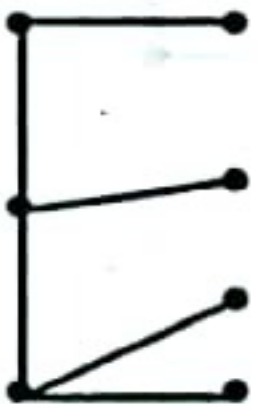
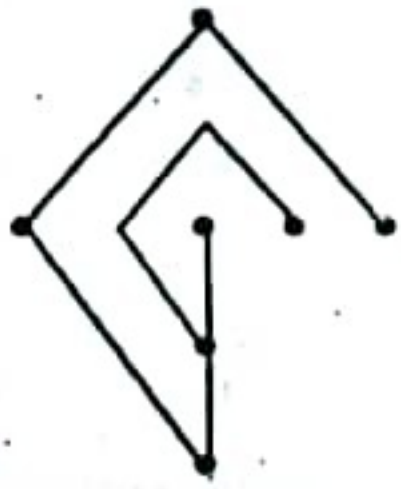


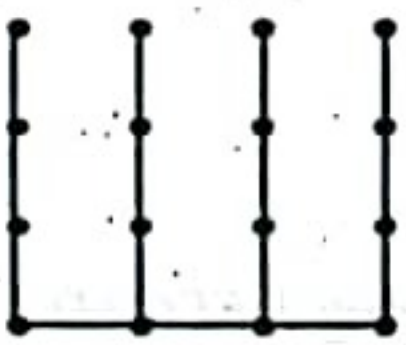
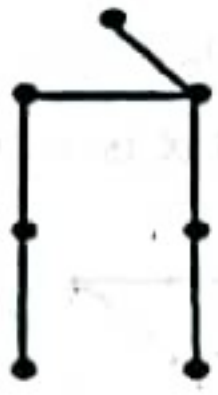
29. (i)

(ii)



30. 

32. (i)  (ii)  (iii) 

(iv)  (v) 

33. 15

35. (i) No such tree since a tree of 9 vertices would have $9-1=8$ edges.

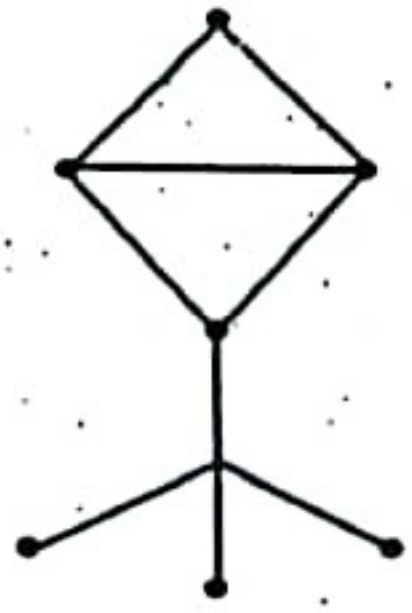
(ii) No such tree since a tree with 6 vertices has 5 edges and hence total degree of 10, not 14.

(iii) 

(iv) No such tree exists because such a graph must contain a circuit (by a previous Th)]

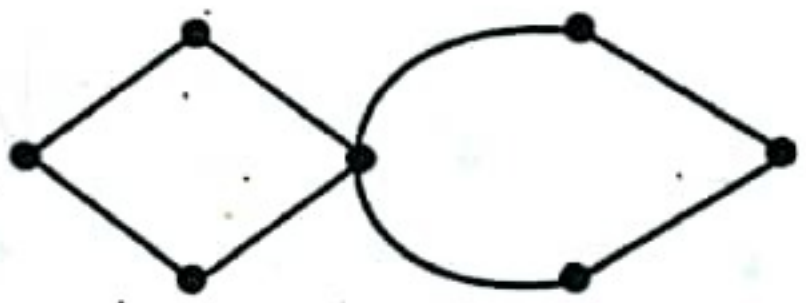
[II] LONG ANSWER QUESTIONS

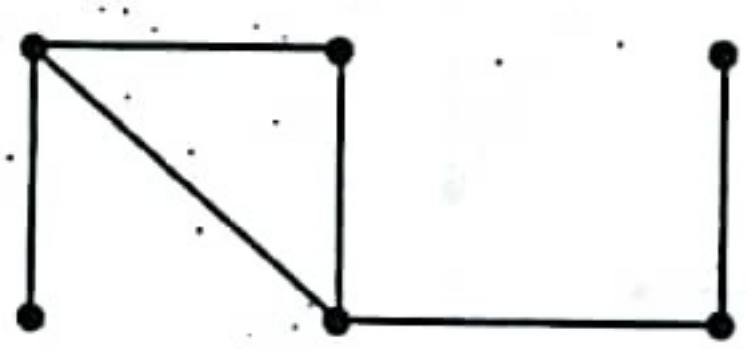
1. Give a list of spanning trees of the following graphs

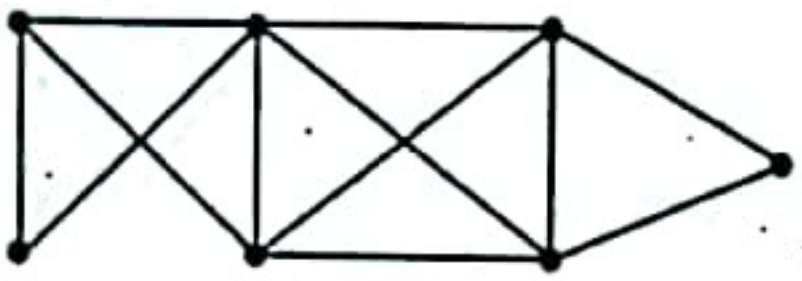
(i) 

(ii) 

2. Use *BFS* algorithm to find a spanning tree of the following graphs:

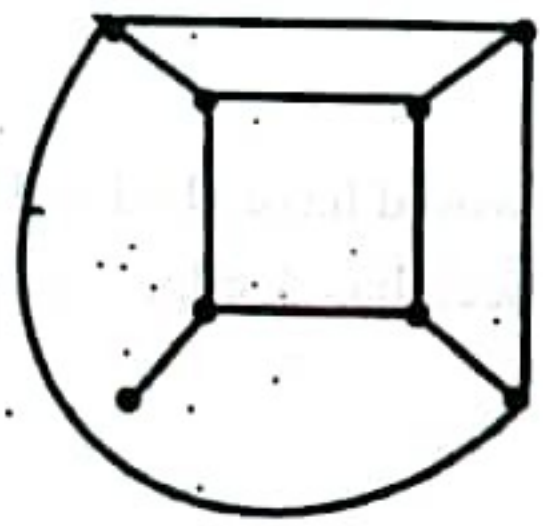
(i)  starting from G .

(ii) 

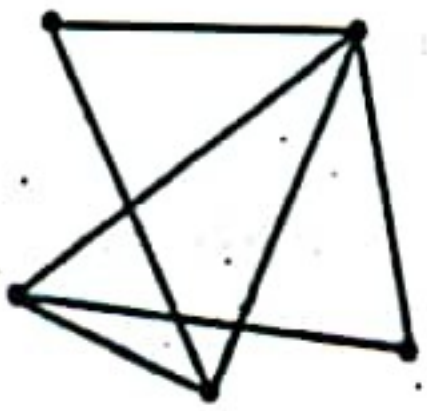
(iii 

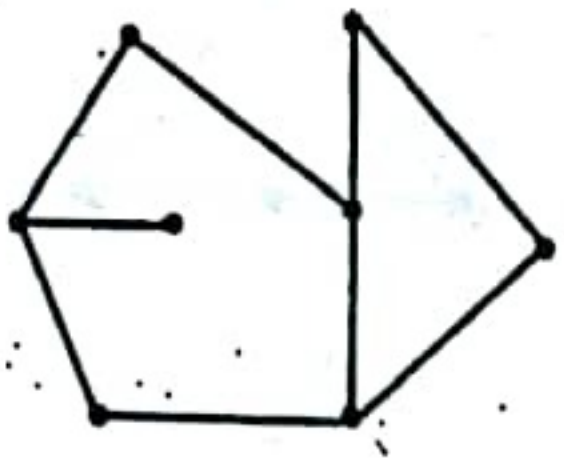
using vertex ordering $v_1v_2v_4v_5[v_3v_6v_7]$

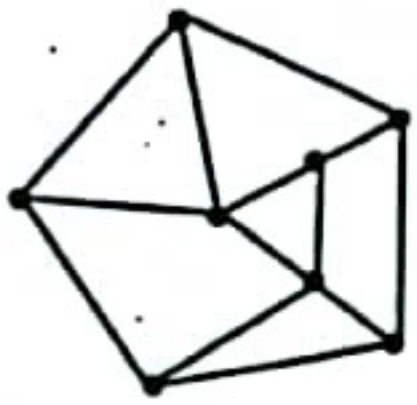
[Hint : Starting vertex is v_1 v_3 .]

