NEHRU COLLEGE OF ENGINEERING AND RESEARCH CENTRE
(NAAC Accredited)
(Approved by AICTE, Affiliated to APJ Abdul Kalam Technological University, Kerala)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

COURSE MATERIALS

MAT 206 GRAPH THEORY

VISION OF THE INSTITUTION

To mould true citizens who are millennium leaders and catalysts of change through excellence in education.

MISSION OF THE INSTITUTION

NCERC is committed to transform itself into a center of excellence in Learning and Research in Engineering and Frontier Technology and to impart quality education to mould technically competent citizens with moral integrity, social commitment and ethical values.

We intend to facilitate our students to assimilate the latest technological know-how and to imbibe discipline, culture and spiritually, and to mould them in to technological giants, dedicated research scientists and intellectual leaders of the country who can spread the beams of light and happiness among the poor and the underprivileged.

DEPARTMENT VISION

Producing Highly Competent, Innovative and Ethical Computer Science and Engineering professionals to facilitate continuous technological advancement.

DEPARTMENT MISSION

- To Impart Quality Education by creative Teaching Learning Process.
- To promote cutting-edge Research and Development Process to solve real world problems with emerging technologies.
- To Inculcate Entrepreneurship Skills among Students.
- To cultivate Moral and Ethical Values in their Profession.

PROGRAMME EDUCATIONAL OBJECTIVES

PEO1: Graduates will be able to Work and Contribute in the domains of Computer Science and Engineering through lifelong learning.

PEO2: Graduates will be able to Analyse, design and development of novel Software Packages, Web Services, System Tools and Components as per needs and specifications.

PEO3: Graduates will be able to demonstrate their ability to adapt to a rapidly changing environment by learning and applying new technologies.

PEO4: Graduates will be able to adopt ethical attitudes, exhibit effective communication skills, Teamworkand leadership qualities.

PROGRAM OUTCOMES (POS)

Engineering Graduates will be able to:

1. **Engineering knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities

with an understanding of the limitations.

6. **The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES (PSO)

PSO1: Ability to Formulate and Simulate Innovative Ideas to provide software solutions for Real-time Problems and to investigate for its future scope.

PSO2: Ability to learn and apply various methodologies for facilitating development of high quality System Software Tools and Efficient Web Design Models with a focus on performance optimization.

PSO3: Ability to inculcate the Knowledge for developing Codes and integrating hardware/software products in the domains of Big Data Analytics, Web Applications and Mobile Apps to create innovative career path and for the socially relevant issues.

COURSE OUTCOMES

C209.1	K2	**Understand** graph, its elements and their properties, types of paths, classification of graphs.
C209.2	K2	**Demonstrate** the fundamental theorems on Eulerian and Hamiltonian graphs.
C209.3	K4	**Illustrate** the working of Prim's and Kruskal's algorithms for finding minimum cost spanning tree and Dijkstra's and Floyd-Warshall algorithms for finding shortest Paths.
C209.4	K3	**Apply** the concept of graph connectivity, planar graphs and their properties on various theorems.
C209.5	K4	**Illustrate** the different graph representation and vertex color problem.

MAPPING OF COURSE OUTCOMES WITH PROGRAM OUTCOMES

CO Vs PO'S Mapping

CO'S	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
C209.1	3	2	3	2	-	-	-	-	-	-	-	-
C209.2	3	3	3	3	-	-	-	-	-	-	-	3
C209.3	3	3	3	3	-	-	-	-	-	-	-	3
C209.4	3	2	2		-	-	-	-	-	-	-	-
C209.5	3	3	3	3	-	-	-	-	-	-	-	3
C209	3	2.6	2.8	2.75	-	-	-	-	-	-	-	3

CO PSO'S Mapping

CO'S	PSO1	PSO2	PSO3
C209.1	3	2	2
C209.2	3	3	-
C209.3	3	3	-
C209.4	3	2	3
C209.5	3	-	3
C209	3	2.5	2.67

Note: H-Highly correlated=3, M-Medium correlated=2, L-Less correlated=1

SYLLABUS

CODE	COURSE NAME	CATEGORY	L	T	P	CREDIT
MAT 206	GRAPH THEORY	BSC	3	1	0	4

Preamble: This course introduces fundamental concepts in Graph Theory, including properties and characterisation of graph/trees and graph theoretic algorithms, which are widely used in Mathematical modelling and has got applications across Computer Science and other branches in Engineering.

Prerequisite: The topics covered under the course Discrete Mathematical Structures (MAT 203)

Course Outcomes: After the completion of the course the student will be able to

CO 1	Explain vertices and their properties, types of paths, classification of graphs and trees & their properties. **(Cognitive Knowledge Level: Understand)**
CO 2	Demonstrate the fundamental theorems on Eulerian and Hamiltonian graphs. **(Cognitive Knowledge Level: Understand)**
CO 3	Illustrate the working of Prim's and Kruskal's algorithms for finding minimum cost spanning tree and Dijkstra's and Floyd-Warshall algorithms for finding shortest paths. **(Cognitive Knowledge Level: Apply)**
CO 4	Explain planar graphs, their properties and an application for planar graphs. **(Cognitive Knowledge Level: Apply)**
CO 5	Illustrate how one can represent a graph in a computer. **(Cognitive Knowledge Level: Apply)**
CO 6	Explain the Vertex Color problem in graphs and illustrate an example application for vertex coloring. **(Cognitive Knowledge Level: Apply)**

Mark Distribution

Total Marks	CIE Marks	ESE Marks	ESE Duration
150	50	100	3 hours

Continuous Internal Evaluation Pattern:

Attendance : 10 marks

Continuous Assessment Tests : 25 marks

Continuous Assessment Assignment : 15 marks

Internal Examination Pattern:

Each of the two internal examinations has to be conducted out of 50 marks

First Internal Examination shall be preferably conducted after completing the first half of the syllabus and the Second Internal Examination shall be preferably conducted after completing remaining part of the syllabus.

There will be two parts: Part A and Part B. Part A contains 5 questions (preferably, 2 questions each from the completed modules and 1 question from the partly covered module), having 3 marks for each question adding up to 15 marks for part A. Students should answer all questions from Part A. Part B contains 7 questions (preferably, 3 questions each from the completed modules and 1 question from the partly covered module), each with 7 marks. Out of the 7 questions in Part B, a student should answer any 5.

End Semester Examination Pattern: There will be two parts; Part A and Part B. Part A contain 10 questions with 2 questions from each module, having 3 marks for each question. Students should answer all questions. Part B contains 2 questions from each module of which student should answer anyone. Each question can have maximum 2 sub-divisions and carries 14 marks.

Syllabus

Module 1

Introduction to Graphs : Introduction- Basic definition – Application of graphs – finite, infinite and bipartite graphs – Incidence and Degree – Isolated vertex, pendant vertex and Null graph. Paths and circuits – Isomorphism, sub graphs, walks, paths and circuits, connected graphs, disconnected graphs and components.

Module 2

Eulerian and Hamiltonian graphs : Euler graphs, Operations on graphs, Hamiltonian paths and circuits, Travelling salesman problem. Directed graphs – types of digraphs, Digraphs and binary relation, Directed paths, Fleury's algorithm.

Module 3

Trees and Graph Algorithms : Trees – properties, pendant vertex, Distance and centres in a tree - Rooted and binary trees, counting trees, spanning trees, Prim's algorithm and Kruskal's algorithm, Dijkstra's shortest path algorithm, Floyd-Warshall shortest path algorithm.

Module 4

Connectivity and Planar Graphs : Vertex Connectivity, Edge Connectivity, Cut set and Cut Vertices, Fundamental circuits, Planar graphs, Kuratowski's theorem (proof not required), Different representations of planar graphs, Euler's theorem, Geometric dual.

Module 5

Graph Representations and Vertex Colouring : Matrix representation of graphs- Adjacency matrix, Incidence Matrix, Circuit Matrix, Path Matrix. Coloring- Chromatic number, Chromatic polynomial, Matchings, Coverings, Four color problem and Five color problem. Greedy colouring algorithm.

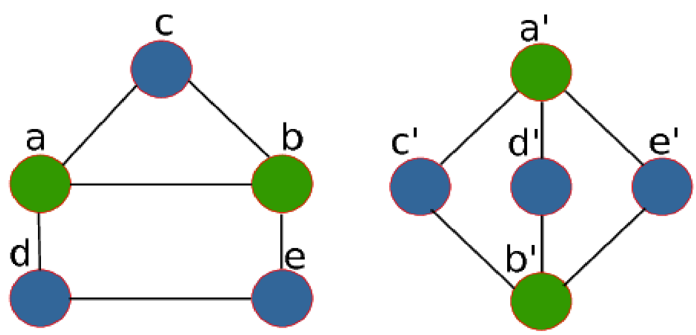
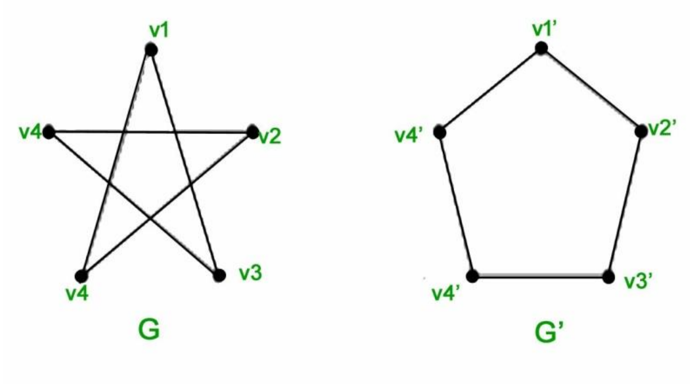
Text book:

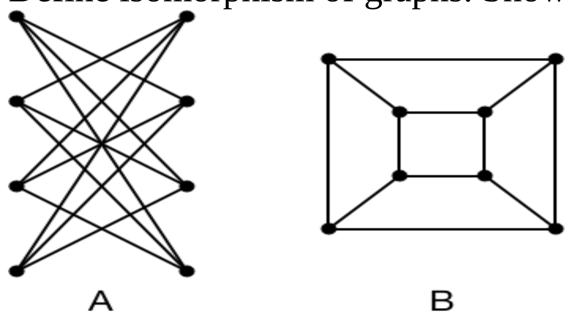
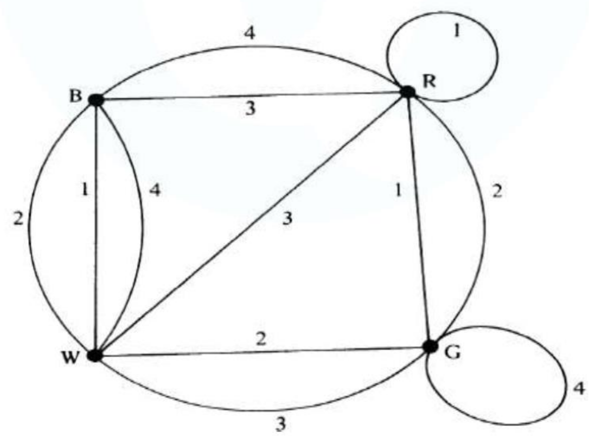
1. Narsingh Deo, Graph theory, PHI,1979

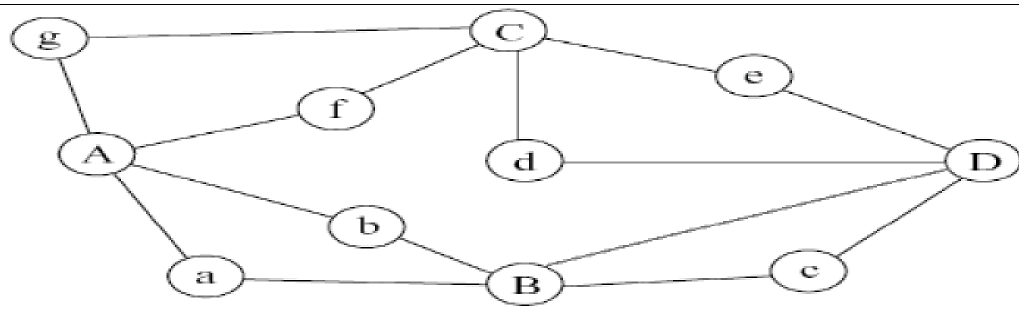
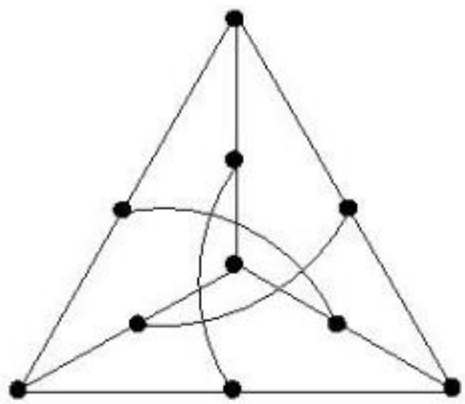
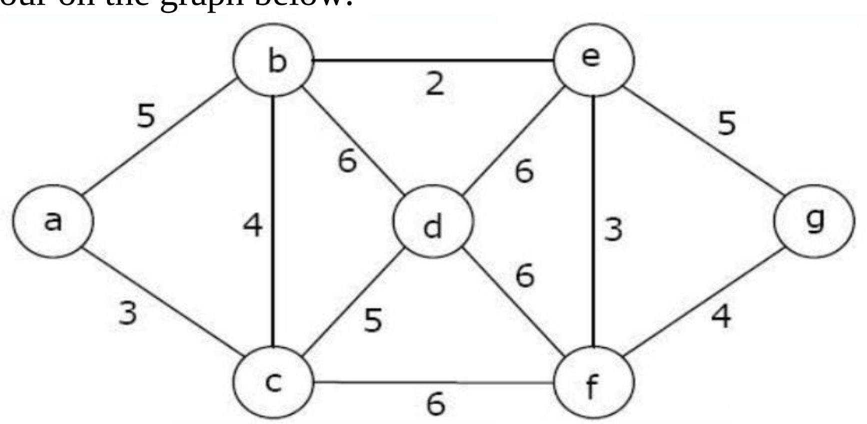
Reference Books:

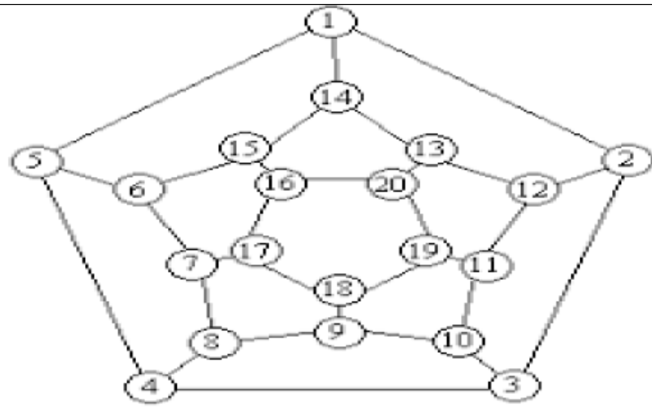
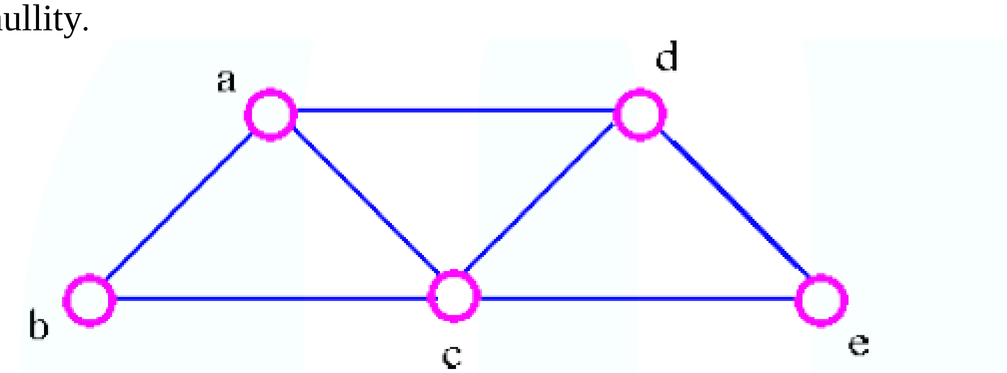
1. R. Diestel, *Graph Theory,* free online edition, 2016: diestel-graph-theory.com/ basic.html.
2. Douglas B. West, Introduction to Graph Theory, Prentice Hall India Ltd.,2001
3. Robin J. Wilson, Introduction to Graph Theory, Longman Group Ltd.,2010
4. J.A. Bondy and U.S.R. Murty. Graph theory with Applications

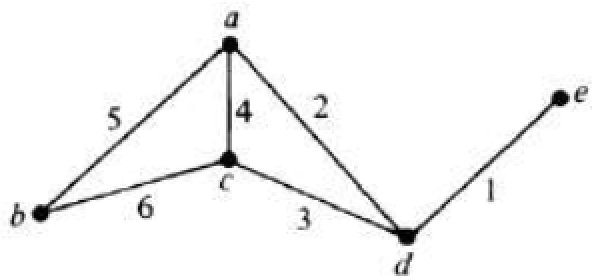
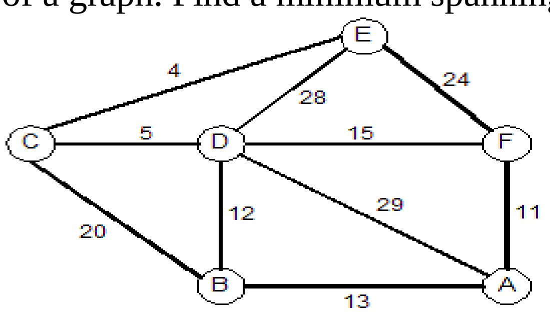
QUESTION BANK

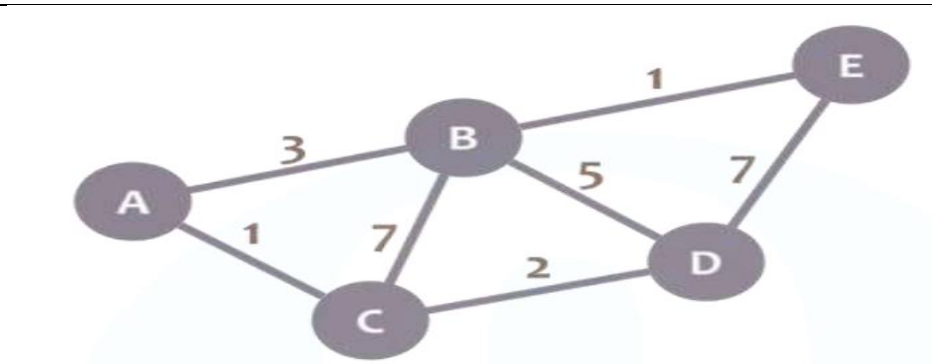
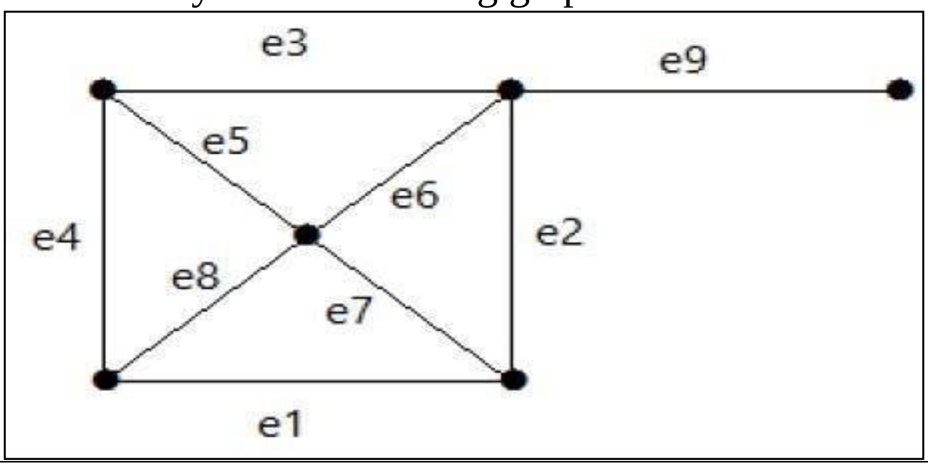
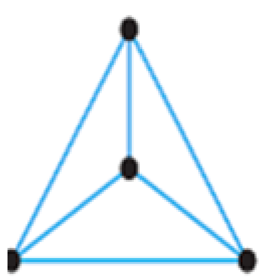
	MODULE I		
	QUESTIONS	**CO**	**KL**
1	Define pendant vertex, isolated vertex and null graph with an example each.	CO1	K1
2	Discuss isomorphism of graphs. Show that the graphs are isomorphic. 	CO1	K2
3	Explain about Walk and Path with examples.	CO1	K2
4	Write and explain Edge Disjoint and Vertex Disjoint Sub graphs with Example.	CO1	K3
5	Differentiate Circuit and Cycle With Example.	CO1	K4
6	Define isomorphism of graphs. Show that the graphs are isomorphic. 	CO1	K1
7	Prove that a simple graph with n vertices and k components can have	CO1	K5

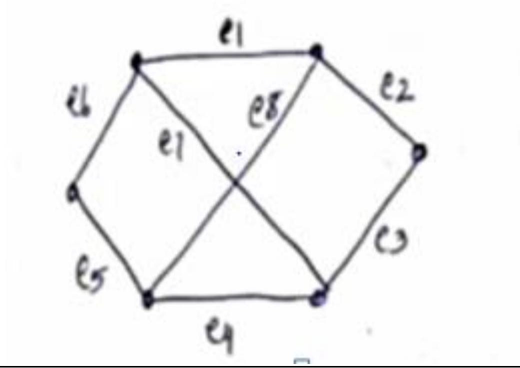
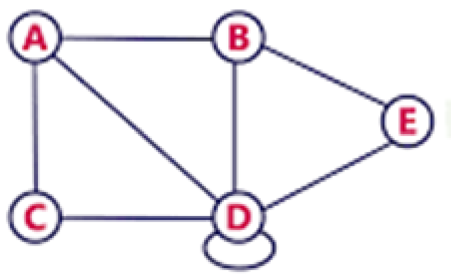
	at most (n-k)(n-k+1)/2 edges		
8	Write and explain 4 Different applications of Graphs.	CO1	K3
9	Consider a graph G with 5 vertices: v1,v2, v3, v4 and v5 and the degrees of vertices are 5,4,3,1 and 2 respectively. Is it possible to construct such a graph G? Justify your answer.	CO1	K3
10	Define isomorphism of graphs. Show that the graphs are isomorphic. 	CO1	K1
11	Explain about Isolated Vertices and Pendant Vertices of a Graph in detail.	CO1	K2
12	Define sub graphs. What are edge disjoint and vertex disjoint sub graphs? Construct two edge disjoint sub graphs of the graph G. 	CO1	K1
	MODULE II		
1.	Test whether the given graph is an Euler graph and if yes, give the Euler line. Justify your answer.	CO2	K4

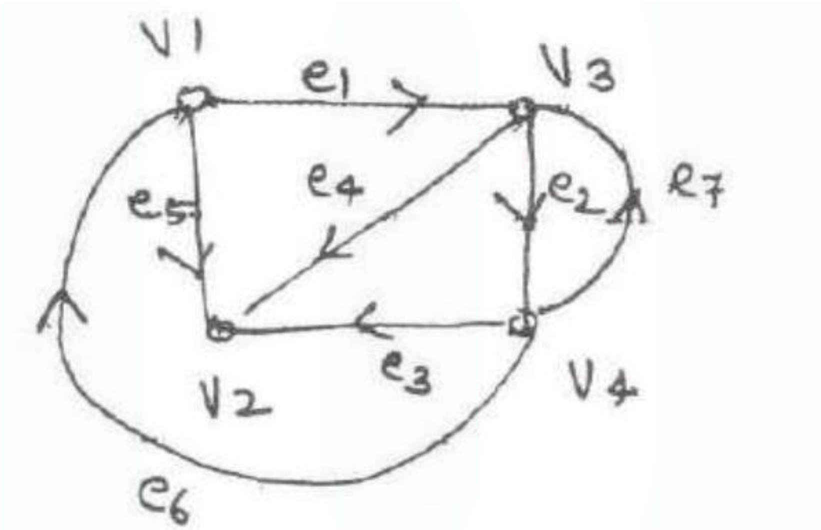
			
2.	Define Hamiltonian Graph. Show that the given graph is Hamiltonian or not? 	CO2	K1
3.	Illustrate travelling salesman problem. Print a travelling salesman’s tour on the graph below. 	CO2	K3
4.	Explain about digraphs and its types with neat diagrams.	CO2	K2
5.	Prove that a given connected graph G is an Euler Graph if and only if all vertices of G are of even degree SIC Assembler Algorithm	CO2	K5
6.	Give Hamiltonian circuit of the following graph.	CO2	K2

			
7	Illustrate Digraphs with example.	CO2	K3
8	Explain about Operations of Graph-Union, Intersection, Ring sum, Decomposition and deletion with examples.	CO2	K2
9	Explain Euler Graphs with example	CO2	K2
10	Explain about unicursal graphs with example	CO2	K2
11	Explain about Arbitrarily Traceable Graph in detail.	CO2	K2
12	Explain digraphs and binary relation on digraphs.	CO2	K2
MODULE III			
1	Explain Eccentricity of a tree	CO3	K2
2	Illustrate about Spanning tree. Find any two spanning trees T1, T2 of the graph G given below. Also find the branch set, chord set, rank and nullity. 	CO3	K3
3	Explain Kruskal's Algorithm with an example.	CO3	K2

4	Explain Center of a tree and how to find the center.	CO3	K2
5	Explain the concept of Rooted Tree and Binary Tree in detail.	CO3	K2
6	Prove that every tree has either one or two centers. Sketch all spanning trees of the given graph  	CO3	K5
7	Prove that in a graph G, if there is exactly one path between every pair of vertices, then G is a tree.	CO3	K5
8	Write note on minimum and maximum levels of a tree. Sketch two different binary trees on 13 vertices, one having maximum height and other having minimum height.	CO3	K3
9	Prove that there are n^{n-2} labeled trees with n vertices (n>=2)	CO3	K5
10	Discuss Kruskal's algorithm used to find minimum cost spanning tree of a graph. Find a minimum spanning tree in the graph below.  	CO3	K2
11	Define Rank and Nullity of a Graph	CO3	K1
12	Illustrate about Fundamental Circuits.	CO3	K3
13	Write Dijkstra's Shortest path algorithm and apply this algorithm to find the shortest path	CO3	K3

			
	MODULE IV		
1	Define Planar Graph with example	CO4	K1
2	Give the fundamental cutset ,edge connectivity and vertex connectivity of the following graph 	CO4	K2
3	Prove that Kuratowiskis first graph and second graph is non planar	CO4	K5
4	Find out the geometrical dual of the given graph 	CO4	K1
5	Differentiate Fundmental Circuits and fundamental cutsets with an example	CO4	K4
6	Explain about Geometric dual .Find out the dual of the given graph	CO4	K5

7	Compare Vertex connectivity and Edge connectivity	CO4	K4
8	Describe fundamental circuit and cutsets of the following graph 	CO4	K2
9	Prove that a Connected Planar Graph with n vertices and e edges has e-n+2 regions	CO4	K5
MODULE V			
1	Explain Incidence Matrix with an example	CO5	K2
2	Illustrate Adjacency Matrix and Incidence Matrix. Find out the adjacency matrix and incidence matrix of the given graph. 	CO5	K3
3	Explain in detail about Graph colouring and chromatic number with examples	CO5	K2
4	Explain chromatic number of planar graphs and complete graphs with example	CO5	K2
5	Prove that Every Tree with two or more vertices is 2 Chromatic and A graph with atleast one edge is 2 chromatic if and only if it has no circuits of add length	CO5	K5
6	Discuss the chromatic polynomial of graph with 5 vertices .	CO5	K2
7	Illustrate Circuit Matrix with an example	CO5	K3
8	Write note on Chromatic polynomial with an example.	CO5	K3

9	Give the incidence matrix of the graph G. Also write the properties of incidence matrix 	CO5	K2

APPENDIX 1	
CONTENT BEYOND THE SYLLABUS	
SL NO	**TOPIC**
1	**Application of graphs in solving various Engineering problems**
2	**Importance of graph theoretic algorithms in different fields of our society**

MODULE NOTES

Introduction to graph

Basic definition :-

A linear graph $G = (V, E)$ consists of a set of objects $V = \{v_1, v_2, \ldots\}$ called vertices and another set $E = \{e_1, e_2, \ldots\}$, whose elements are called edges, such that each edge e_k is identified with an unordered pair (v_i, v_j) of vertices.

The vertices v_i, v_j associated with edge e_k are called the end of vertices of e_k.

The most common representation of graph is by means of a diagram, in which the vertices are represented as points and each edge as a line segment joining its end vertices

Sample graph is shown below.

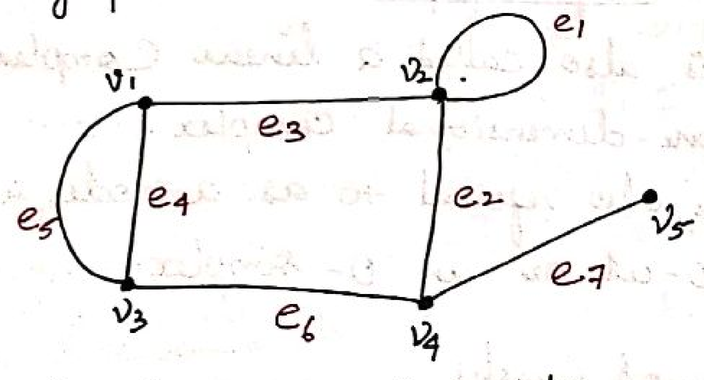


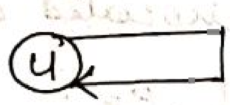
fig: Graph with five vertices and seven edges.

There are mainly two type of Edges

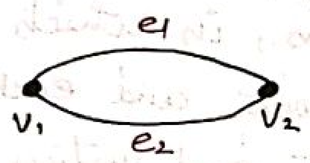
Directed :- Ordered pair of vertices. Represented as (u, v) directed from vertex u to v.

undirected : Unordered pair of vertices. Represented as {u, v}. Disregards any sense of direction and treats both end vertices interchangeably

An edge to be associated with a vertex pair (v_i, v_i) Such an edge having the same vertex as both its end vertices is called a <u>self loop</u>. Or an edge joining a vertex to itself is called a <u>loop</u> (self-loop)



If more than one edge associated with a given pair of vertices is called <u>parallel Edges</u>



A graph that has neither self-loops nor parallel edges is called a <u>simple graph</u>

A graph is also called a linear complex, a 1-complex or a one-dimensional complex.

A vertex is also referred to as a node, a junction, a point, 0-cell or an 0-simplex.

<u>Applications of Graphs</u>

Graph Theory has a wide range of applications in engineering, physical, social and biological science. Following are the four examples among hundreds of such applications

Konigsberg Bridge problem
utilities problem
Electrical Network problems
Seating problem.

Konigsberg Bridge problem.

This is the best known example in graph theory. It was a long standing problem solved by Leonhard Euler by means of a graph.

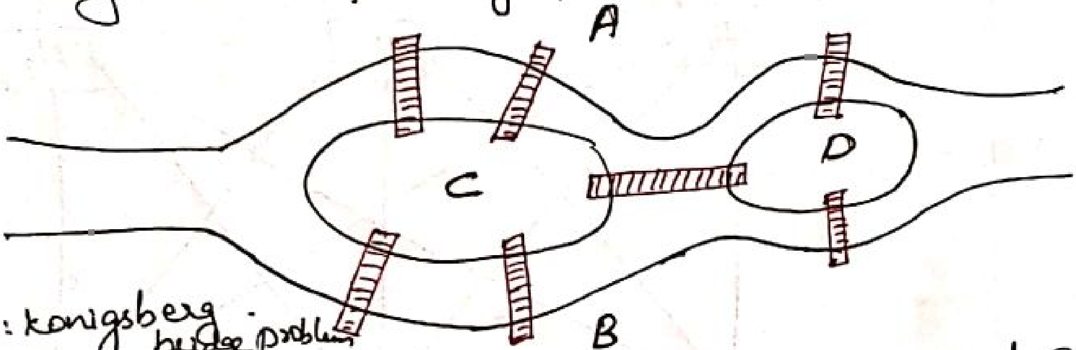


fig: konigsberg bridge problem

Two islands C and D formed by the pregel River in Konigsberg were connected to each other and to the banks A and B with seven Bridges. The problem was to start at any of the four land areas of the city A, B, C or D walk over each of the seven bridges exactly once and returning to the starting point.

Euler represented this situation by means of the following graph. The vertices represent the land areas and the edges represent the bridges

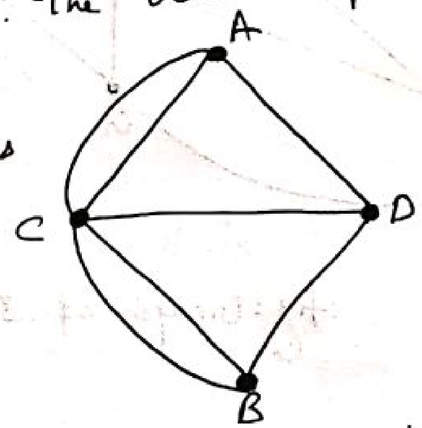


fig: Graph of konigsberg bridge

The konigsberg bridge problem is the same as the problem of drawing figures without lifting the pen from the paper and without retracing a line.

<u>Utilities problem:-</u>

There are three houses H_1, H_2 and H_3 each to be connected to each of the three utilities - water(W) gas (G) and electricity (E)

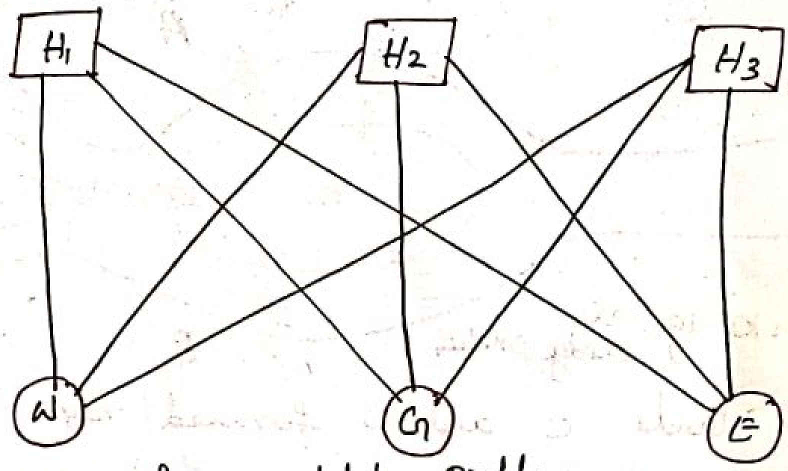


fig: 3-utilities problem.

The following graph represents the utilities problem

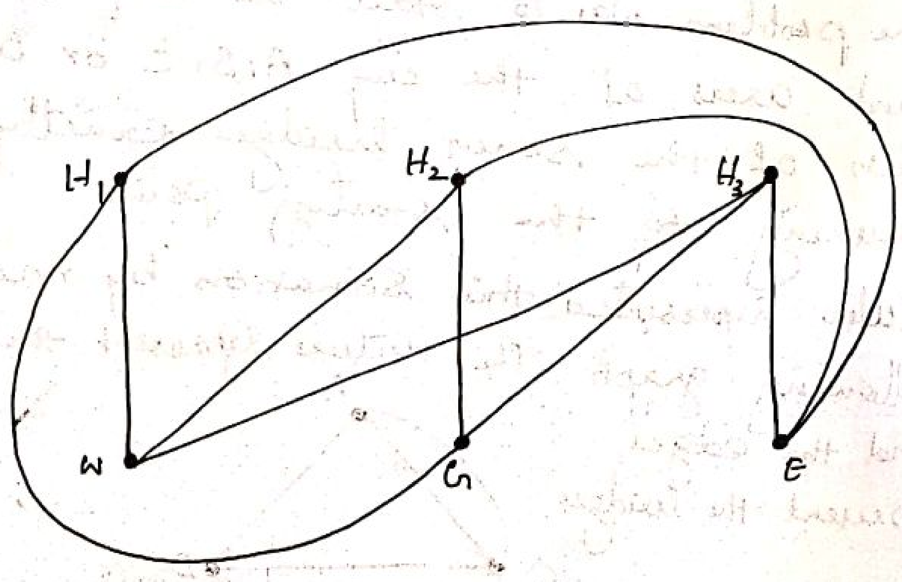


fig: Graph of 3-utilities problem

The above graph cannot be drawn in the plane without edges crossing over. Thus the answer to the problem is no.

Electrical Network problems:-

properties of an electrical network are functions of only two factors.