(iv)  using vertex ordering *HFEGADCDB*

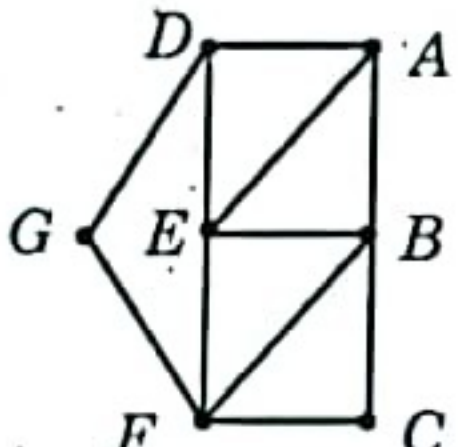
Hint : Starting vertex is H]

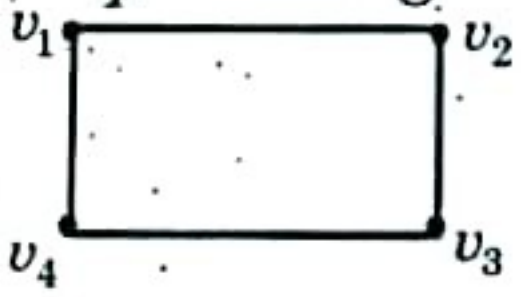
(v)  starting from *d*.

(vi)  using vertex ordering

(vii)  with vertex ordering *CABEFHGD*

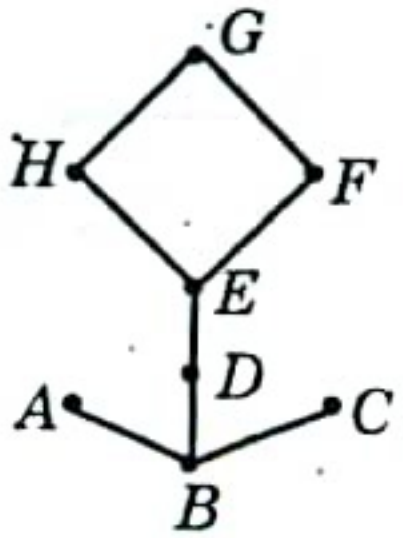
3. Use *DFS* algorithm to find a spanning tree of the following graph.

(i) 

(ii) 

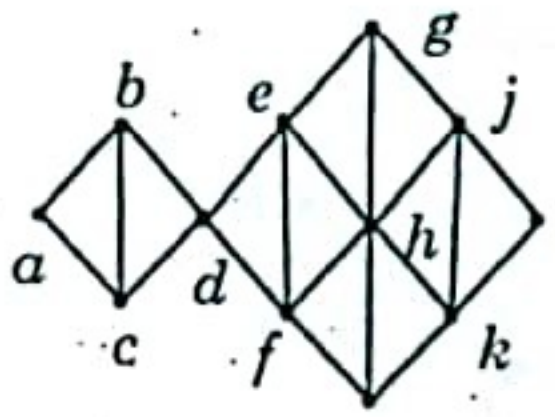
with vertex ordering $v_2v_1v_4v_3$

[Hint : Start with $v_2, v_1 \cdots$]

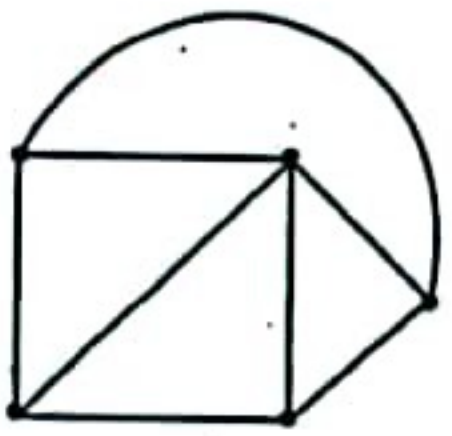
(iii) 

with vertex ordering *ABCDEHGF*

[Hint : Start with the vertex *A* and consider the path *ABC* then *B*, *D*, *E*, ... etc.]

(iv) 

starting from the vertex *z*.

(v) 
with vertex ordering *C*, *D*, *A*, *B*, *E*.

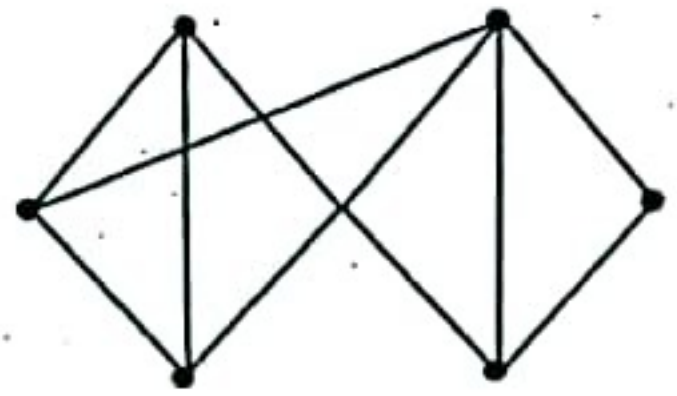
[Hint : Start with *C*, *D*, etc.]

4. Let C be a cycle in a connected graph G which is not a tree. Prove that the complement of arbitrary spanning tree of G contains at least one edge of C.

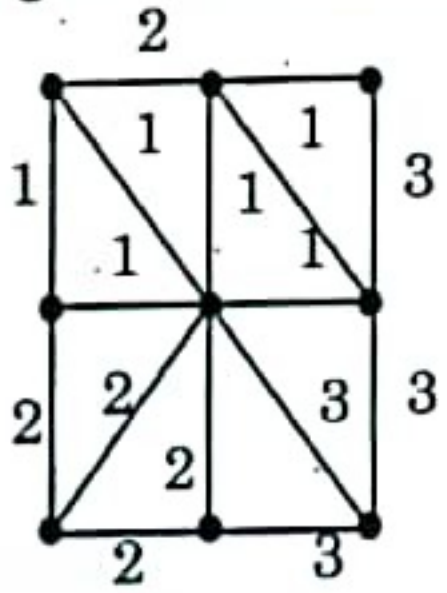
[Hint : Any spanning tree has a chord in C. Since chord constitute the complement of a spanning tree, the result follows].