1. The nature and value of the elements forming the network such as resistors, inductors, transistors

2. The way these elements are connected together, that is the topology of the network.

A graph of an electrical network, the junctions are represented by vertices and branches are represented by edges, regardless of the nature and size of the electrical elements. An electrical network and the graph are shown below

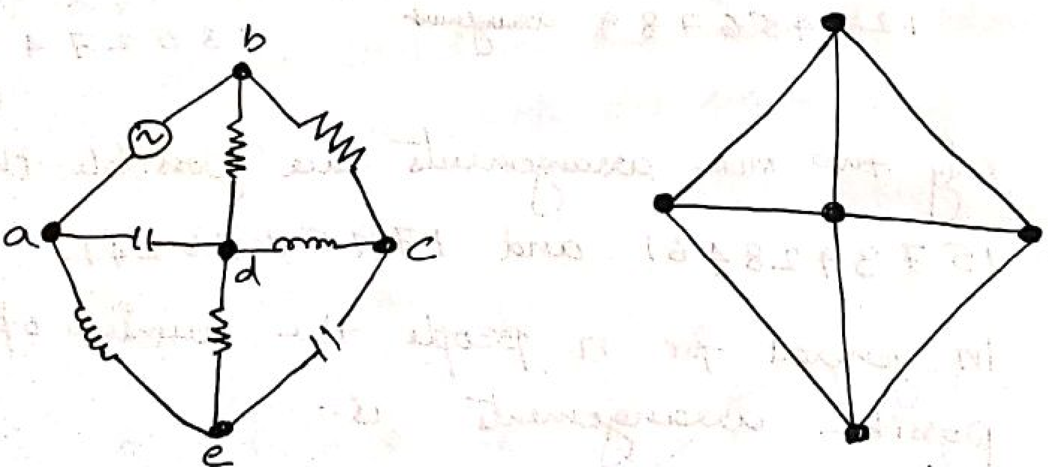
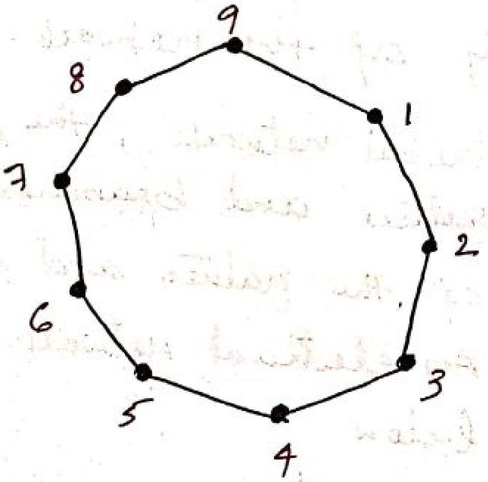


fig: Electrical Network and its graph

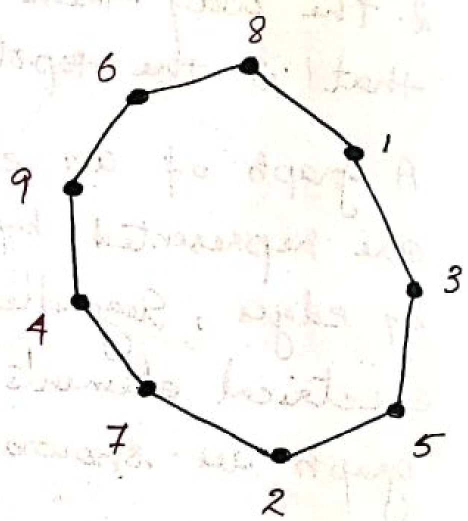
Seating problem

Nine members of a new club meet each day for lunch at a round table. They decide to sit such that every member has different neighbors at each lunch. How many days can this arrangement last?

This situation can be represented by a graph with nine vertices such that each vertex represents a member and an edge joining two vertices represent the relationship of sitting near to each other. The following figures shows two possible arrangements



1 2 3 4 5 6 7 8 9 arrangement



1 3 5 2 7 4 9 6 8 arrangements

Only two more arrangements are possible. They are 1 5 7 3 9 2 8 4 6 1 and 1 7 9 5 8 3 6 2 4 1.

In general for n people the number of such possible arrangements is:

$\frac{n-1}{2}$ for n is odd.

$\frac{n-2}{2}$ for n is even.

Finite and Infinite Graphs

A graph with a finite number of vertices and finite number of edges is called a <u>finite graph</u>. Otherwise if a graph contains infinite number of vertices and infinite number of edges then it is called <u>infinite graph</u>.

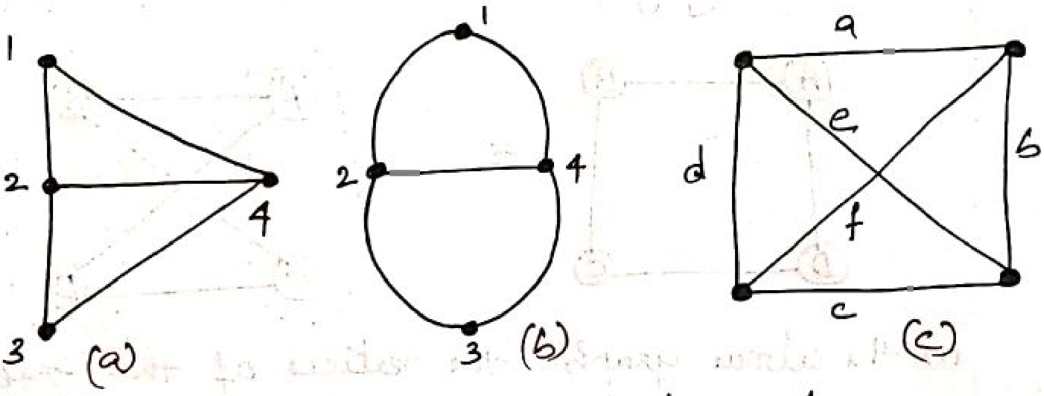


fig: Examples of finite graphs.

Note :-
In the above graph (a) and (b) the lines are drawn as straight and curved. But the two graph are same since the incidence between the edges and vertices of the two graph are same.

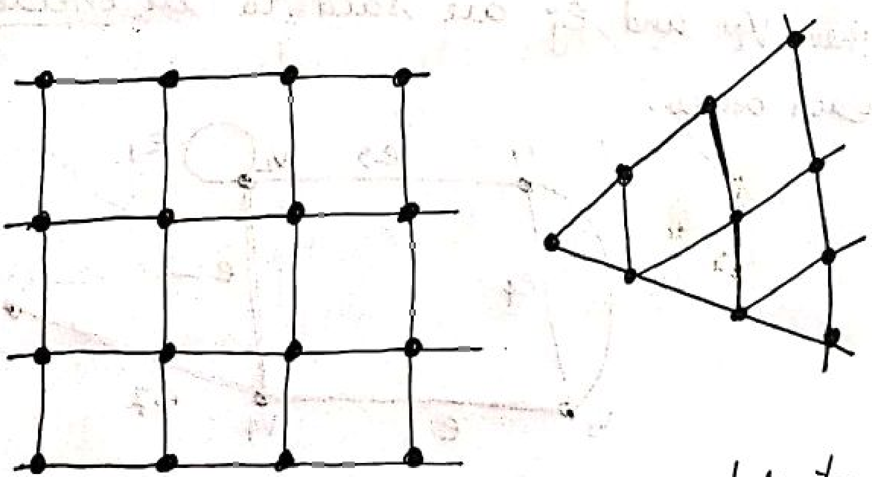


fig: Examples of Infinite graph

Bipartite Graph

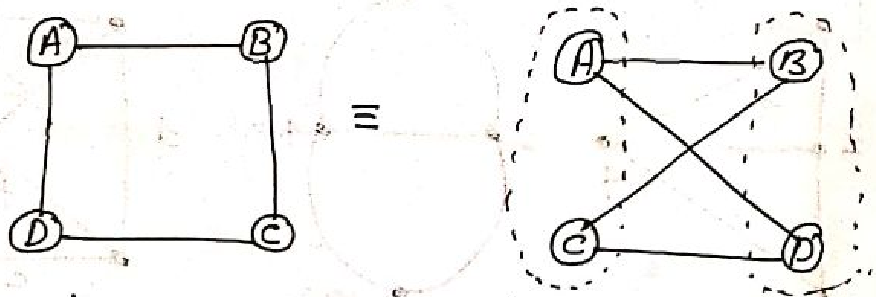
A bipartite graph is a special kind of graph with the following properties.

→ It consists of two sets of vertices x and y.

→ The vertices of set x join only with the vertices of set y

→ The vertices within the same set do not join

The following graph is an example of a bipartite graph



In the above graph the vertices of the graph can be decomposed into two sets.

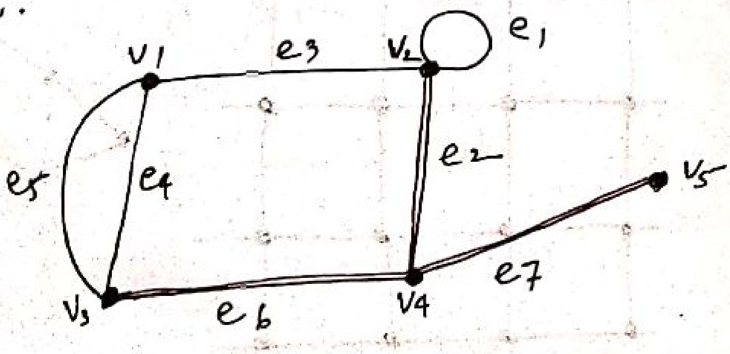
The two sets are $X = \{A, C\}$ and $Y = \{B, D\}$

The vertices of set X join only with the vertices of set Y and vice versa.

The vertices within the set X & Y do not join

Incidence and Degree

When a vertex v_i is an end vertex of some edge e_j, then v_i and e_j are said to be incident with each other.



In the above graph, edge e_2, e_6 and e_7 are incident with vertex V_4.

Two nonparallel edges are said to be adjacent if they are incident on a common vertex.

eg: In the above graph edge e_2 and e_7 are incident on the vertex V_4. so e_2 and e_7 are adjacent edges.

In the above graph the adjacent edges are

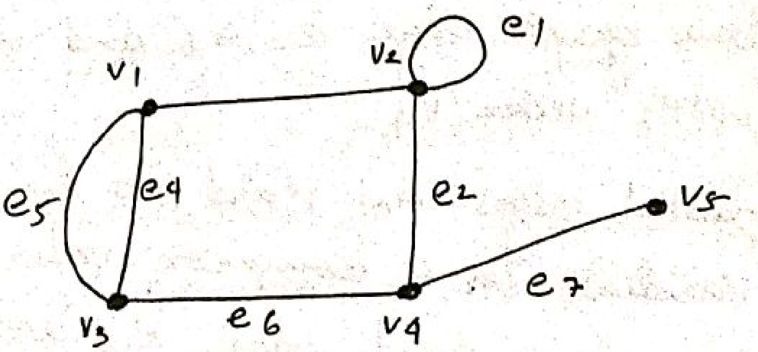
e_2, e_3
e_1 e_2
e_1 e_3
e_3 e_4
e_5 e_3
e_4 e_6
e_5 e_6
e_6 e_2
e_6 e_7

Two vertices are said to be adjacent if they are the end vertices of the same edge.

Eg: In the above graph V_4 and V_5 are adjacent vertices. since their edge e_7 is having end vertices as V_4 and V_5

But the vertices V_1 and V_4 are not adjacent becoze these vertices do not share a common edge.

The number of edges incident on a vertex V_i, with self loops counted twice is called degree $d(\bar{V_i})$ of vertex V_i



In the above graph $d(v_1) = d(v_3) = d(v_4) = 3$

$d(v_2) = 4$ and $d(v_5) = 1$

The degree of vertex is sometimes referred as <u>Valency</u>

Consider a graph G with e edges and n vertices $v_1, v_2, \ldots v_n$. Since each edge contribute two degree the sum of the degrees of all vertices in G is twice the number of edges in G. That is

$$\boxed{\sum_{i=1}^{n} d(v_i) = 2e}$$

In the above given graph there are 5 vertices v_1, v_2, v_3, v_4 and v_5 and total 7 edges.

$d(v_1) + d(v_2) + d(v_3) + d(v_4) + d(v_5) = 3+4+3+3+1 = 14$

which is twice the number of edges

<u>Theorem - I</u>

The number of vertices of odd degree in a graph is always even.

Proof:-

If we consider the vertices with odd and even degrees separately, the quantity in the left side of $\sum_{i=1}^{n} d(v_i) = 2e$ can be expressed as the sum of two sums, each taken over vertices of even and odd degrees, respectively.

$$\sum_{i=1}^{n} d(v_i) = \sum_{even} d(v_j) + \sum_{odd} d(v_k) \rightarrow (1)$$

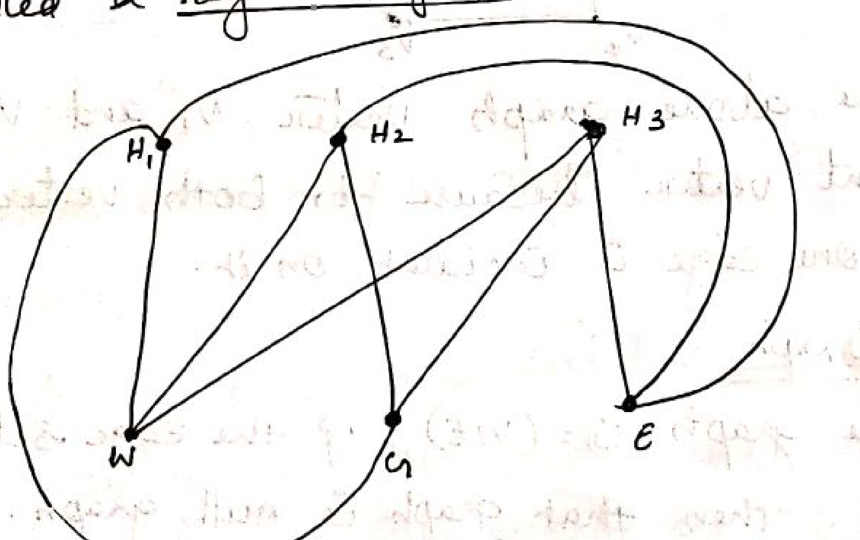
since the left-hand side in equ (1) is even and the first expression on the right-hand side is even (being a sum of even numbers), the second expression must also be even:

$$\sum_{odd} d(v_k) = \text{an even number.} \rightarrow (2)$$

In equ (2) each $d(v_k)$ is odd, the total number of terms in the sum must be even to make the sum an even number. Hence the theorem proved.

A graph in which all vertices are of equal degree is called a Regular graph

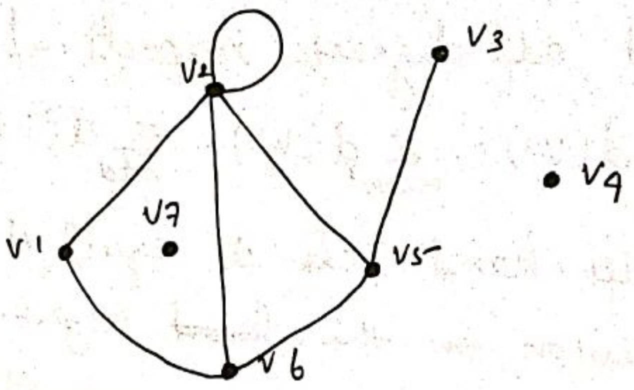
Eg:



In this graph all vertices are of same degree $d(v)=3$.

Isolated Vertex

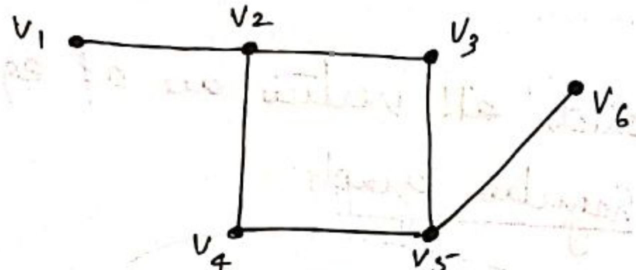
A vertex having no incident edge is called an isolated vertex. or Isolated vertices are vertices with



In the above graph vertex V_4 and V_7 are isolated vertex because no edges are incident on both the vertex V_4 and V_7.

Pendant Vertex

A vertex of degree one is called a pendant vertex or an end vertex



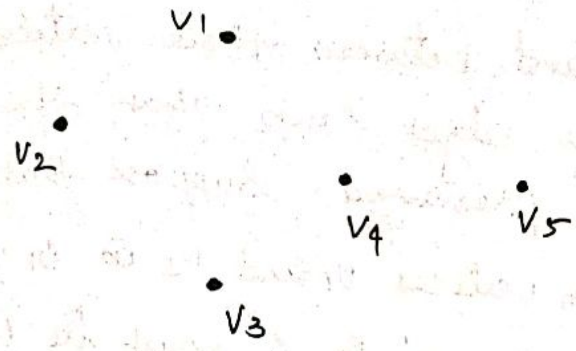
In the above graph vertex V_1 and V_6 are pendant vertex because for both vertex V_1 and V_6 only one edge is incident on it.

Null Graph

In the graph $G = (V, E)$; If the edge set E is empty. Then that graph is null graph. or A Graph without any edge is considered as null graph

In null graph every vertex is an isolated vertex.

the following graph shows an example of null graph.



In the above all 5 vertices are isolated vertex and there is no edge present in that graph. i.e. $E = \{\phi\}$. So the given graph is null graph.

Note:-

In null graph, the edge set may be empty, but not the vertex set V, if set V is empty then there is no graph. i.e A null graph must have atleast one vertex.

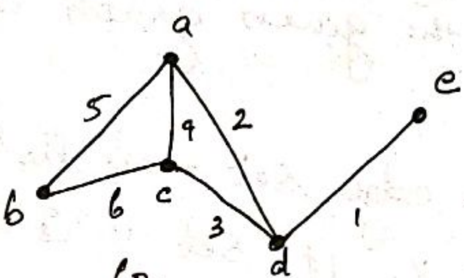
Paths and circuits :-

Isomorphism

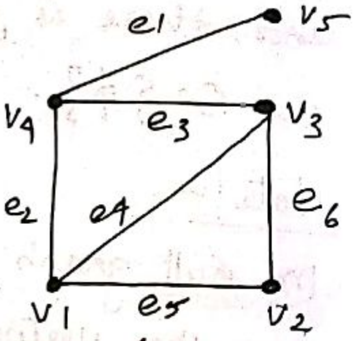
The word meaning of isomorphism is the identical behaviour.

A graph can exist in different forms having the same number of vertices, edges and also the same edge connectivity. Such graphs are called isomorphic graphs.

Two graphs G and G' are said to be isomorphic if there is a one-to-one correspondence between their vertices and between their vertices and between their edges such that the incidence relationship is preserved. Suppose that edge e is incident on vertices v_1 and v_2 in G; then the corresponding edge e' in G' must be incident on the vertices v_1' and v_2' that correspond to v_1 and v_2.



G_1



G_2

The above two given graphs are isomorphic graphs. The one to one correspondence between the two graphs is as follows.

The vertices in G_1 a, b, c, d and e correspond to the vertices in G_2 V_1, V_2, V_3, V_4 and V_5. The edges 1, 2, 3, 4, 5 and 6 correspond to e_1, e_2, e_3, e_4, e_5 and e_6.

Total vertices in G_1 = 5	Total vertices in G_2 = 5
Total edges in G_1 = 6	Total edges in G_2 = 6
Deg (a) = 3	Deg (V_1) = 3
Deg (b) = 2	Deg (V_2) = 2
Deg (c) = 3	Deg (V_3) = 3
Deg (d) = 3	Deg (V_4) = 3
Deg (e) = 1	Deg (V_5) = 1
a b c form circuit with length 3	$V_1 V_2 V_3$ forms a circuit with length 3
a d c form circuit of length 3	$V_1 V_4 V_3$ form a circuit

Therefore the above graph maintains the one to one correspondence and is an isomorphic graph

Consider the following two graphs.