5. Find by Kruskal's Algorithm, a minimal spanning tree of the weighted graph G shown as

G:

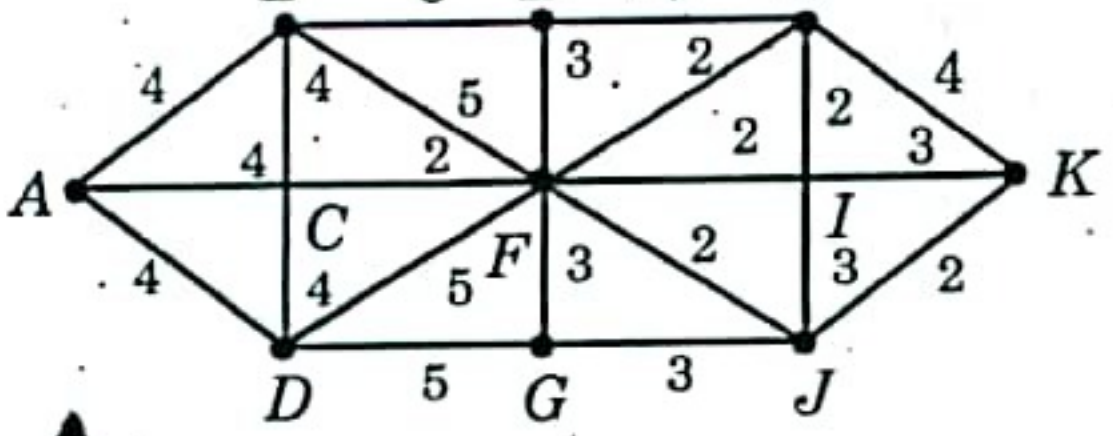


6. Find by Kruskal's Algorithm, a minimal spanning tree of the weighted graph :

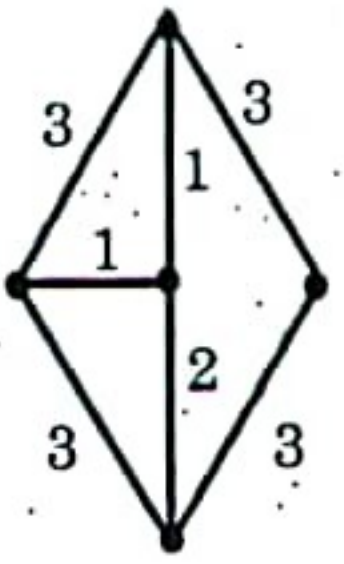


7. By Kruskal's Algorithm find a minimal (or shortest) spanning tree and the corresponding weight of the sp tree in the following graphs :

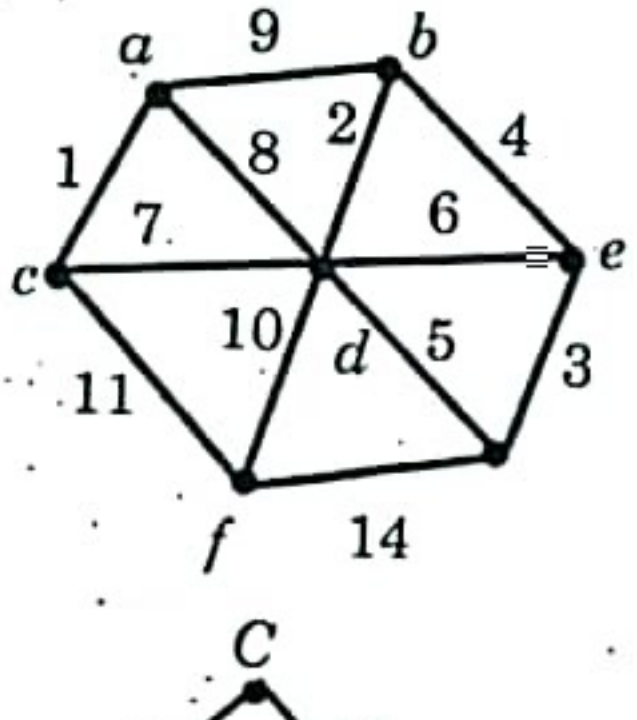
(i)



(ii)

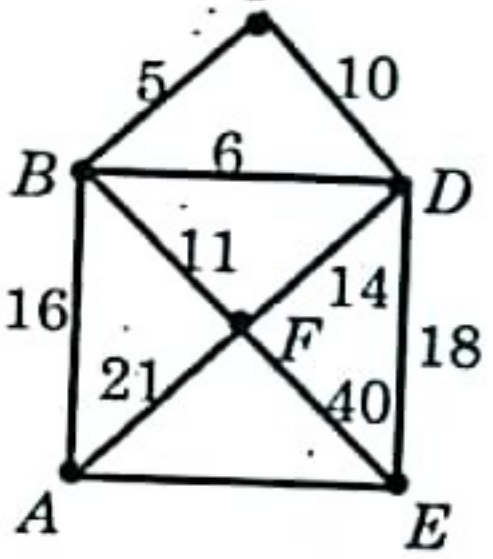


(iii)

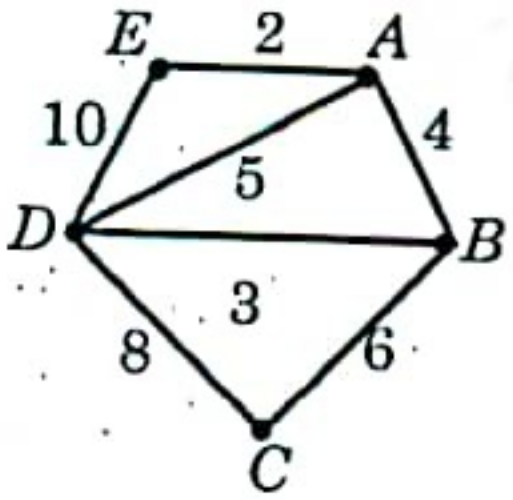


C

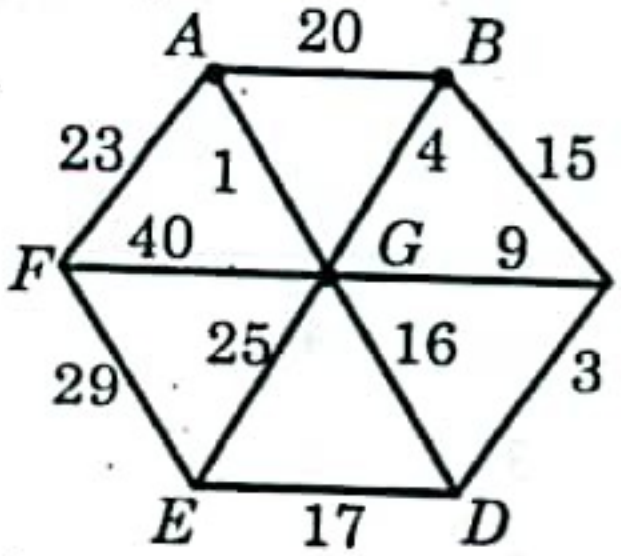
(iv)



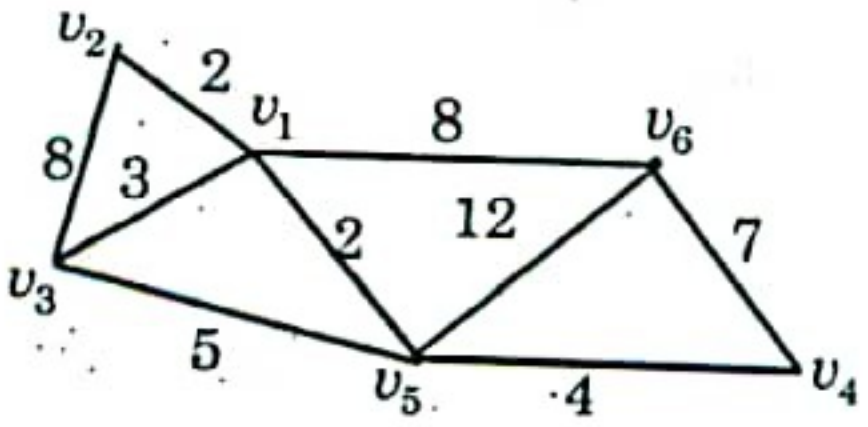
(v)