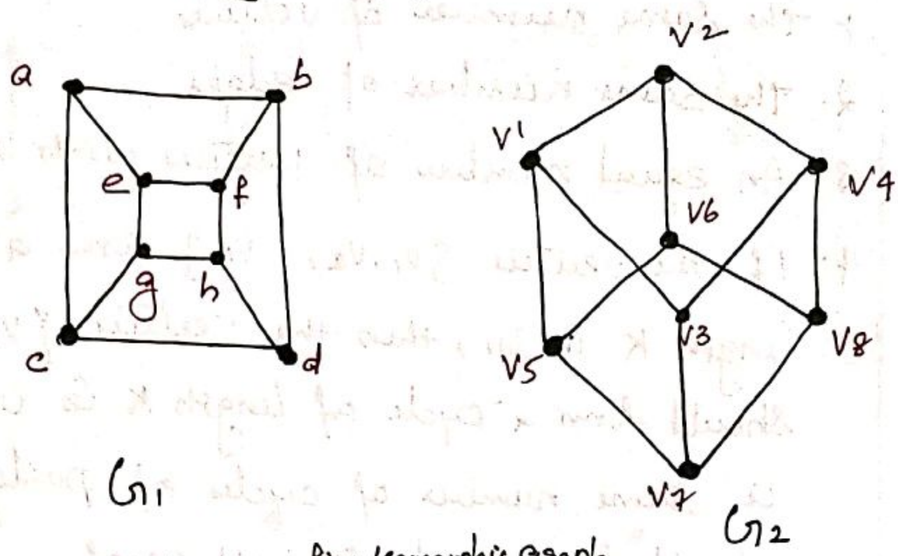


fig: Isomorphic graph

The above 2 graph consists of 8 vertices & 12 edges. All vertices possess 3 degree. The following is the one to one correspondence.

a – V_1
b – V_2
c – V_3
d – V_4
e – V_5
f – V_6
g – V_7
h – V_8.

abcd form 4 length circuit – $V_1V_2V_3V_4$ form 4 length circuit

efgh form 4 length circuit – $V_5V_6V_7V_8$ form 4 length circuit

acge form 4 length circuit – $V_1V_3V_5V_7$ form 4 length circuit

Since all the criterias of isomorphism are satisfied by the above two graphs. They are considered as isomorphic graph.

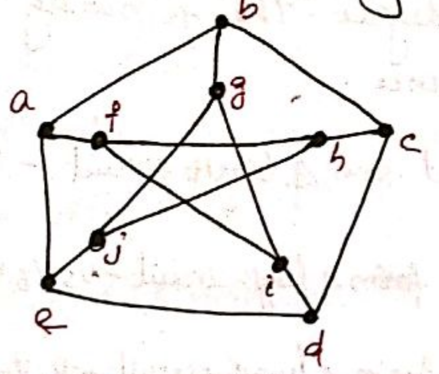
Definition of isomorphism :-

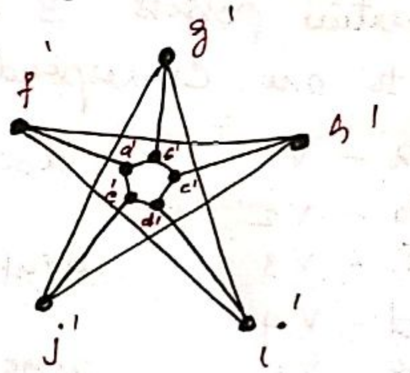
Two graphs G_1 and G_2 are said to be isomorphic if it possess the following properties

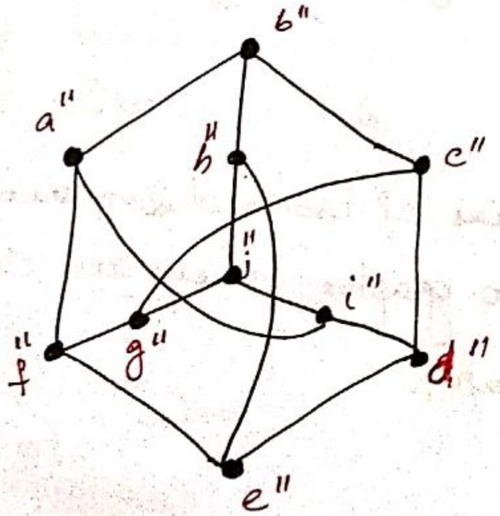
1. The same number of vertices
2. The same number of edges
3. An equal number of vertices with a given degree
4. If the vertices $\{v_1, v_2, \ldots v_k\}$ forms a cycle of length k in G_1, then the vertices $\{v_1', v_2', \ldots v_k'\}$ should form a cycle of length k in G_2 also.

   i.e. same number of cycles of particular length should be present in both graph.

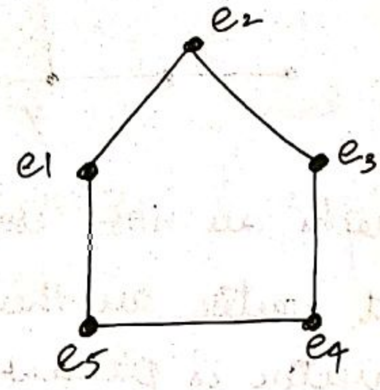
Consider the following 3 graphs.

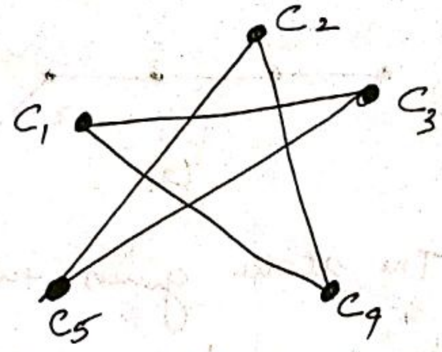


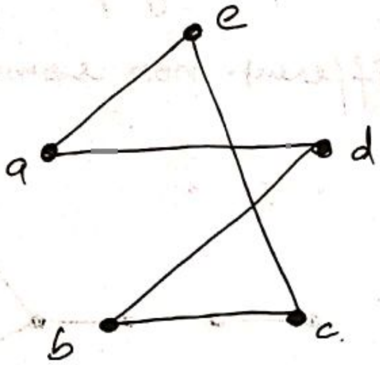


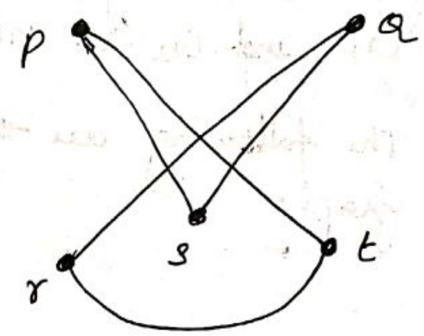


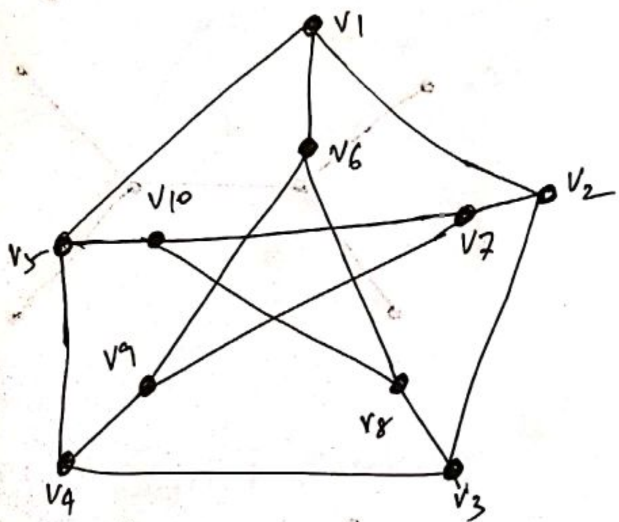
Following are the different set of isomorphic graph

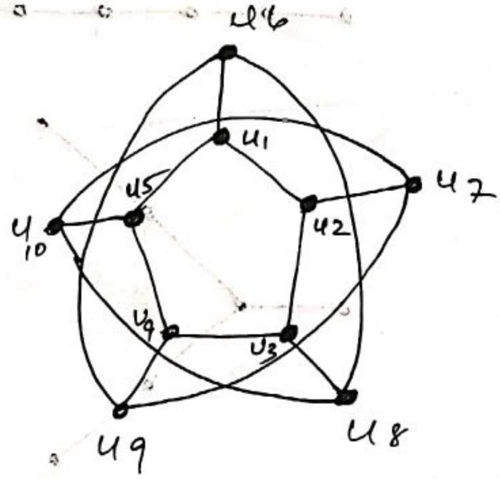


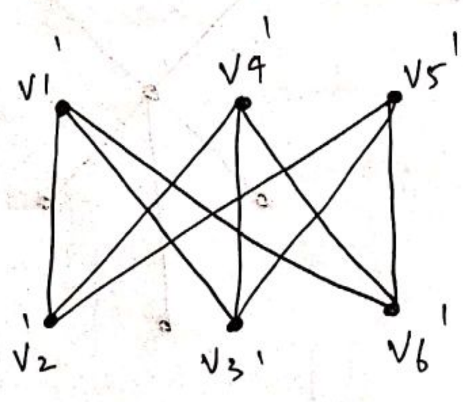


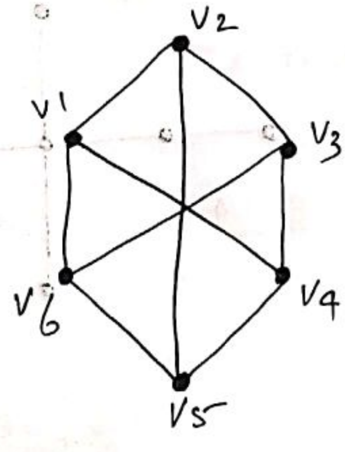




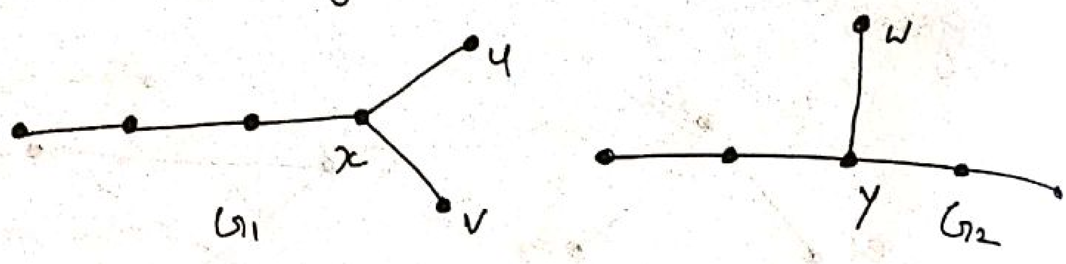






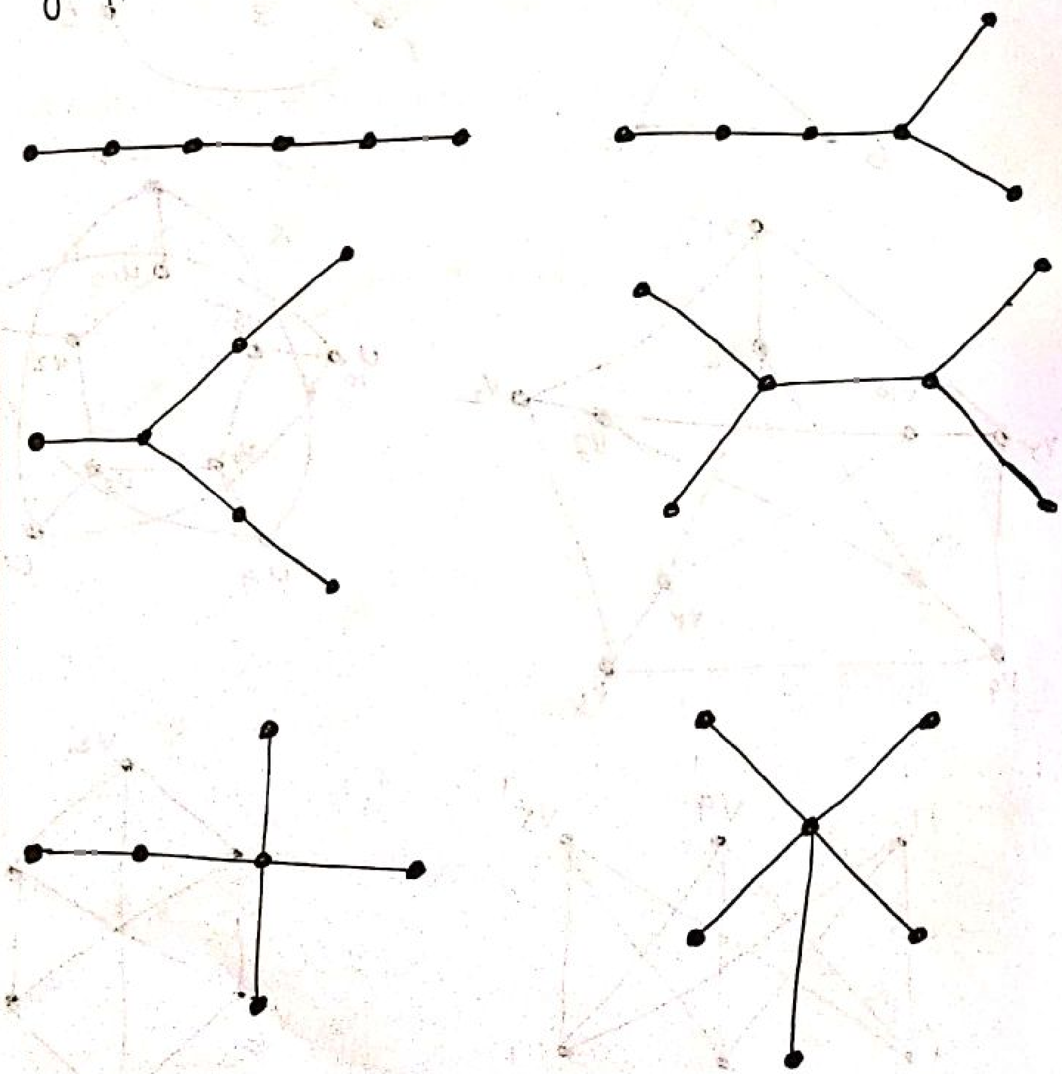


Non isomorphic graphs Example.



The above given two graphs are not isomorphic. In G_1 graph two pendant vertex are there but in G_2 only one pendant vertex is present. So G_1 and G_2 are not isomorphic graph.

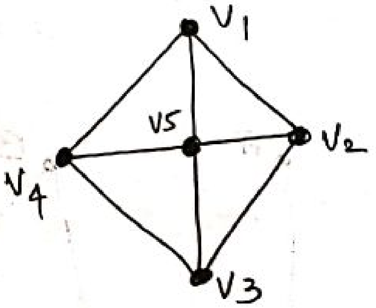
The following are the different non isomorphic graphs.



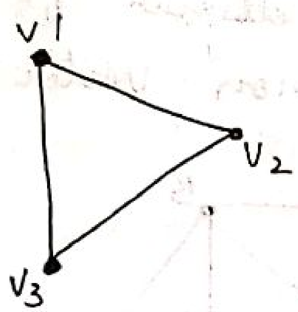
Subgraphs

A graph 'g' is said to be a Subgraph of the graph 'G' if all the vertices and all the edges of g are in G and each edge of g has the same end vertices in g as in G.

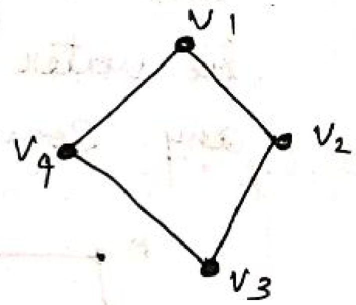
Example:-



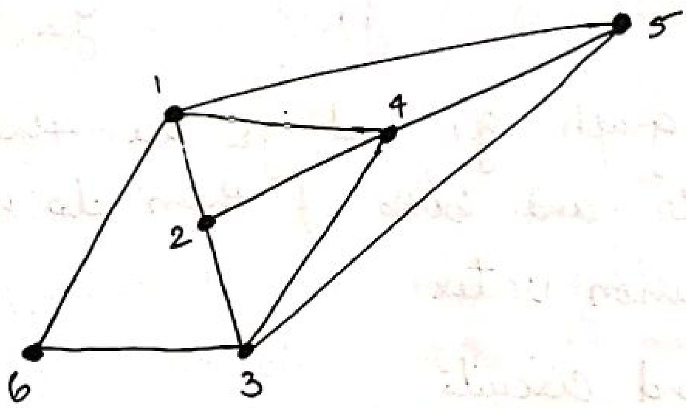
G



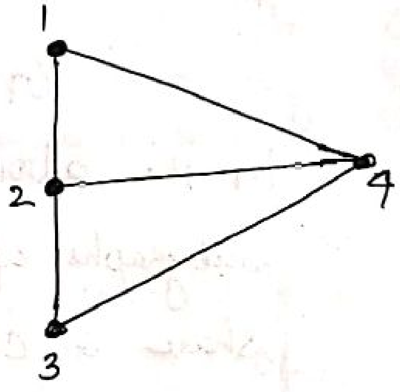
g_1



g_2



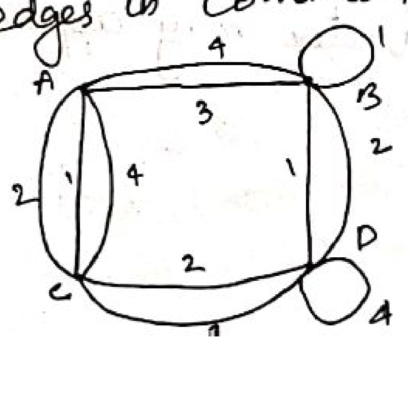
G



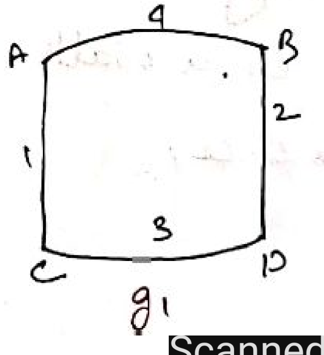
g_1

Edge Disjoint Subgraph

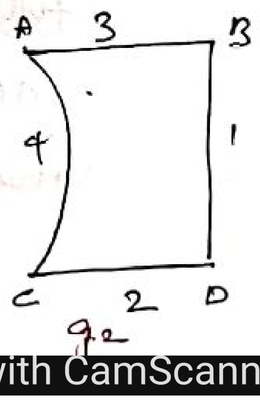
Two Subgraphs g_1 and g_2 of a graph G are said to be edge disjoint if g_1 and g_2 do not have any edges in common.



G



g_1

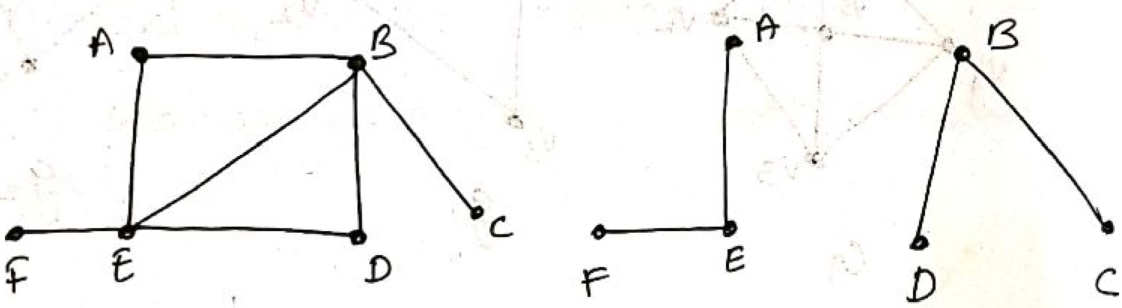


g_2

In the above graphs, g_1 and g_2 are the edge disjoint subgraph of G. The subgraph g_1 and g_2 do not have any common edge.