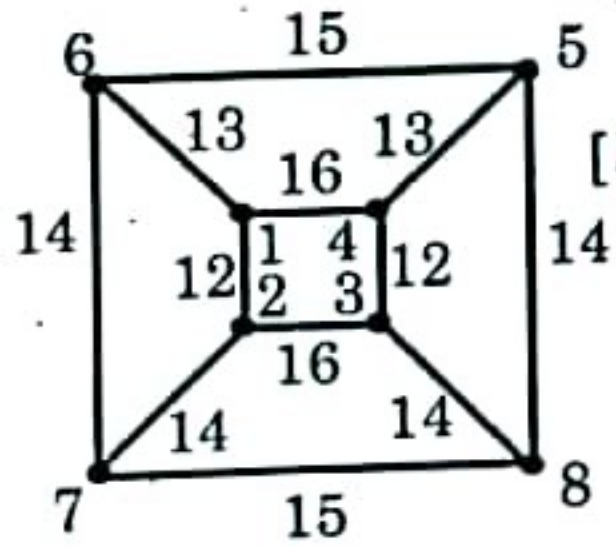
(vi)



(vii)

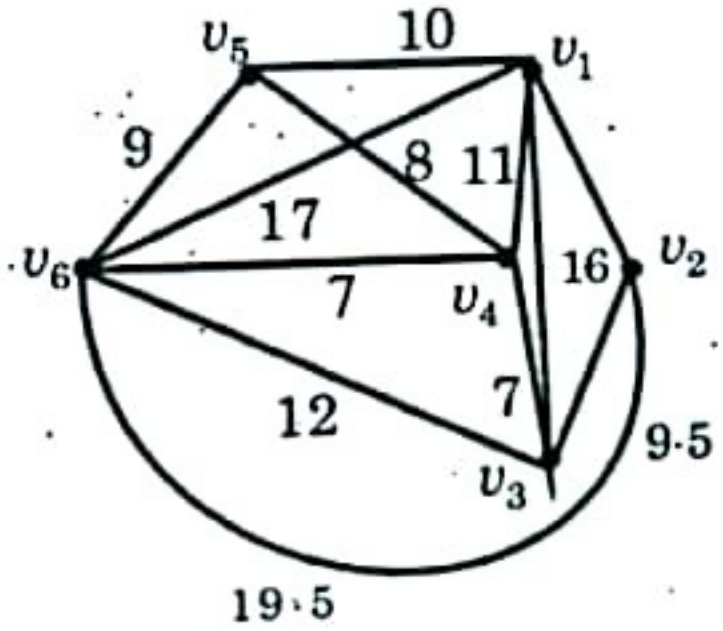


(viii)

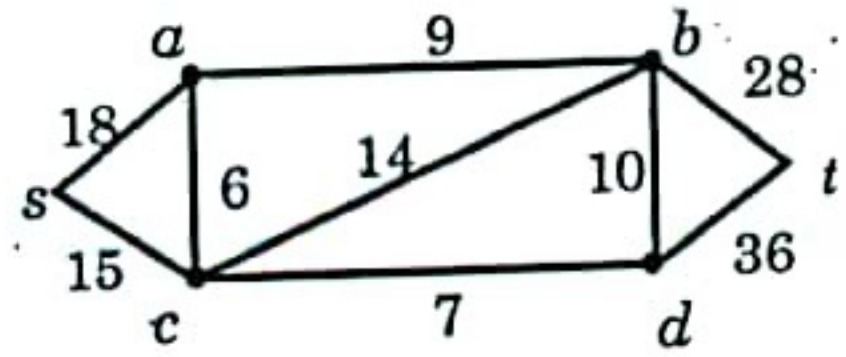


[*W.B.U. Tech, 2002*]

(ix)



(x)



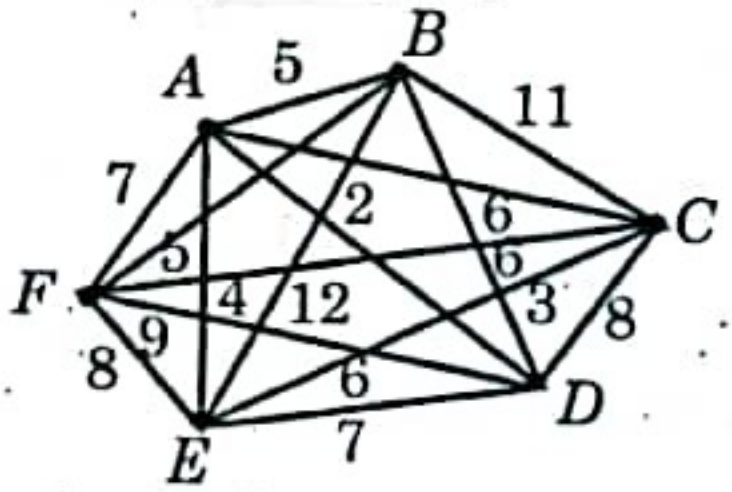
[*W.B.U Tech, 2003*]

8. The following table shows the distances, in kilometers, between six villages in India. Find a minimal spanning tree connecting the six villages applying Kruskal's Algorithm :

	A	*B*	*C*	*D*	*E*	*F*
A	-	5	6	12	4	7
B	5	-	11	3	2	5
C	6	11	-	8	6	6
D	12	3	8	-	7	9
E	4	2	6	7	-	8
F	7	5	6	9	8	-

[*W.B.U.Tech 2004*]

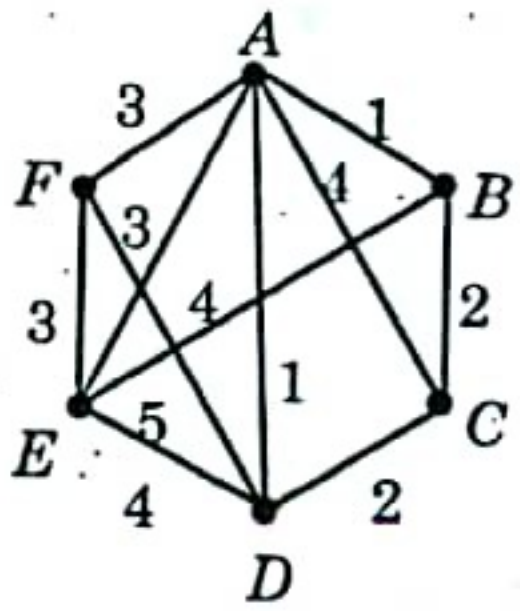
[Hint : The corresponding graph is



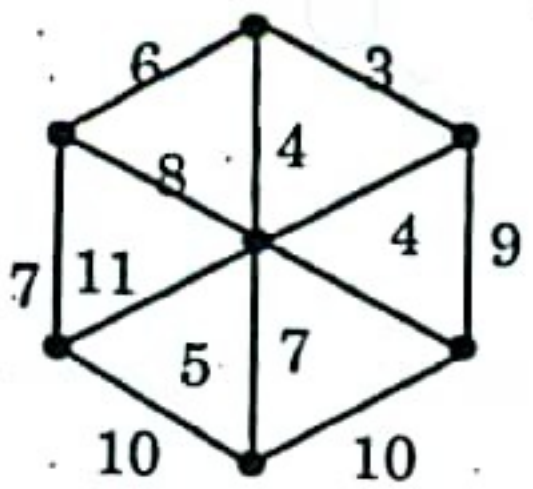
Then proceed according to Kruskal's method]

9. By Prim's Algorithm find a minimal (or shortest) spanning tree in the following graphs and find the corresponding minimum weight :

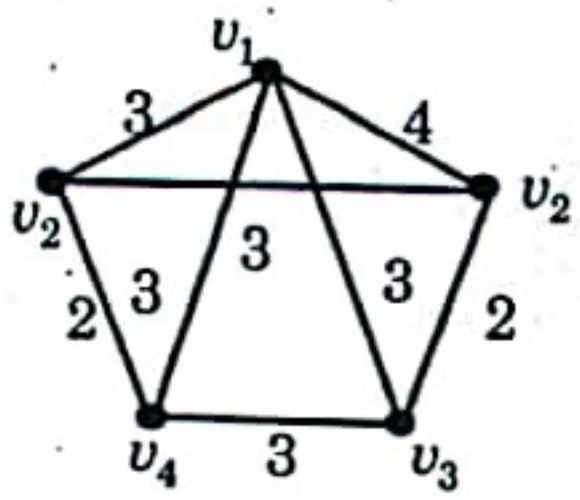
(i)



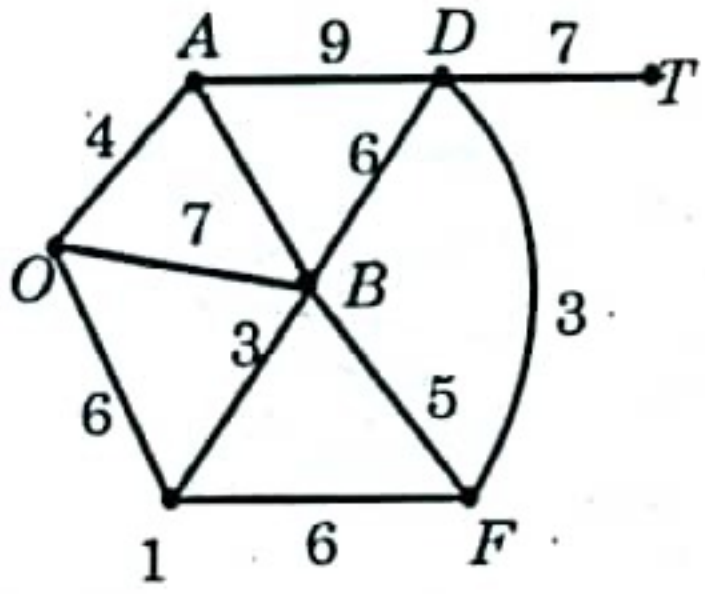
(ii)



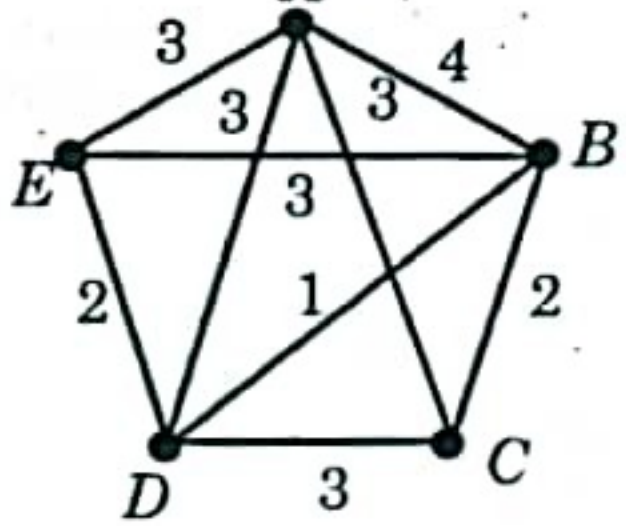
(iii)



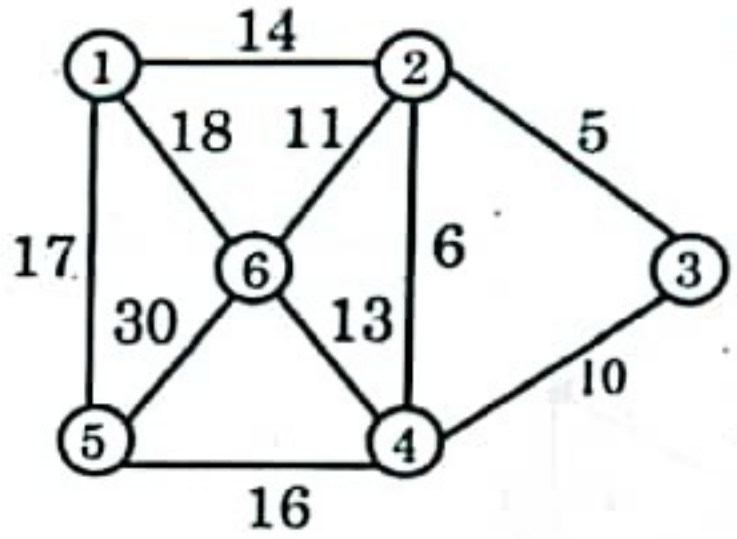
(iv)



(v)



(vi)



[*W.B.U.Tech 2005,2017*]

10. Let G be a graph having no cycles, n vertices and k connected components. Then prove that G has $n-k$ edges.

[Hints : $n = n_1 + n_2 + \cdots + n_k$; $n_i =$ No. of vertices in i-th component. Applying Theorem 7 of Art 5.3.3 the No. of edges in the i-th component is $n_i - 1$

$$\therefore \text{total No. of edges} = \sum_{i=1}^{k}(n_i - 1) = n - k \,]$$

11. Let T be a tree and let v be a vertex of maximum degree in T. If $\deg(v) = k$ prove that T has at least k vertices of degree 1.

12. Prove that a graph G is a tree iff G is connected but $G - e$ is disconnected for any edge e of G.

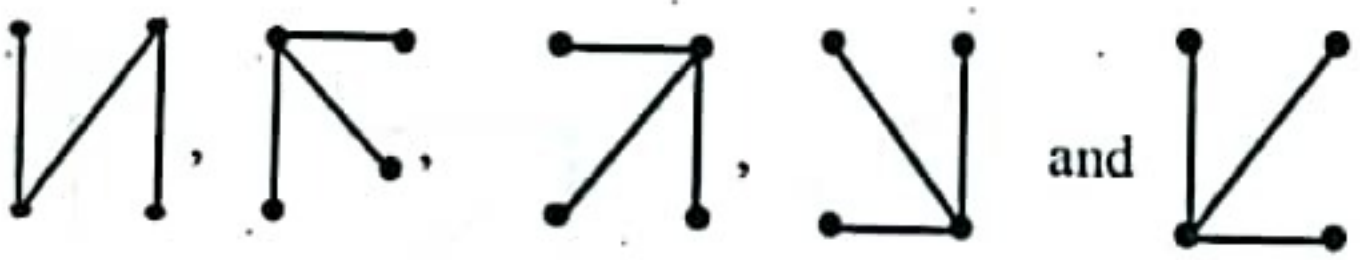
[Hint : Nothing but the Theorem 8 of Art 5.3.3]

13. Let u and v be two non-adjacent vertices of a tree. Prove that if u and v are joined by an edge then the newly formed graph contains a cycle.

14. Show that a path is its own spanning tree.

15. How will you construct a graph if you are given all its spanning trees.
[Hint : Follow an Illustrative Example]

16. Construct the graph whose all the spanning trees are



[Hint : See Illus. Ex. 5 of Art 4.2.8]

17. Construct the graph whose all the spanning trees are

18. Show that every edge of a connected graph is a branch of some spanning tree of G.

[Hint : See inside of proof of a previous Th]

Is it true that any arbitrary edge of a connected graph is a chord for some spanning tree of the graph?

19. State some necessary conditions of the n positive integers $x_1, x_2, \cdots, x_n$ so that they can be degrees of all the n vertices of a tree ?

20. Which connected simple graphs have exactly one spanning tree.

21. Let G be a connected weighted graph in which every edge belongs to some cycle, e be the edge whose weight is greater than that of any other edge in G. Prove that there exists no shortest spanning tree in G that contains e.

[Ex. 3 of Art 4.2.8 is the Analogous problem]

22. Prove that if G is a connected weighted graph in which no two edges have the same weight then G has a unique minimim spanning tree.