Vertex disjoint Graph

Two subgraph g_1 and g_2 are said to be vertex disjoint if they do not contain any common vertex.

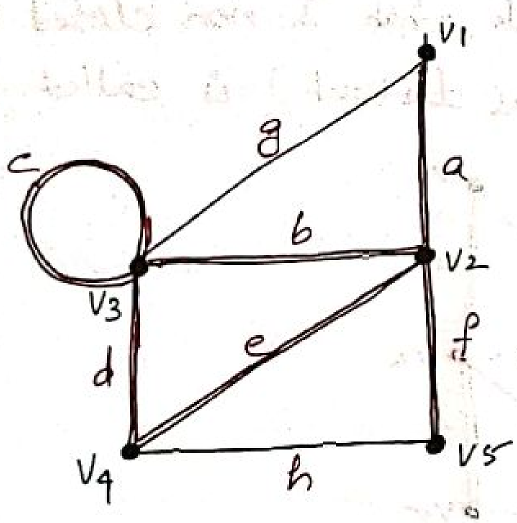


G $\qquad$ g_1 $\qquad$ g_2

In the above graph g_1 and g_2 are the subgraphs of G and both of them do not share a common vertex.

Walks, path and circuits

A walk is defined as a finite alternating sequence of vertices and edges beginning and ending with vertices, such that each edge is incident with the vertices preceding and following it. No edge appears more than once in a walk. A vertex may appear more than once.

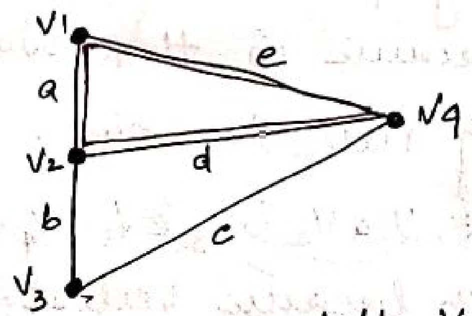


In the above given graph $V_1 a V_2 b V_3 c V_3 d V_4 e V_2 f V_5$ is a walk as shown in the figure.

A walk is referred to as an edge train or a chain. The set of vertices and edges constituting a given walk in a graph G is clearly a subgraph of G. Vertices with which a walk begins and ends are called its <u>terminal vertices</u>.

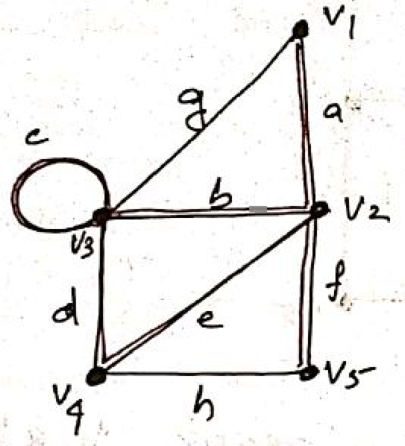
In the walk $V_1 a V_2 b V_3 c V_3 d V_4 e V_2 f V_5$ of the above graph vertices V_1 and V_5 are the terminal vertices.

<u>Closed Walk</u>:- A walk which begins and end at the same vertex.



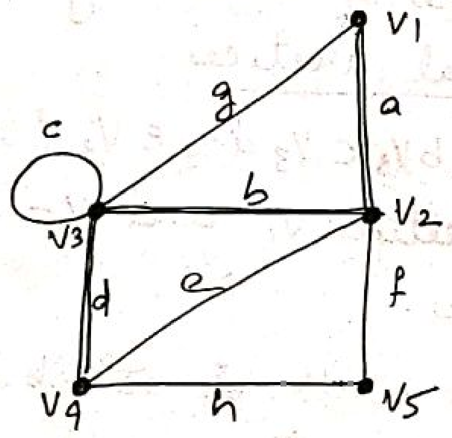
In the above graph the walk $V_1 a V_2 d V_4 e V_1$ is having V_1 vertice as begin and end vertices. So it is a closed walk

Open walk:- A walk that is not closed (i.e. the terminal vertices are distinct) is called an open walk.



In the above graph $v_1 a v_2 b v_3 c v_3 d v_4 e v_2 f v_5$ the sequence of walk having v_1 and v_5 as start and end vertex. So it is an open walk since v_1 and v_5 are distinct vertex.

An open walk in which no vertex appears more than once is called a path



In the above graph the walk $v_1 a v_2 b v_3 d v_4$ is a path because in that walk no vertices are appearing more than once.

But the walk $v_1 a v_2 b v_3 c v_3 d v_4 e v_2 f v_5$ is not a path because vertice v_3 is repeating on that path.

The number of edges in a path is called the <u>length of a path</u>

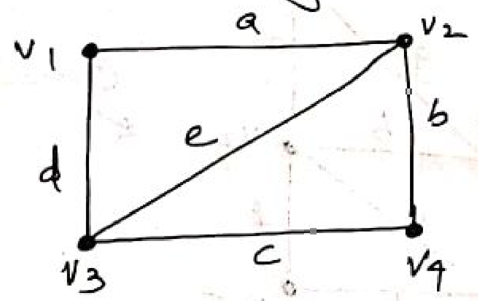
<u>Note:-</u>

Self-loop can be included in a walk but not in a path.

The terminal vertices of a path are of degree one, and the rest of the vertices are of degree two. This degree is counted only with respect to the edges included in the path and not the entire graph.

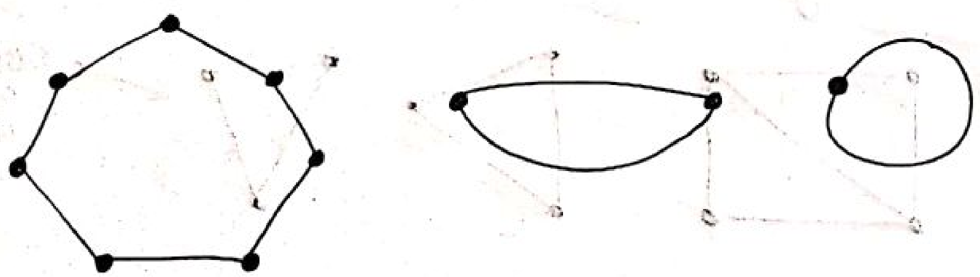
<u>Circuit:-</u>

A closed walk in which no vertex appears more than once is called a circuit. A circuit is closed, non intersecting walk.



In the above graph $V_1 a V_2 e V_3 d V_1$ is a circuit.

Different Types of circuits are given below.



Every vertex in a circuit is of degree two, If the ~~Subgraph~~ circuit is a subgraph of another graph, one must count degrees contributed by the edges in the circuit only.

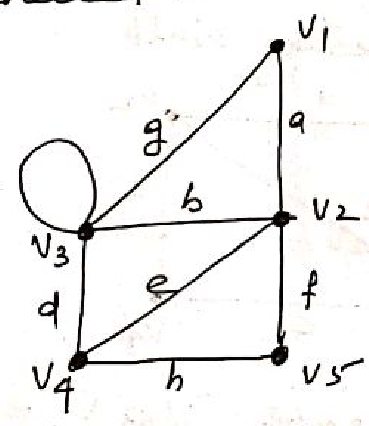
A circuit is also called a cycle

Connected Graphs

A graph is connected, If any vertex is reached from any other vertex by traveling along the edges.

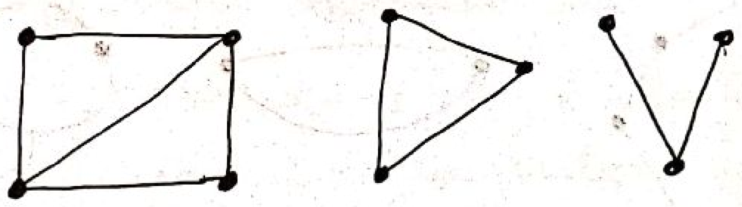
A graph G is said to be connected if there is atleast one path between every pair of vertices in G. otherwise G is disconnected.

Note:- A null graph of more than one vertex is disconnected.



Connected graph

The above graph is a connected graph because all vertices are connected atleast with one edges in the graph.



Disconnected graph

The disconnected graph consists of two or more connected graphs. Each of the connected subgraphs is called a component.

the above two graphs are two disconnected graph with two components.

Theorem - II

A graph G is disconnected if and only if its vertex set V can be partitioned into two nonempty, disjoint subsets V_1 and V_2 such that there exists no edge in G. whose one end vertex is in subset V_1 and the other in subset V_2.

Proof:-

Suppose that such a partitioning exists. consider two arbitrary vertices a and b of G, such that $a \in V_1$ and $b \in V_2$. No path can exist between vertices a and b; otherwise there would be at least one edge whose one end vertex would be in V_1 and the other in V_2. Hence if a partition exists, G is not connected.

Conversely let G be a disconnected graph. Consider a vertex 'a' in G. Let V_1 be the set of all vertices that are joined by paths to 'a'. Since G is disconnected, V_1 does not include all vertices of G. The remaining vertices will form (non-empty) set V_2. No vertex in V_1 is joined to any in V_2 by an edge. Hence the partition.

Theorem - III

If a graph (connected or disconnected) has exactly two vertices of odd degree, there must be a path joining these two vertices

Proof:-
Let G be a graph with all even vertices except vertices v_1 and v_2 which are odd. From theorem-1 which holds for every graph and therefore for every component of a disconnected graph, no graph can have an odd number of odd vertices. Therefore in graph G, v_1 and v_2 must belong to the same component, and hence must have a path between them.

Theorem - IV

A simple graph (i.e. a graph without parallel edges or self loops) with n vertices and k components can have at most $(n-k)(n-k+1)/2$ edges

Proof:-

Let the no. of vertices in each of the k components of a graph G be $n_1, n_2, \ldots n_k$

$$\begin{array}{cccccc} 1 & 2 & 3 & \cdots & k \\ | & | & | & \cdots & | \\ n_1 & n_2 & n_3 & & +n_k \end{array}$$

Thus we have

$$n_1 + n_2 + \cdots + n_k = n$$

where $n_i \geq 1$

$$\sum_{i=1}^{k} n_i = n$$

$$\sum_{i=1}^{k} n_i - 1 = n - k$$

The proof of the theorem depends on an algebraic inequality.

$$\sum_{i=1}^{k} n_i^2 \leq n^2 - (k-1)(2n-k)$$

Now the maximum number of edges in the i^{th} component of G is $\frac{1}{2} n_i(n_i - 1)$. Therefore the maximum number of edges in G is

$$\frac{1}{2} \sum_{i=1}^{k} n_i(n_i - 1) = \frac{1}{2} \sum_{i=1}^{k} (n_i^2) - \frac{n}{2}$$

$$= \frac{1}{2}\left[n^2 - (k-1)(2n-k)\right] - \frac{n}{2}$$

$$= \frac{1}{2}\left[n^2 - (2nk - k^2 - 2n + k\right] - \frac{n}{2}$$

$$= \frac{1}{2}\left[n^2 - 2nk + k^2 + 2n - k - n\right]$$

$$= \frac{1}{2}\left[(n-k)^2 + (n-k)\right]$$

$$= \frac{(n-k)}{2}\left[(n-k) + 1\right]$$

Hence proved.

problems.

1. A simple graph G has 24 edges and degree of each vertex is 4. Find the number of vertices.

Solution :-

Number of edges = 24

Degree of each vertex = 4

Let the number of vertices in the graph = n.

Using theorem - I

Sum of degree of all vertices = 2 × Number of edges

Substituting the values we get.

$n \times 4 = 2 \times 24$

$n = 2 \times 6$

$\therefore n = 12$

Thus the no: of vertices in the graph = 12

2. A graph contain 21 edges, 3 vertices of degree 4 and all other vertices of degree 2. Find total number of vertices

Solution

Given,

No: of edges = 21

No: of 4 degree vertices = 3

All other vertices are of degree 2

Let no: of vertices in the graph = n

Using theorem - I

Sum of degree of all vertices = 2 × No: of edges.

Substituting the values, we get.

$3 * 4 + (n-3) \times 2 = 2 \times 21$

$12 + 2n - 6 = 42$

$2n = 42 - 6$

$2n = 36$

$\therefore n = 18$

Total no: of vertices in the graph = 18

3. A simple graph contains 35 edges. four vertices of degree 5, five vertices of degree 4 and four vertices of degree 3. Find the number of vertices with degree 2.

Solution

No: of edges = 35

No: of 5 degree vertices = 4

No: of 4 degree vertices = 5

No: of 3 degree vertices = 4

Let the number of 2 degree vertices in the graph = n

Using theorem - I

Sum of degree of all vertices = 2 × No: of edges

Substituting this values, we get.

$4\times5+5\times4+4\times3+n\times2 = 2\times35$

$20+20+12+2n = 70$

$52+2n=70$

$2n = 70-52$

$2n=18$

$\therefore n=9$

$\therefore$ No: of 2 degree vertices in the graph $=9$

4. A graph has 24 edges and degree of each vertex is k, then what is the possible no: of vertices 20 or 15 or 10 or 8

solution :-

No: of edges $=24$

Degree of each vertex $=k$.

Let number of vertices in the graph $=n$.

Using theorem-I we have.

Sum of degree of all vertices $=$ 2 x Number of edges

Substituting this value we get.

$n\times k = 2\times 24$

$k = 48/n$

<u>Note</u>:- the degree of any vertex is a whole number

Check the option one by one.

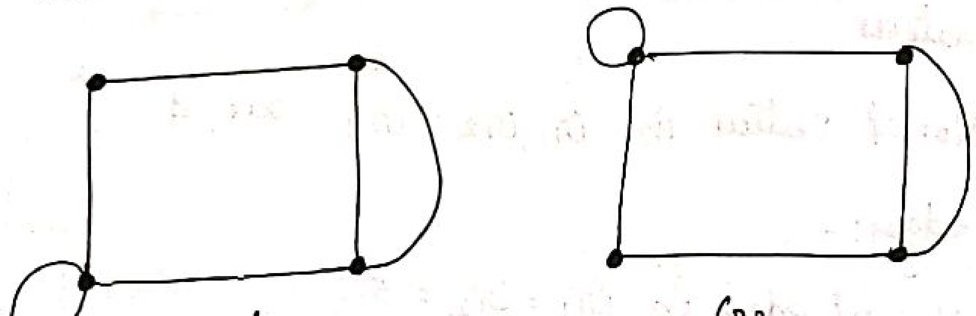
for $n=20$ $k=48/20 = 2.4$ not possible

for $n=15$ $k=48/15 = 3.2$ not possible.

for $n=10$, $k=4.8$ not possible

for $n=8$, $k=6$ this is possible.

5. Check whether the following two graphs are isomorphic or not.



G_1 G_2.

No: of vertices in $G_1 = 4$ = No: of vertices in G_2

No: of edges in $G_1 = 6$ = No: of vertices in G_2

Degree of $G_1 = 2, 3, 3, 4$.

Degree of $G_2 = 2, 3, 3, 4$

G_1 forms a 4 length cycle using vertex degree as 2, 3, 3, 4

G_2 also forms a 4 length cycle using vertex degree as 2, 3 3, 4.

G_1 form a 2 length cycle with vertex degree 3, 3

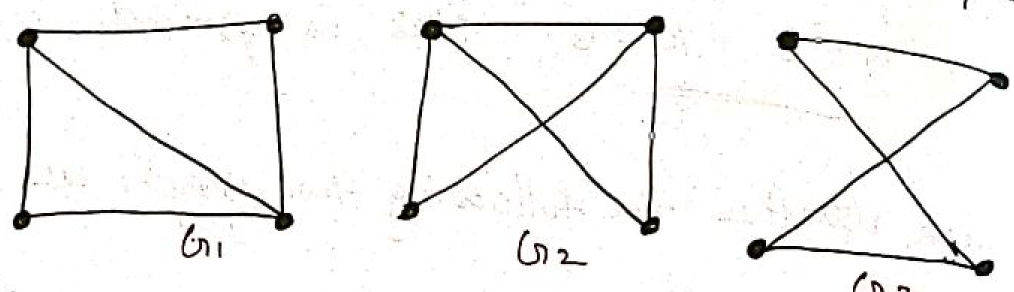
G_2 also forms a 2 length cycle with vertex degree 3, 3.

G_1 and G_2 forms a self loop at 4 degree vertices

so all the conditions of isomorphism are satisfied

Thus the given graph is isomorphism.

2. which of the following graph are isomorphic



vertices.

No: of vertices in G1, G2, G3 are 4

edges:-

No: of edges in G1 = G2 = 5

No: of edges in G3 = 4

∴ graph G3 violates the edges conditions so it is not an isomorphic graph.

Continuing with G1 and G2.

Degree Sequence :-

Degree Sequence of G1 = 2, 3, 2, 3

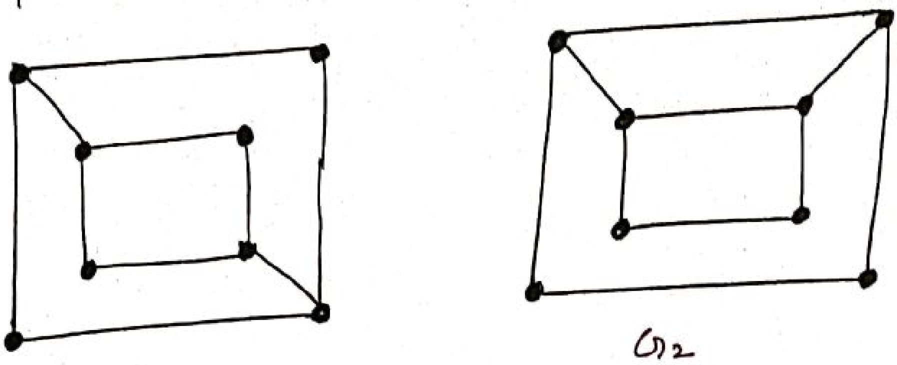
Degree sequence of G2 = 2, 3, 3, 2

Both G1 and G2 have same degree sequence

Graph G1 and G2 contains two cycles of length 3 which is formed by the vertices having degree {2, 3, 3}.

Since all the conditions satisfies by the graph G1 and G2. Both are isomorphic graphs.

3. Check whether the following two graphs are isomorphic.



G_1 G_2

Vertices :-

No: of vertices in G_1 = 8 = No: of vertices in G_2

Edges:-

No: of edges in G_1 = 10 = No: of edges in G_2

Degree Sequence :-

Degree sequence of graph $G_1 = \{2, 2, 2, 2, 3, 3, 3, 3\}$

Degree Sequence of graph $G_2 = \{2, 2, 2, 2, 3, 3, 3, 3\}$

In graph G_2 3 degree vertices forms 4 lengths cycle.

In graph G_2 3 degree vertices do not form a 4 length cycle.

∴ The two graph G_1 and G_2 violates the condition of isomorphism. Hence the given two graphs are not isomorphic graph.

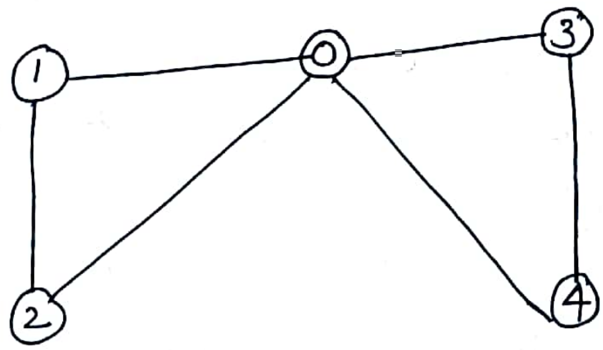
Euler Graph

Graph Theory was born in 1736 with Euler's famous paper in which he solved the konigsberg bridge problem. In the same paper Euler posed a problem In what type of graph 'G' is it possible to find a closed walk running through every edge of G exactly once.?

If some closed walk in a graph consists of all the edges of the graph, then the walk is called an Euler line and the graph is an Euler graph.

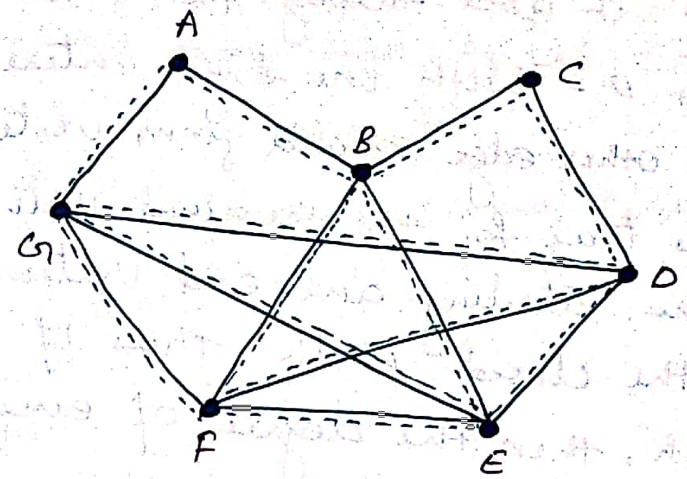
A walk in Euler graph is always connected. Since the Euler line contains all the edges of the graph, an Euler graph is always connected except for any isolated vertices the graph may have.

Example1:-



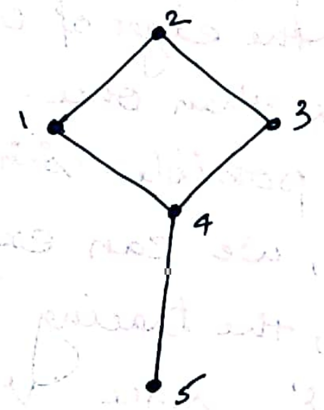
In the above graph all vertices possess even degree and it consist of an Eulerian cycle 210340 It is a closed walk since start and end vertices are same.

Example 2:



In the above graph all vertices possess even degree and the closed walk is "A B C D G E D F B E F G A"

Example 3:



In the above graph we cannot perform a closed walk which includes all the edges. Since the graph possess 2 odd degree vertices (vertex 4 & 5)

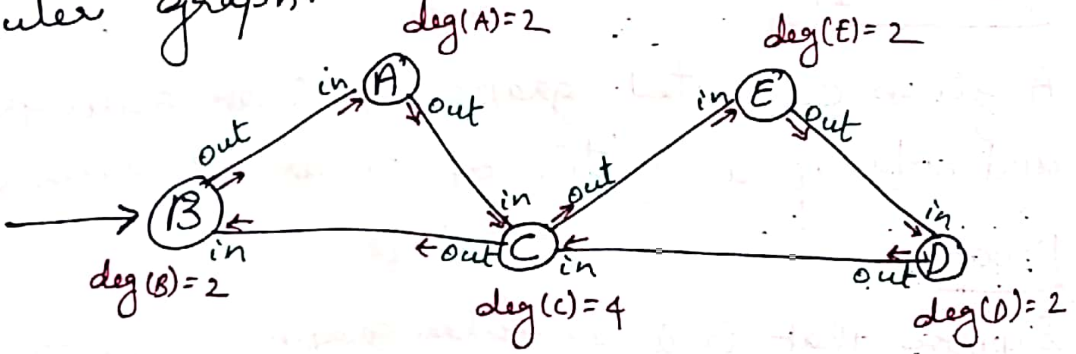
Theorem 1:

A given connected graph G is an Euler graph if and only if all vertices of G are of even degree.

Proof:-

Suppose that G is an Euler graph. Therefore it contains an Euler line. when every time the walk meets a vertex 'v' it goes through two "new" edges incident on v - with one edge entered on vertex 'v' and the other edge exited from vertex 'v'. This condition is true for the terminal vertices also because the starting and end vertices are same in the closed walk. Thus If G is a Euler graph, then the degree of every vertex is even.

To prove sufficiency of condition, assume that all vertices of G are of even degree. A walk is constructed starting from an arbitrary vertex 'v' and going through the edges of G such that no edge is traced more than once. We continue tracing as far as possible. Since every vertex is of even degree, we can exit from every vertex we enter, the tracing cannot stop at any vertex but 'v'. Since 'v' is also of even degree, we shall reach to the vertex 'v' when the tracing comes to an end. If this closed walk includes all the edges of G, then G is an Euler graph.

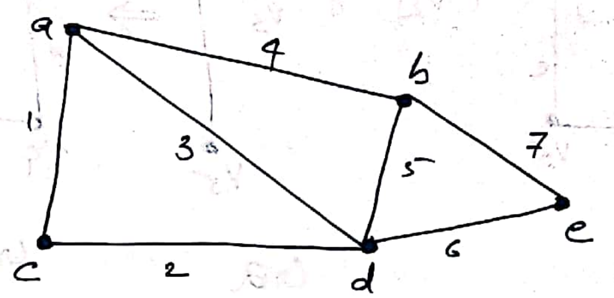


Unicursal graph.

An open walk that includes all edges of a graph without retracing any edge is called a unicursal line or an open Euler line.

A graph that has a unicursal line is called as Unicursal graph

Example:-



The above is the unicursal graph with a walk a 1 c 2 d 3 a 4 b 5 d 6 e 7 b which includes all edges.

Note:-

By adding an edge in between the initial and final vertices of a unicursal line we will get an Euler line.

Thus a connected graph is Unicursal iff it has exactly two vertices of odd degree.

Operations on Graph

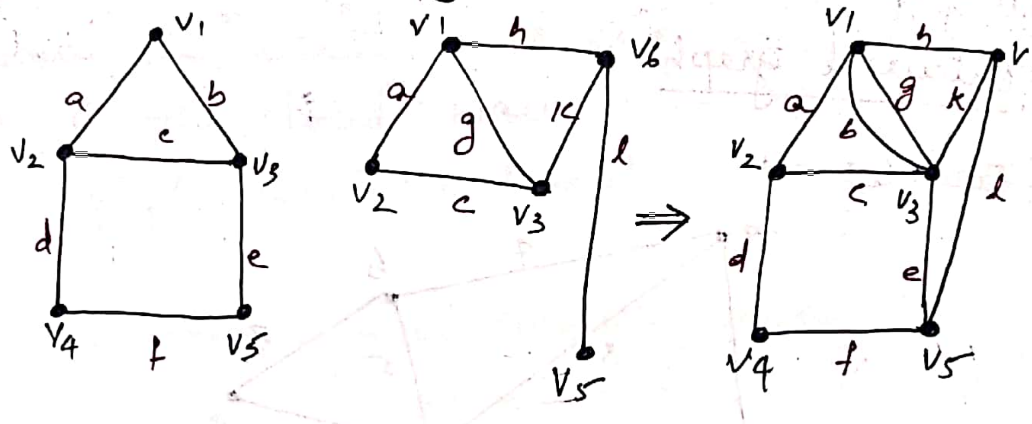
Following are the different operations performed on graph

→ Union
→ Intersection
→ Fusion
→ Ring Sum
→ Decomposition
→ Deletion

Scanned with CamScanner

Union:-

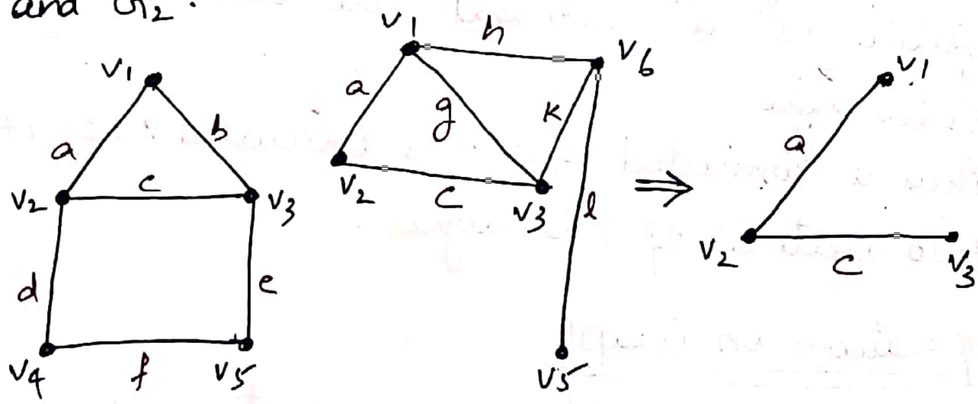
The Union of two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ is another graph G_3 which can be written as $G_3 = G_1 \cup G_2$ whose vertex set $V_3 = V_1 \cup V_2$ and the edge set $E_3 = E_1 \cup E_2$



G_1 $\quad$ G_2 $\quad$ $G_1 \cup G_2$

Intersection:-

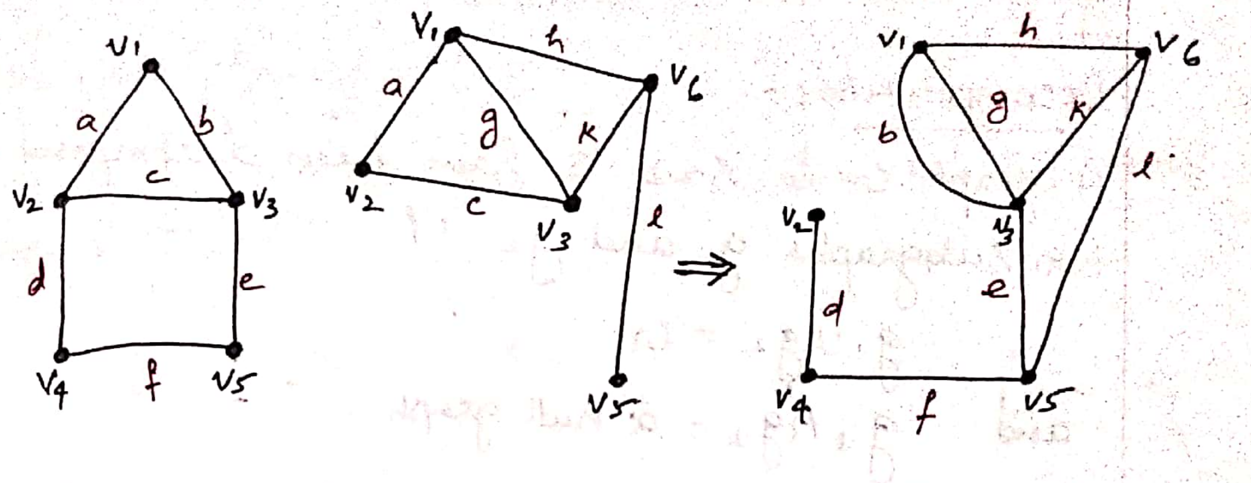
The intersection of two graphs G_1 and G_2 can be written as $G_1 \cap G_2$ which is a graph consisting only of those vertices and edges that are both in G_1 and G_2.



G_1 $\quad$ G_2 $\quad$ $G_1 \cap G_2$

Ring Sum:-

The ring sum of two graphs G_1 and G_2 can b written as $G_1 \oplus G_2$ is a graph consisting of the vertex set $V_1 \cup V_2$ and of edges that are either in G_1 or G_2, but not in both.



G_1 $\qquad$ G_2 $\qquad$ $G_1 \oplus G_2$

The above three operations are commutative

i.e

$$G_1 \cup G_2 = G_2 \cup G_1$$

$$G_1 \oplus G_2 = G_2 \oplus G_1$$

$$G_1 \cap G_2 = G_2 \cap G_1$$

If G_1 and G_2 are edge disjoint then $G_1 \cap G_2$ is a null graph, and $G_1 \oplus G_2 = G_1 \cup G_2$. If G_1 and G_2 are vertex disjoint then $G_1 \cap G_2$ is empty.

For any graph G,

$$G \cup G = G \cap G = G$$

$$G \oplus G = \text{a null graph.}$$

If g is a subgraph of G then $G \oplus g$ is that subgraph of G which remains after all the edges in g have been removed from G. Therefore $G \oplus g$

is written as $G-g$, whenever $g \subseteq G$. Therefore $G \oplus g = G - g$ is often called the complement of g in G.

Decomposition:-

A graph G is said to have been decomposed into two subgraphs g_1 and g_2 if

$$g_1 \cup g_2 = G$$

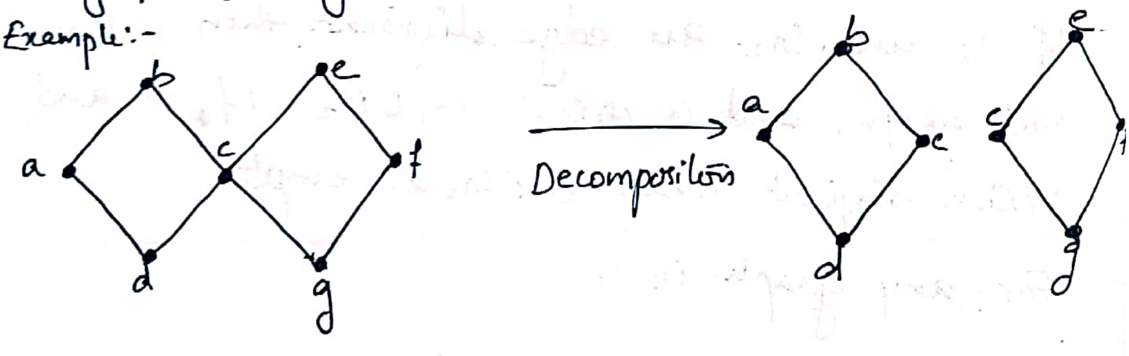
and $$g_1 \cap g_2 = \text{a null graph}.$$

Every edge of G occurs either in g_1 or in g_2, but not in both. Some of the vertices may occur in both g_1 and g_2.

In decomposition, isolated vertex are disregarded.

A graph containing m edges $\{e_1, e_2, \ldots e_m\}$ can be decomposed in $2^{m-1}-1$ different ways into pairs of subgraph g_1, g_2

Example:-



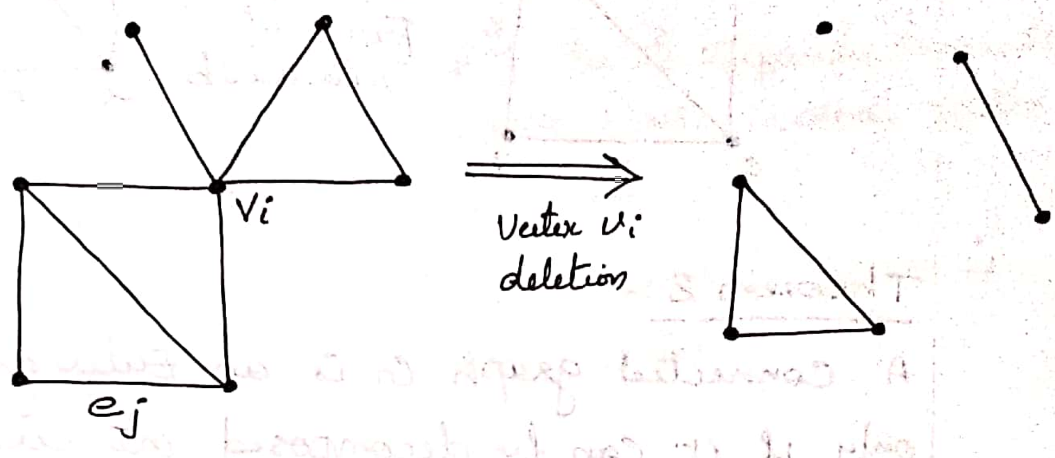
Deletion:-

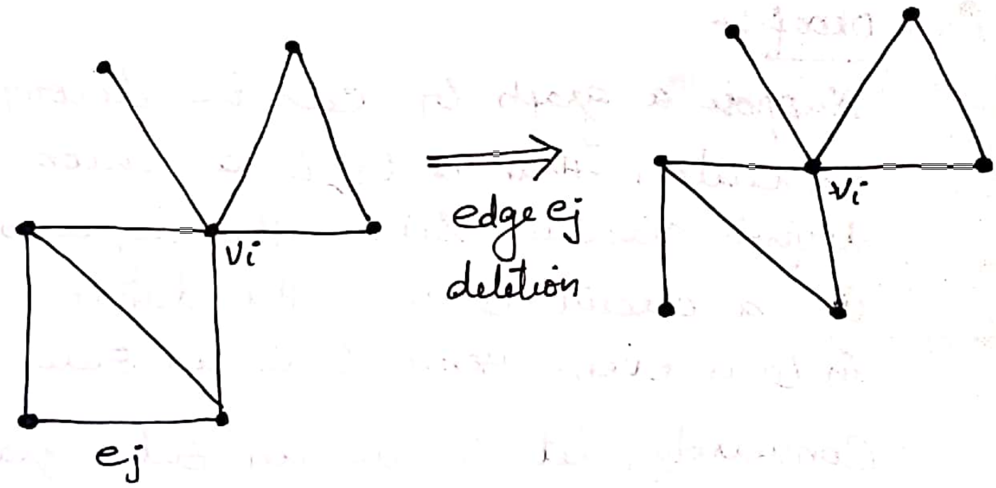
If v_i is a vertex in graph G, then $G - v_i$ denotes a subgraph of G obtained by deleting (removing) v_i from G. Deletion of a vertex always implies the deletion of all edges incident on that vertex.

If e_j is an edge in G, then $G - e_j$ is a subgraph of G obtained by deleting e_j from G. Deletion of an edge does not imply deletion of its end vertices.

Therefore $G - e_j = G \oplus e_j$.

The following graph shows the vertex deletion and edge deletion.