[Hint : For two distinct minimal sp. tree S_1, S_2,

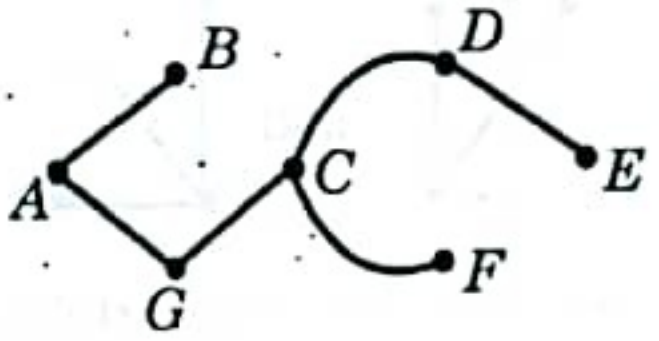
$$\sum_{n-1\ No.} w_i = \sum_{n-1\ No.} w_j \Rightarrow w - w_k = w - w_p \Rightarrow w_k = w_p$$

a contradiction where $w =$ weight of G]

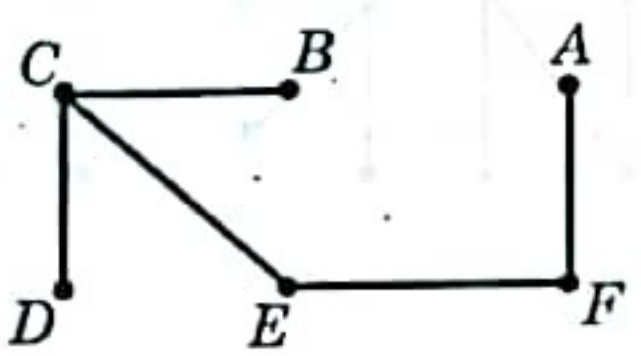
23. Give an algorithm to find a minimal spanning tree of a weighted connected undirected graph.

ANSWERS

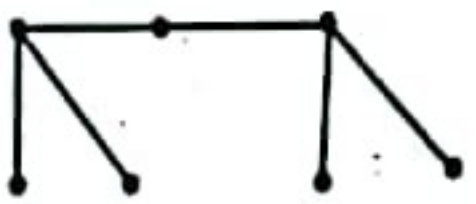
2. (i)



(ii)

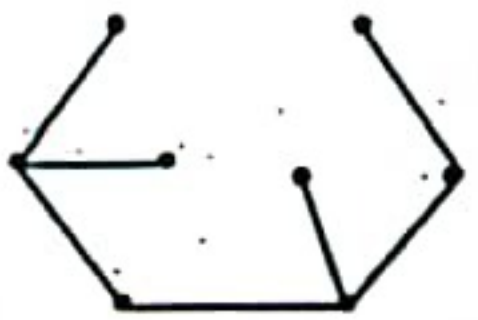


(iii)

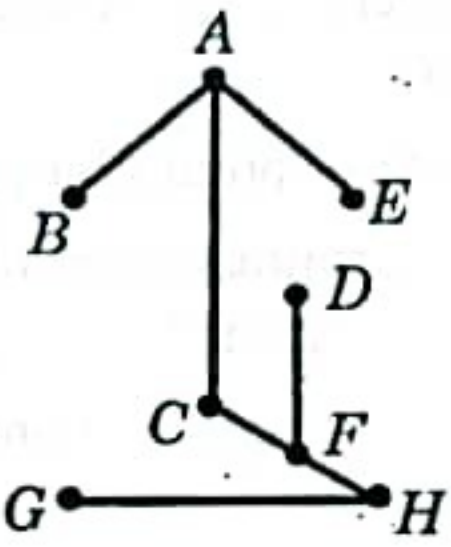


(iv)

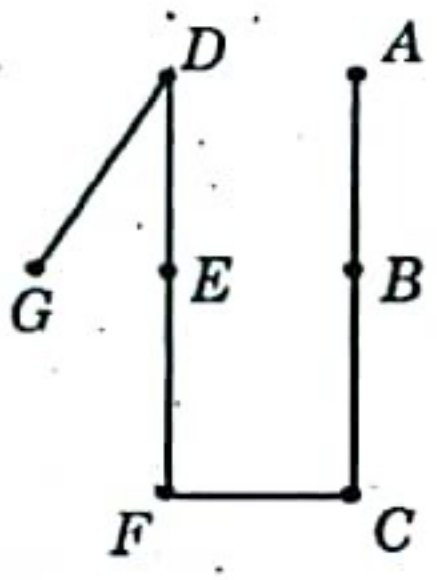
(vi)



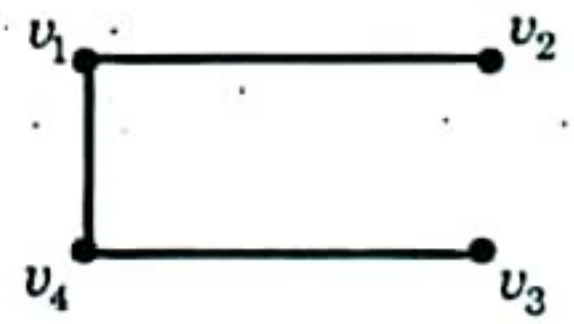
(vii)



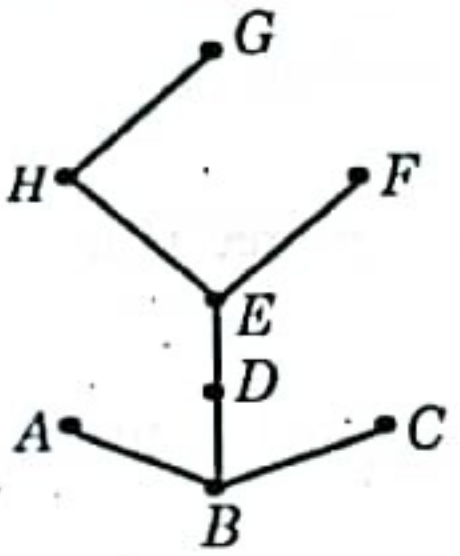
3. (i)



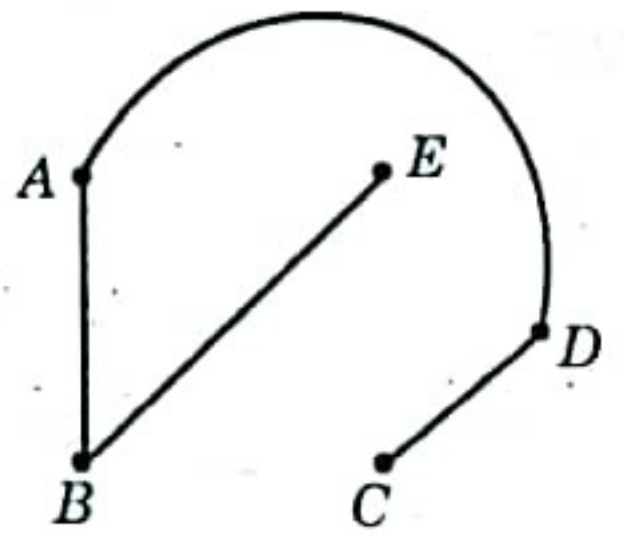
(ii)



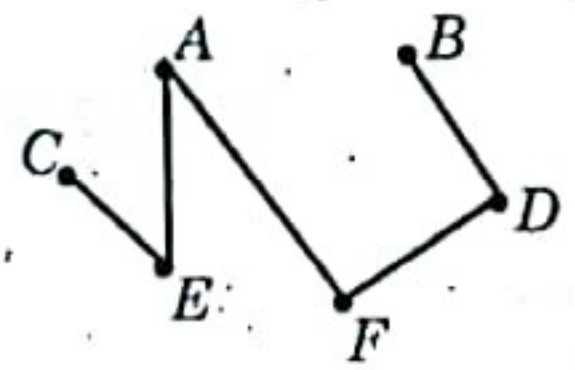
(iii)