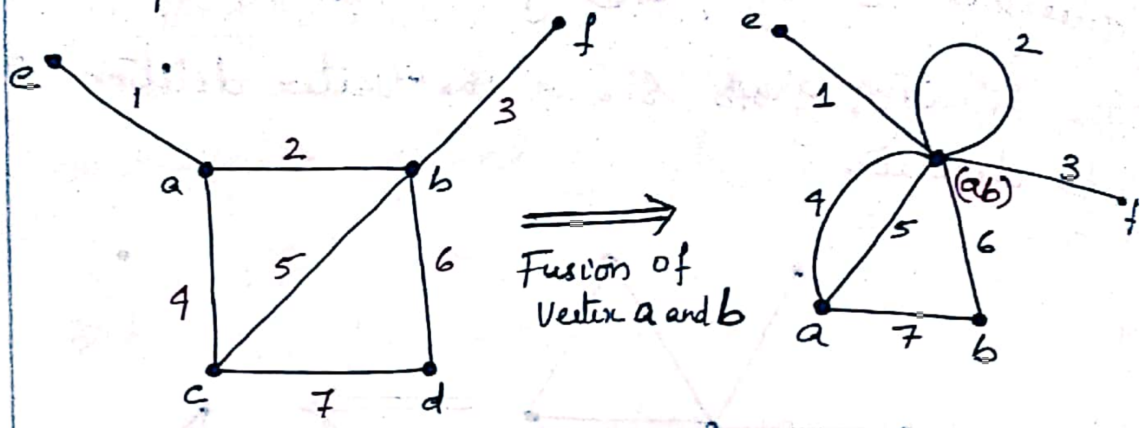


Fusion:-

A pair of vertices a, b in a graph are said to be fused (merged or identified) if the two vertices are replaced by a single new vertex such that every edge that was incident on either a or b or on both is incident on the new vertex.

Thus the fusion of two vertex does not alter the number of edges, but it reduces the number of vertices by one.

Example:-



Theorem 2:-

A connected graph G is an Euler graph if and only if it can be decomposed into circuits.

Proof:-

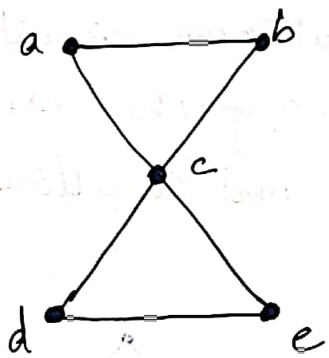
Suppose a graph G can be decomposed into circuits, that is G is a union of edge-disjoint circuits. Since the degree of every vertex in a circuit is two, the degree of every vertex in G is even. Hence G is an Euler graph.

Conversely, let G be an Euler graph. Consider a vertex v_1, there are atleast two edges incident at v_1. Let one of these edges be between v_1 and v_2. Since vertex v_2 is also of even degree, it must have at least another edge, say between v_2 and v_3. Proceeding in this fashion, we eventually arrive at a vertex that has previously been traversed thus forming a circuit Γ. Let us remove Γ from G. All vertices in the remaining graph must also be of even degree. From the remaining graph removes another circuit in exactly the same way as we removed Γ from G. Continue this process until no edges are left. Hence the theorem.

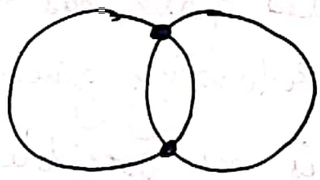
Arbitrarily Traceble graphs

An Eulerian graph G is said to be arbitrarily traceable from a vertex v if every walk with initial vertex v can be extended to an Euler line of G.

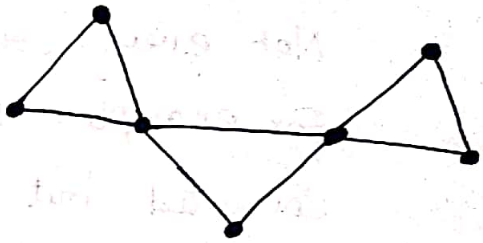
A graph is said to be arbitrarily traceable if it is arbitrarily traceable from every vertex.



Arbitrarily traceable graph from c



Arbitrarily traceable graph from all vertices

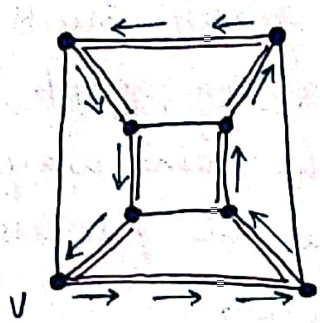


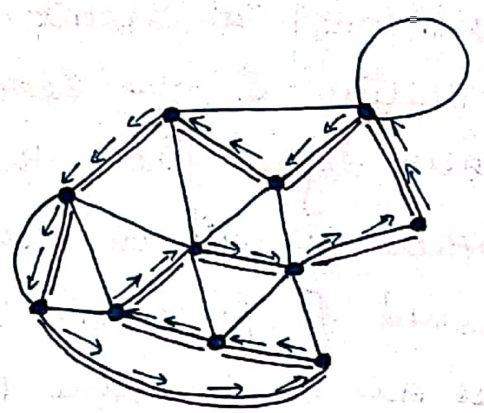
Euler graph, not arbitrarily traceable

Hamiltonian paths and circuits

Hamiltonian circuit in a connected graph is defined as a closed walk that traverses every vertex of G exactly once, except the starting vertex at which the walk also terminates.

Example :-

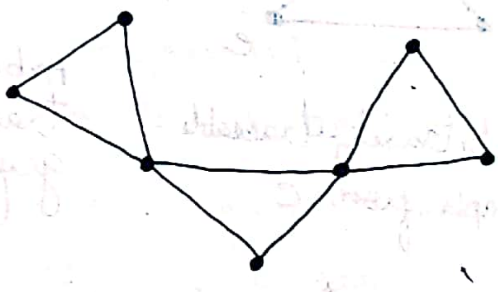




The above two graphs shows the hamiltonian circuit in the red line. The flow of graph is shown in green arrow lines.

A circuit in a connected graph G is said to be Hamiltonian if it includes every vertex of G. Hence a hamiltonian circuit in a graph of n vertices consists of exactly n edges.

Not every connected graph has Hamiltonian circuit. For example: The following given two graphs are connected but it is not possible to find hamiltonian circuit on this graph.



Hamiltonian path :-

If we remove any one edge from a Hamiltonian circuit then hamiltonian path is formed. A Hamiltonian path in a Graph G traverses every vertex of G.

Since a hamiltonian path is a subgraph of a hamiltonian circuit.

Every graph that has a hamiltonian circuit also has a Hamiltonian path. However many graphs with Hamiltonian paths that have no hamiltonian circuits. The length of a Hamiltonian path is a connected graph of n vertices is $n-1$

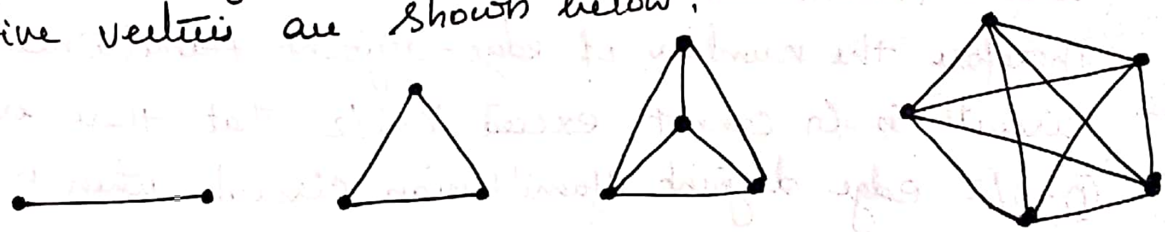
NB:-

Hamiltonian circuit cannot include a self-loop or a set of parallel edges.

What type of graph is guaranteed to have Hamiltonian circuits?

Complete graph of three or more vertices constitute one such class.

Complete graph:- It is a simple graph in which there exists an edge between every pair of vertices is called a complete graph. Complete graphs of two, three, four and five vertices are shown below.



Complete graph of two, three, four & five vertices

A complete graph is sometimes referred as a universal graph or a clique. Since every vertex is joined with every other vertex through one edge, the degree of every

Vertex is n-1 in a complete Graph G of n Vertices. The total number of edges in G is $n(n-1)/2$.

Let the Vertices in a complete graph be numbered as $v_1, v_2, v_3, \ldots v_n$. Since an edge exists between any two Vertices, we can start from v_1 and traverse to v_2 and v_3 and so on to v_n and finally v_n to v_1. This is a Hamiltonian circuit.

NB:-

A graph may contain more than one Hamiltonian circuit.

Theorem: 3

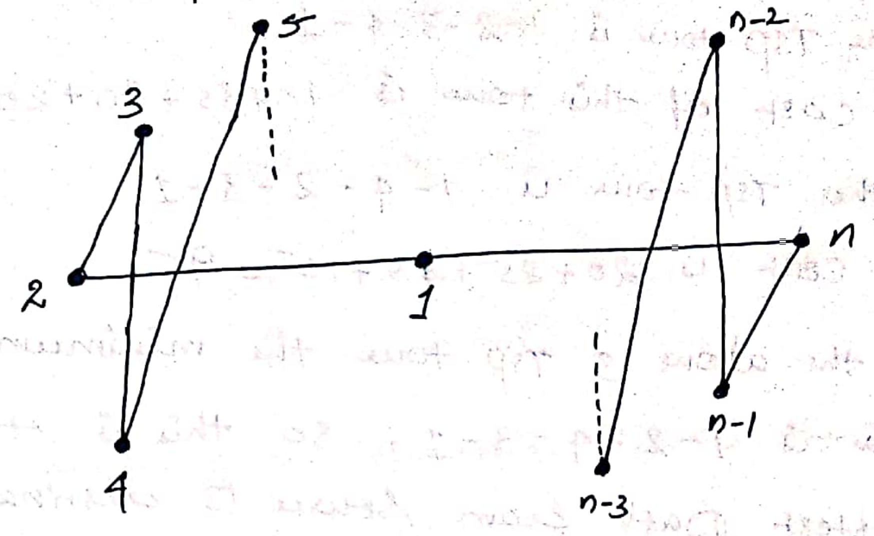
In a complete graph with n vertices there are $(n-1)/2$ edge disjoint Hamiltonian circuits, If n is an odd number ≥ 3.

Proof:-

A complete graph G of n Vertices has $n(n-1)/2$ edges and a Hamiltonian circuit in G consists of n edges. Therefore the number of edge-disjoint Hamiltonian circuits in G cannot exceed $(n-1)/2$. That there are $(n-1)/2$ edge disjoint Hamiltonian circuits when n is odd.

The Subgraph shown in the following figure shows a Hamiltonian circuit. Keeping the vertices fixed on a circle, rotate the polygonal pattern clockwise by $360/(n-1)$, $2*360(n-1)$, $3*360(n-1)$ ..... $(n-3)/2 * 360/n$ degrees.

Each rotation produces a Hamiltonian circuit that has no edge in common with any of the previous ones. Thus we have $(n-3)/2$ new Hamiltonian circuits, all edges disjoint from the one in the ~~above~~ below figure and also edge disjoint among themselves. Hence the theorem proved.



Hamiltonian circuit, n is odd.

Travelling Salesman problem

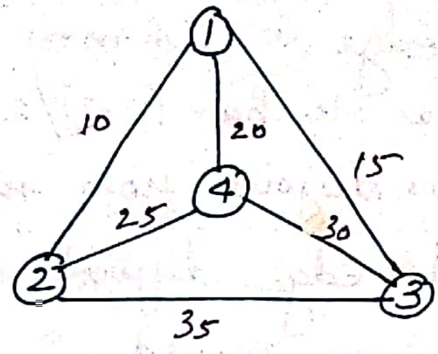
Traveling salesman problem is stated as A salesman is required to visit a number of cities during a trip. Given a set of cities and distance between every pair of cities, the problem is to find the shortest possible route that visits every city exactly once and return to the starting point.

The cities can be represented as vertices and the roads between them as edges. We will get a graph. In this graph every edge e_i is associated with a weight w_i

Example.

TSP tour in the graph is 1-2-4-3-1

The Cost of the tour is 10+25+30+15 = 80



Another TSP tour is 1-2-3-4-1.
The cost of this tour is 10+35+30+20 = 95

Another TSP tour is 1-4-2-3-1
The cost is 20+25+35+15 = 95

Of the above 3 TSP tour the minimum cost Tour is 1-2-4-3-1. So this is the shortest path from source to destination city in the travel.

Directed Graph

A directed graph (or digraph) is consists of a set of vertices $V = \{v_1, v_2, \ldots\}$, a set of edges $E = \{e_1, e_2, \ldots\}$ and a mapping ψ that maps every edge onto some ordered pair of vertices (v_i, v_j).

A digraph is also known as directed network or oriented graph.

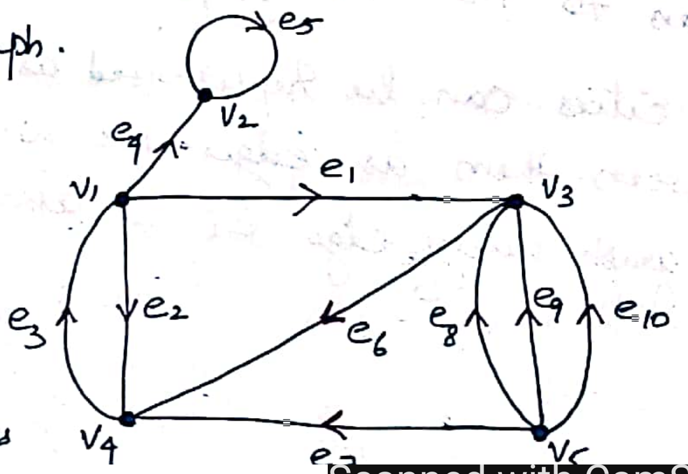
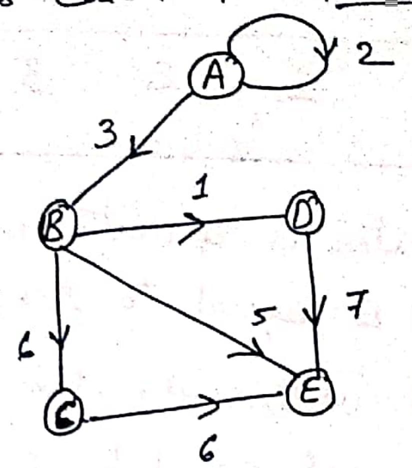


fig: Directed graph with 5 vertices & 10 edges

In a digraph an edge is not only incident on a vertex, but is also incident out of a vertex. The vertex v_i which edge e_k is incident out of is called the initial vertex of e_k. The vertex v_j which e_k is incident into is called the terminal vertex of e_k.

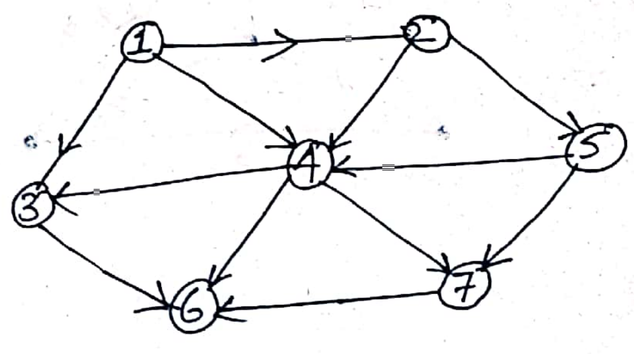


In the above graph the vertex E is the terminal vertices of edge 6, 5 and 7. The vertex B is the initial vertices of edge 1, 5 & 6.

An edge for which the initial and terminal vertices are the same forms a self loop. In the above graph the edge '2' is having same vertex A as the initial and terminal vertices and forms the loop.

The number of edges incident out of a vertex v_i is called the out-degree of v_i and is represented as $d^+(v_i)$.
The number of edges incident into v_i is called the in-degree of v_i and is represented as $d^-(v_i)$.

for example.



In the above graph. The in degree and out degree is as given below:

Vertex	v_1	v_2	v_3	v_4	v_5	v_6	v_7
Out-degree	3	2	1	3	2	0	1
In-degree	0	1	2	3	1	3	2

An isolated vertex is a vertex in which in-degree and out-degree is equal to zero.

For a pendant vertex the sum of the in-degree and out-degree is equal to 1 ie

$$d^+(v_i) + d^-(v_i) = 1 \quad \text{if } v_i \text{ is pendant vertex}$$

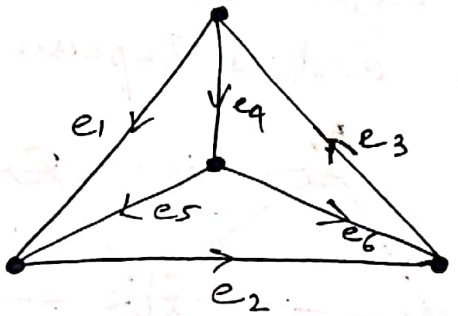
Types of Digraph

Variety of digraphs are due to the choice of assigning a direction to each edge.

Simple digraph:-

A graph that has no self-loop or parallel edges is called a simple digraph.

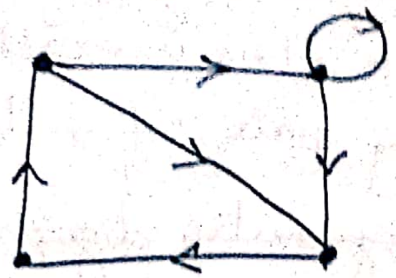
Example:-



Asymmetric digraph:-

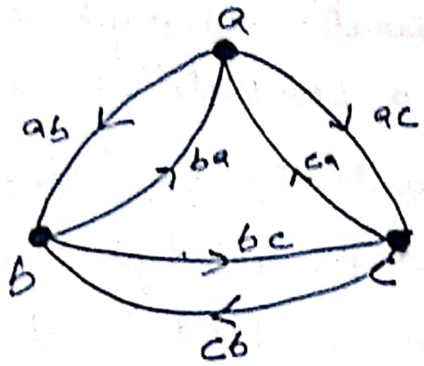
Digraphs that have at most one directed edge between a pair of vertices, but are allowed to have self loops, are called asymmetric or antisymmetric.

Example:-



Symmetric digraph:-

Digraphs in which for every edge (a,b) there is an edge (b,a)



A digraph that is both simple and symmetric is called a simple symmetric digraph.

A digraph that is both simple and asymmetric will be a simple asymmetric.

Complete digraphs:-

A complete undirected graph was defined as a simple graph in which every vertex is joined to every other vertex exactly by one edge. For digraphs we have two types of complete graphs.

A complete symmetric digraph is a simple digraph in which there is exactly one edge directed from every vertex to every other vertex.

A complete asymmetric digraph is an asymmetric digraph in which there is exactly one edge between every pair of vertices.

A complete asymmetric digraph of n vertices contains $n(n-1)/2$ edges, but a complete symmetric digraph of n vertices contains $n(n-1)$ edges.

A complete asymmetric digraph is also called a tournament or a complete ~~tournament~~.

A digraph is said to be balanced if for every vertex v_i the in-degree equals the out-degree that is $d^+(v_i) = d^-(v_i)$

A balanced digraph is said to be regular if every vertex has the same in-degree and out-degree as every other vertex.

Dirac's Theorem

A graph with $n \geq 3$ vertices, and every vertex having degree $> n/2$ is hamiltonian

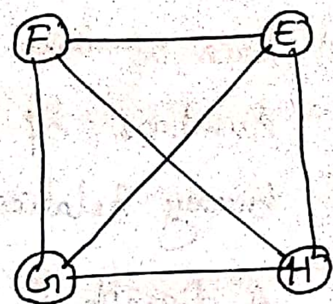
Proof:-

Graph is connected. Consider a path $u_1, u_2, \ldots u_k$ which is the longest path.