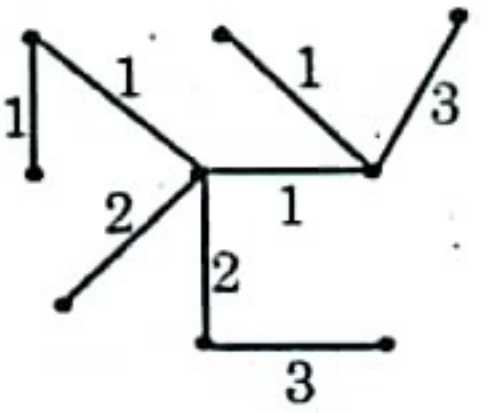
(v)



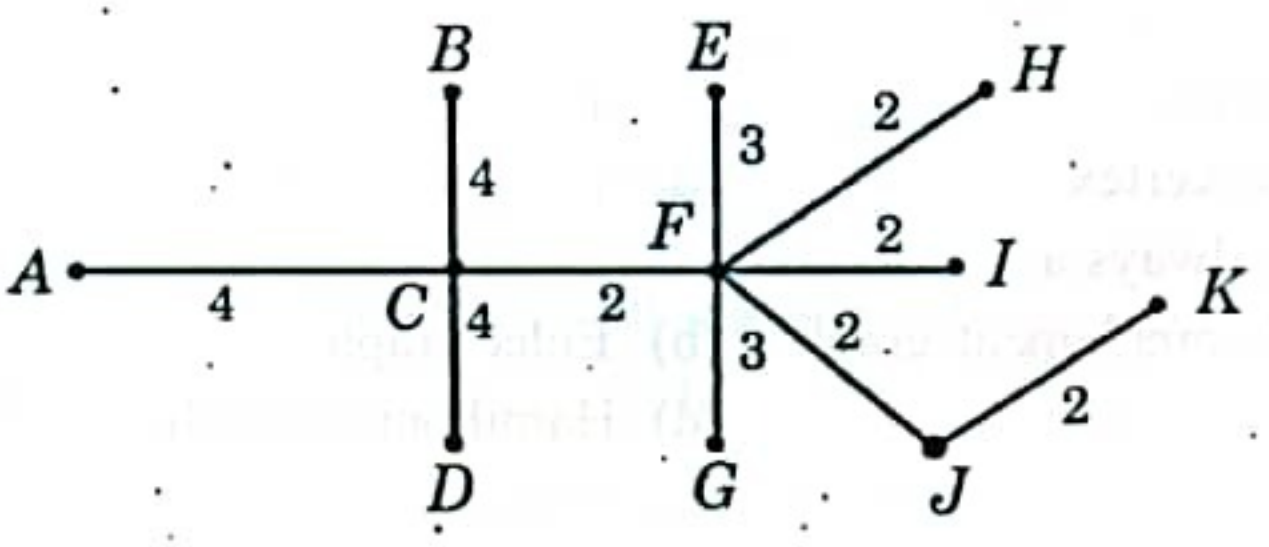
5.



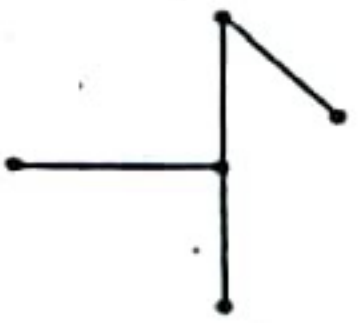
6.



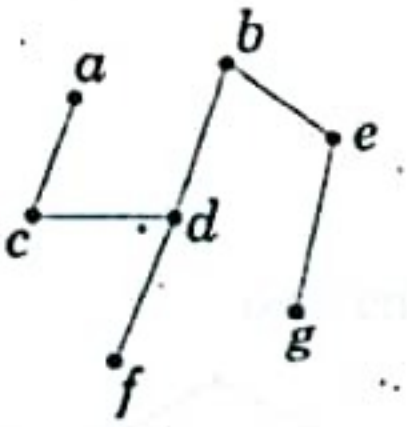
7. (i)



(ii)



(iii)



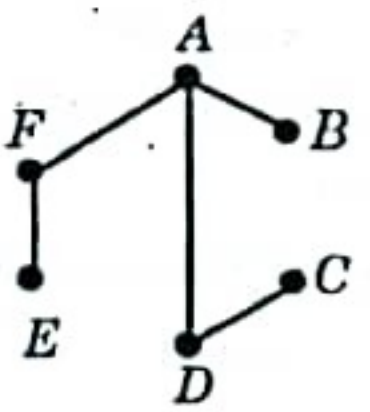
(iv) AB, AE, BC, BD form the shortes sp. tree

(v) AB, AE, BC, BD form the shortes sp. tree

(vi) AF, AG, GB, GC, CD, DE form the shortest sp. tree

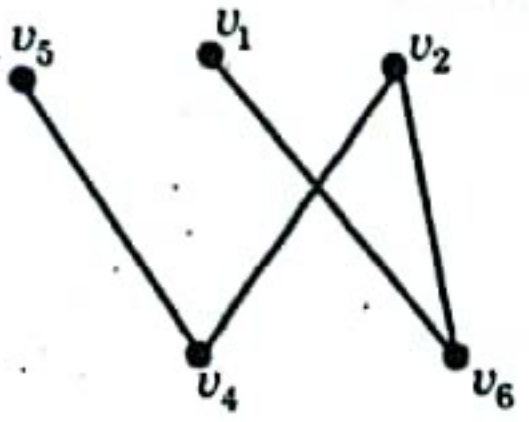
(x) d, c, s, a, b, t ; $wt = 55$

8. Wt. of minimal sp. tree = 20



9. (i) wt of the shortest spanning tree is 10 ;

(iii)



(iv) *OA, AB, BC, BF, FD, DT* form the sp. tree

(v) AC, CB, BD, DE form the short sp. tree

(vi) minimum weight $= 52$; the minimum sp. tree is not shown here.

18. No. e.g consider a path as a graph.

19. At least two $x_i = 1; \sum x_i = 2(n-1)$ by Theorem **20.** Tree.

[III] Multiple Choice Questions

1. Tree is a connected grap without any *[WBUT 2012]*

(a) pendant vertex (b) circuit

(c) odd vertex (d) even vertex

[Hint : See definition of Tree. Remember every cycle is a circuit.]

2. Tree contains at least

(a) one vertex (b) two vertex

(b) three vertex (d) none of these

3. A tree is always a

(a) self complement graph (b) Euler graph

(c) simple graph (d) Hamiltonian graph

4. In a graph a pair of parallel edges form a

(a) path (b) loop (c) Euler line (d) circuit

5. Eccentricity of a vertex becomes larger as vertex lying

(a) extreme of a graph (b) central position of a graph

(c) none of the two

6. In the graph  eccentricity of B is

(a) 1 (b) 0 (c) 2 (d) 3

7. A minimally connected graph becomes disconnected when one of the following is removed

(a) one vertex (b) one edge (c) two edge (d) none

8. A tree is a

(a) any connected graph (b) minimally connected graph

(c) Euler graph (d) none

9. A minimally connected graph can not have a

(a) cycle (b) component

(c) even vertex (d) pendant vertex

10. A binary tree has exactly

(a) two vertices of degree 2 (b) one vertex of degree 2

(c) one vertex of degree 1 (d) one vertex of degree 3.

11. Each vertex (except one) of a binary tree has degree

(a) 1 or 2 (b) 2 or 3 (c) 1 or 3 (d) 2 or 4

12. Sum of the degrees of all vertices of a binary tree is even if the tree has

(a) odd number of vertices (b) even number of vertices

(c) four vertices (d) none of these

13. The root of a binary tree is the vertex having degree

(a) 1 (b) 2 (c) 3 (d) 4

14. A binary tree should have at least

(a) one vertex (b) two vertices

(c) three vertices (d) four vertices

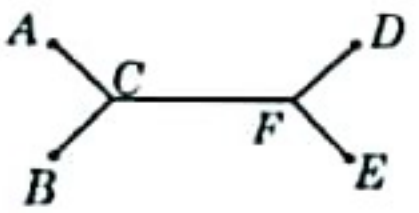
15. A binary tree has exactly

(a) one root (b) two root (c) three root (d) none

16. A pendant vertex of a tree cannot be

(a) centre of the tree (b) internal vertex of the tree

(c) none of the two

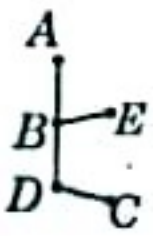
17. In the graph 

the centre is the vertex

(a) C (b) A (c) B (d) E

18. The eccentricity of the vertex of a graph having only one vertex is

(a) 0 (b) 1 (c) 2 (d) 3

19. Which one of the vertices of the tree 

is the root

(a) C (b) E (b) B (d) D

20. If A, B is a pair of vertices in a graph then which one of the following is true

(a) A and B are pendant vertices

(b) the distance of A and B is least

(c) there exists a path from A to B

(d) there may exist two paths from A to B

21. Addition of an edge between any two vertices of a tree creates

(a) Eular line (b) circuit

(c) longest path (d) regular graph

22. Addition of an edge between any two vertices of a tree

(a) may create no circuit (b) creates only one circuit

(c) may create two circuits (d) none.

23. The degree of the origin of the longest path in a tree is

(a) 1 (b) 2 (c) 3 (d) none

24. The minimum number of pendant vertices in a tree with five vertices is

(a) 1 (b) 2 (c) 3 (d) 4

25. The number of edges in a tree with 50 vertices is

[W.B.U.Tech 2007]

(a) 48 (b) 49 (c) 50 (d) 51

26. A graph with 10 vertices and 9 edges

(a) is a tree (b) is Euler graph

(c) may be a disconnected graph (d) none

27. A connected graph with 150 vertices and 149 edges is

(a) not a minimally connected graph (b) Euler graph

(c) binary tree (d) tree

28. A minimally connected graph is a

(a) binary tree (b) Hamiltonian graph

(c) tree (d) regular graph

29. If an edge of a tree is deleted then it becomes

(a) disconnected (b) binary tree

(c) spanning subgraph (d) none of these

30. The minimum number of line segments to inter connect 200 distinct points is

(a) 198 (b) 199 (c) 200 (d) 201

31. If 50 points are interconnected by 49 line segments then the resulting structure is a

(a) a tree (b) a graph with at least one circuit

(c) not a graph at all (d) complete graph

32. The number of centres of a tree is

(a) 0 or 1 (b) 1 or 2 (c) 2 or 3 (d) 3 or 4

33. The number of vertices of a tree with 36 edges is

(a) 35 (b) 36 (c) 37 (d) 38

34. The possible number of vertices in a binary tree is

(a) 4 (b) 6 (c) 2 (d) 5

35. The possible number of vertices in a binary tree is

(a) 1 (b) 2 (c) 3 (4) 4

36. The number of pendant vertices of a binary tree with 49 vertices is

(a) 25 (b) 26 (c) 27 (d) none of these

37. The number of internal vertices of a binary tree with 47 vertices is

(a) 21 (b) 22 (c) 23 (d) 24

38. If a binary tree has 15 pendant vertices then the number of vertices of the tree is

(a) 27 (b) 28 (c) 29 (d) 30

39. If a binary tree has 18 internal vertices then the number of vertices of the tree is

(a) 19 (b) 37 (c) 35 (d) none of these

40. If a binary tree has 20 pendant vertices then the number of internal vertices of the tree is

(a) 20 (b) 21 (c) 23 (d) 19

41. If a graph G has n number of vertices then the number of vertices of a spanning tree of G is

(a) n (b) $n-1$ (c) $n+1$ (d) none of these

42. Which of the following statement is true for a graph G

(a) a spanning tree is a super Graph of G

(b) a spanning tree is a subgraph of G

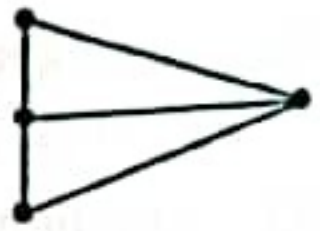
(c) a spanning tree may not be a tree at all

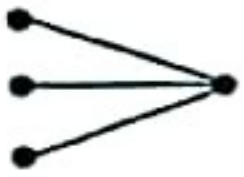
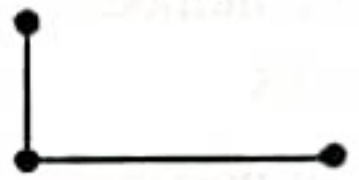
(d) G may not have a spaning tree

43. Among all trees obtained from a graph G spanning tree is

(a) smallest (b) largest

(c) the one containing the even vertices (d) none of these

44. Which one of the following is a spanning tree of the graph 

(a)  (b)  (c)  (d) 

45. If T be a tree obtained from a graph G then a chord of T is

(a) an edge of T (b) an edge of G

(c) an edge of the complement of G (d) an edge of G but not of T

46. Let G be a graph with 10 vertices and 12 edges. Then the number of chords of a spanning tree of G is

(a) 3 (b) 2 (c) 4 (d) none

47. If T be a spanning tree in a graph G then the co-tree of T contains

(a) all the chords of T (b) all the vertices of G which are not of T

(c) all the edges of G (d) none

48. If T and $\overline{T}$ are tree and Co-tree in a graph G then $T \cup \overline{T} =$

(a). $\overline{T}$ (b) G - the edges of T

(c) G (d) none

49. A graph G has a spanning tree iff G is

(a) regular (b) connected

(c) simple (d) tree

50. If the chords of a spanning tree T of a graph G are removed from G then we obtain .

(a) T (b) $G-T$

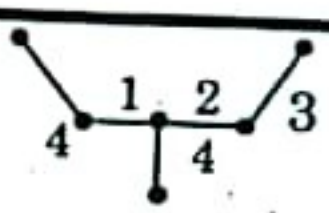
(c) the complement of G (d) none

51. To make a graph (with e edges, n vertices) free from any circuit the minimum number of edges to be removed from G is

(a) $e-n$ (b) $e-n+1$ (c) $n-1$ (d) $e-1$

52. A graph has 15 vertices and 20 edges. The least number of edges to be removed from the graph to make it a tree is

(a) 13 (b) 5 (c) 19 (d) 6

53. The weight of the tree  is

(a) 14 (b) 4 (c) 10 (d) none

54. Minimal spanning tree in a graph is unique

(a) True (b) False

55. If every edge of a graph G has equal weight then

(a) minimal spanning tree is not obtained

(b) minimal spanning tree is unique

(c) every spanning tree is minimal spanning tree

(d) every tree will be spanning tree.

56. Minimal spanning tree is found by

(a) Disjkart's Algorithm (b) Ford-Fukerson's Algorithm

(c) Floyad-Algorithm (d) Kruskal's Algorithm

57. Minimal spanning tree is found by

(a) Disjkart's Algorithm (b) Prim's Algorithm

(c) Ford Fulkerson's Algorithm (d) none of these

58. The minimal spanning tree obtained by Kruskal's or Prim's Algorithm is unique.

(a) True (b) False

59. Every tree is a

(a) Circuit (b) Bipartite graph

(c) Complete graph (d) Regular graph

ANSWERS

1. b	2. a	3. c	4. d	5. a	6. c	7. b
8. b	9. a	10. b	11. c	12. a	13. b	14. c
15.a	16. b	17. a	18. a	19. d	20. c	21. b
22.b	23. a	24. b	25. b	26. c	27. d	28. c
29.a	30. b	31. a	32. b	33. c	34. d	35. c
36.a	37. c	38. c	39. b	40. d	41. a	42. b
43.b	44. c	45. d	46. a	47. a	48. c	49. b
50.a	51. b	52. d	53. a	54. b	55. c	56. d
57.b	58. b	59. b				