Hamiltonian path covers all vertices i.e if we consid the longest path then $k = n$.

Consider the graph G with 4 vertices so $n = 4$

Deg(F) = 3
Deg(E) = 3
Deg(G) = 3
Deg(H) = 3



Degree of all vertices is greater than or equal to the half of the no: of vertices in Graph G.
In this graph no: of vertices $n = 4$. Half of no: of vertices $n/2 = 2$. So degrees of all vertices is greater than 2. A hamiltonian cycle can be formed on the graph G which is FEHGF.

Digraphs and binary relation

The theory of graphs and the binary relations are closely related.
In a set of objects X, where

$$X = \{x_1, x_2, \dots\}$$

a binary relation R between pair (x_i, x_j) may exist in which case, we write $x_i R x_j$ and say that x_i has relation R to x_j.
Relation R may for instance be "is parallel to", "is orthogonal to", "is congruent to" in geometry. It may be "is greater than", "is a factor of", "is equal to", and so on in case when X consists of number.

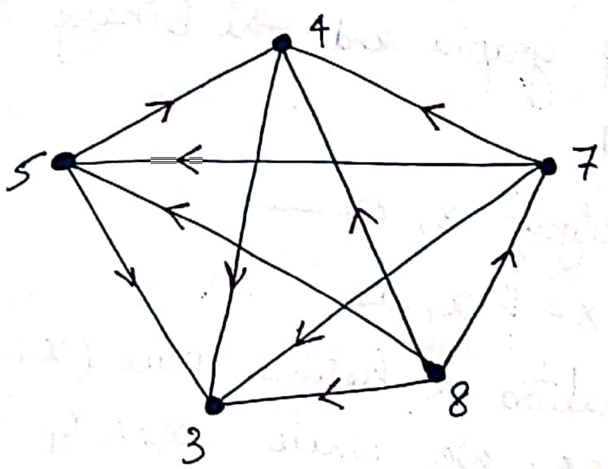
If the set X is composed of people, the relation R may be "is son of", "is spouse of", "is friend of" and so on.

Each of these relationship is defined only on pairs of numbers of the set, and this is why the name binary relation.

A digraph is the natural way of representing a binary relation on a set X. Each $x_i \in X$ is represented by a vertex x_i. If x_i has the specified relation R to x_j, a directed edge is drawn from vertex x_i to x_j for every pair (x_i, x_j)

Example:-

The following graph represents the relation "is greater than" on the set of five numbers {3, 4, 7, 5, 8}



Every binary relation on a finite set can be represented by a digraph without parallel edges.

or Every digraph without parallel edges defines a binary relation on the set of its vertices

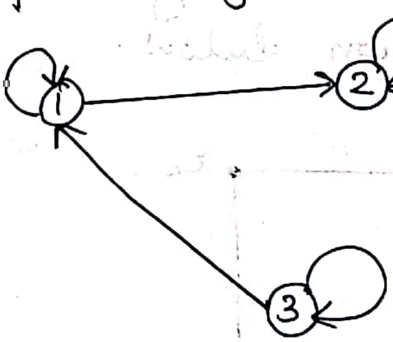
Reflexive Relation:-

For some relation R it may happen that every element is in relation R to itself. For example, a number is always equal to itself, or a line is always parallel to itself. Such a relation R on set X that satisfies

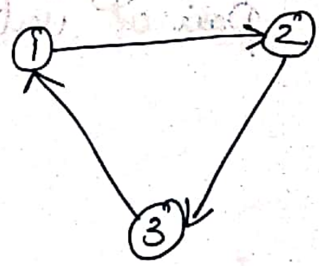
$$x_i R x_i$$

for every $x_i \in X$ is called a reflexive relation. The digraph of a reflexive relation will have a self loop at every vertex. Such a digraph representing a reflexive binary relation on its vertex set may be called a reflexive digraph.

A digraph in which no vertex has a self loop is called an irreflexive digraph.



Reflexive Digraph



irreflexive digraph

Symmetric Relation:-

For some relation R it may happen that for all x_i and x_j if

$x_i R x_j$ holds then $x_j R x_i$ also holds.

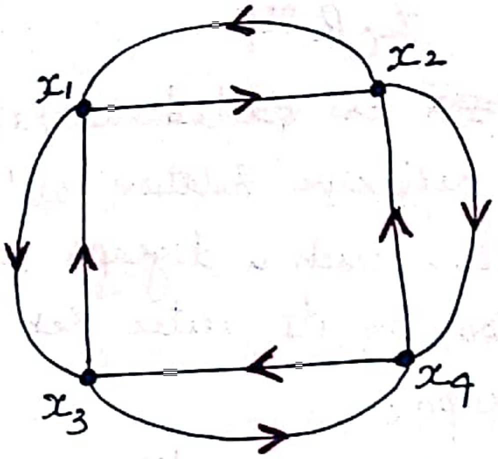
Such a relation is called a symmetric relation

eg:- "Is spouse of" is a symmetric but irreflexive relation.

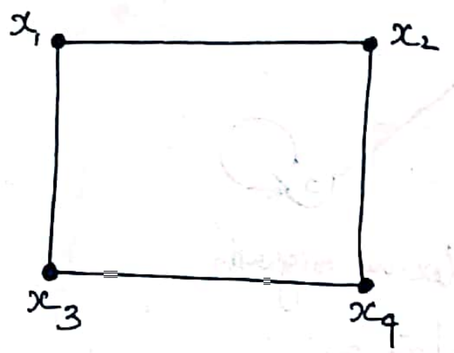
"Is equal to" is both Symmetric and Reflexive

The digraph of a Symmetric Relation is a Symmetric digraph because for every directed edge from vertex x_i to x_j there is a directed edge from x_j to x_i

Example.



The above graph can also be represented by drawing just one undirected edge between every pair of vertices as shown below.



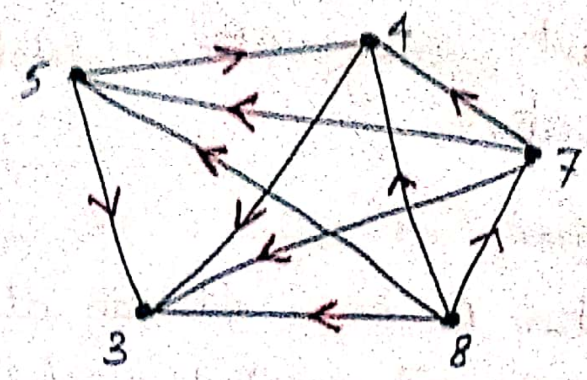
Transitive Relation

A relation R is said to be transitive if for any three elements x_i, x_j and x_k in the set

$$x_i R x_j \text{ and } x_j R x_k$$

always imply $x_i R x_k$

Example: The binary relation "is greater than" is transitive relation. ie $x_i > x_j$ & $x_j > x_k$ implies $x_i > x_k$



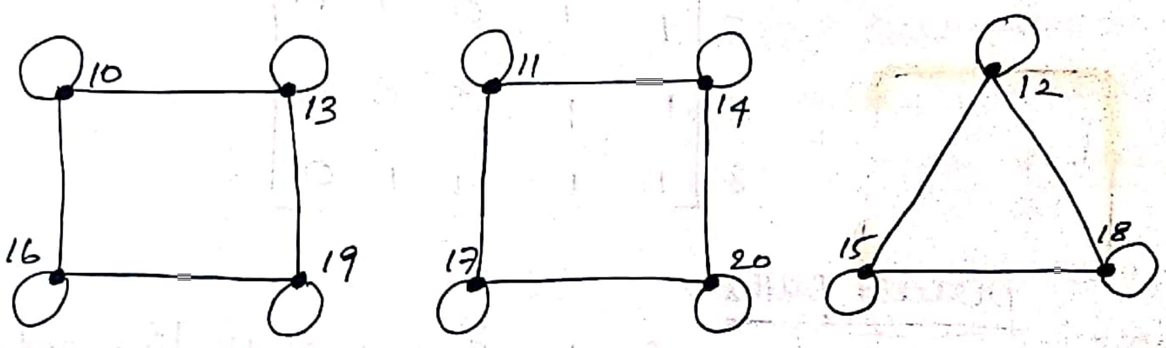
A digraph representing a transitive relation is called a transitive directed graph.

Equivalence Relation :-

A binary relation is called an equivalence relation if it is reflexive, symmetric and transitive.

Examples :- "is parallel to", "is equal to", "is congruent to", "is equal to modulo m", "is isomorphic to"

The graph representing an equivalence relation may be called an equivalence graph.



$\equiv 1 \pmod 3$ $\qquad$ $\equiv 2 \pmod 3$ $\qquad$ $\equiv 0 \pmod 3$

In the above graph vertex set of a graph is divided into three disjoint classes, each is a separate component. Each component is an undirected subgraph with a self-loop at each vertex. In each component every vertex is related to every other vertex.

Relation Matrices

A binary relation R on a set can also be represented by a matrix called a relation matrix. It is a (0,1), n by n matrix where n is the number of elements in the set.

The i, j th entry in the matrix is 1 if $x_i R x_j$ is true and is 0 otherwise.

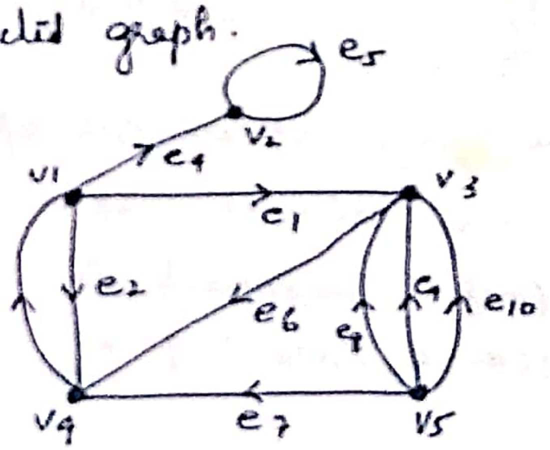
for eg:: the relation matrix of the relation " is greater than" on the set of integers {3, 4, 7, 5, 8} is

$$\begin{array}{c|c} & \begin{matrix} 3 & 4 & 7 & 5 & 8 \end{matrix} \\ \begin{matrix} 3 \\ 4 \\ 7 \\ 5 \\ 8 \end{matrix} & \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 \end{bmatrix} \end{array}$$

Directed paths

Directed path is the sequence of vertices and edges in the directed graph.

Example:-



The sequence of vertices and edges $V_5 e_8 V_3 e_6 V_4 e_3 V_1$ is a path directed from V_5 to V_1 whereas $V_5 e_7 V_4 e_6 V_3 e_1 V_1$ has no such consistent direction from V_5 to V_1.

A directed walk from a vertex V_i to V_j is an alternating sequence of vertices and edges, beginning with V_i and ending with V_j such that each edge is oriented from the vertex preceding it to the vertex following it.

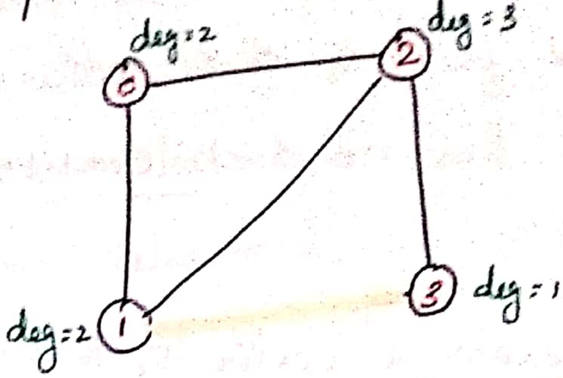
Fleurys' Algorithm

Fleurys Algorithm is used to find Eulerian path or circuit from the connected graphs. The steps to find Eulerian path from a graph is as follows.

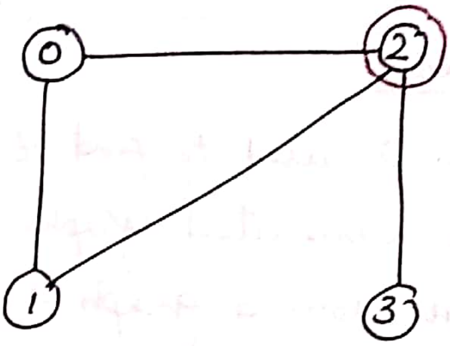
1. Make sure the graph has either 0 or 2 odd vertices
2. If there are 0 odd vertices, we can start from any vertex. If there are 2 odd vertices, then we can start from any one of these vertices
3. Follow edges one at a time. If we have a choice between the bridge and non bridge, always choose the non bridge
4. Stop when there is no edges remaining in the graph.

Scanned with CamScanner

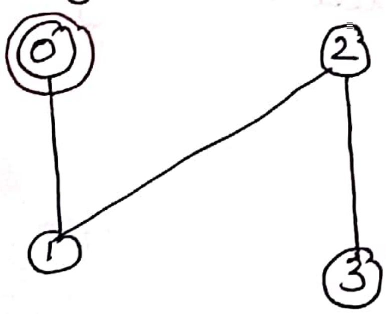
Example:-



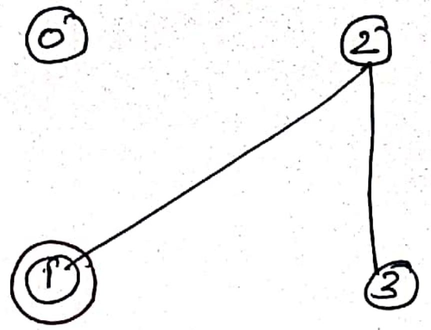
In the above graph there are two vertices with odd degree, "2" and "3". we can start from any one of these vertex.
we can start from vertex "2"



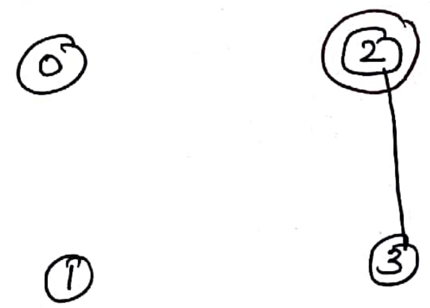
There are three edges going out from vertex "2"
we are not picking the edge 2-3 because that is a bridge from 2-3 edge we cannot further move on the graph since 3 is an end vertex.
we can pick any of the remaining edge
"2-1" or "2-0"
we can choose edge 2-0 and remove this edge from graph and move to vertex "0"



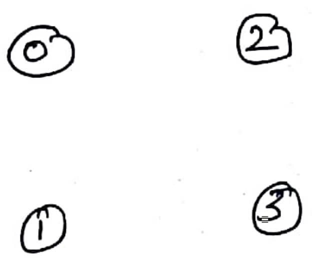
There is only one edge from vertex 0 it is 0-1 so we can choose and remove edge 0-1 and can reach to vertex "1"



There is only one edge from vertex '1" it is 1-2. so we can choose that edge and remove from the graph and move to vertex "2"



Again there is only one edge from vertex '2" it is 2-3. so we can pick that edge and remove from graph and reach to vertex "3".

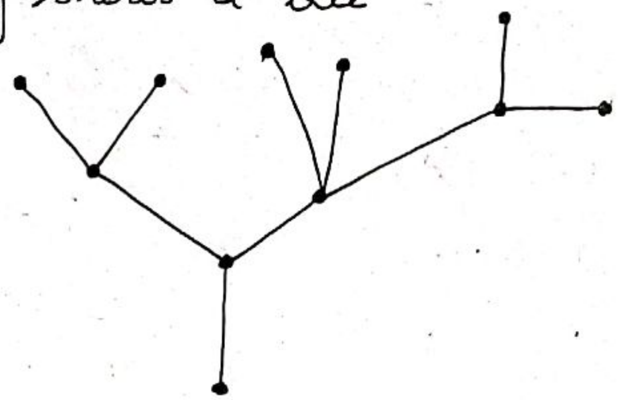


Now there is no more edges left in the graph. so we can stop our algorithms and the Euler tour we get is "2-0. 0-1 1-2 2-3"

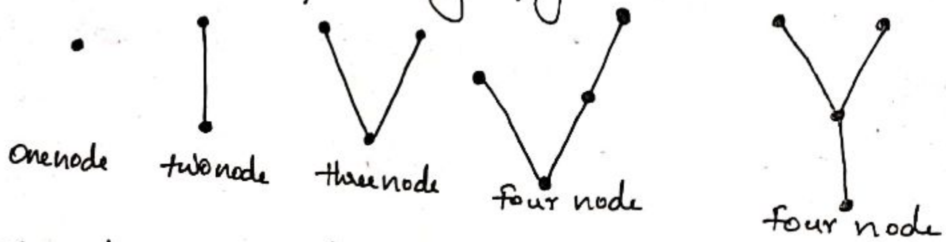
Trees and Graph Algorithms

Trees.

A tree is a connected graph without any circuit
The following shows a tree



Trees with, one, two, three and four vertices are shown in the following figure

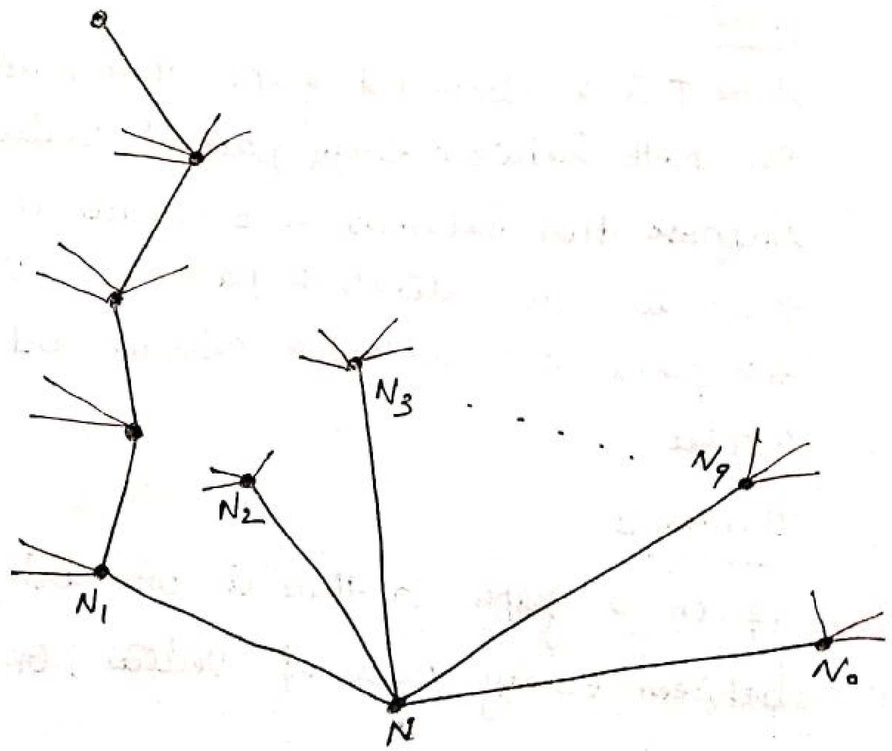


The edges of a tree are known as branches
Elements of tree are called their nodes
The nodes without child nodes are called leaf nodes.
Since we are considering finite graph, a graph must have atleast one vertex and therefore so must a tree must have atleast one node.

A tree has to be a simple graph, that is having neither a self-loop nor parallel edges.

Application of Trees.

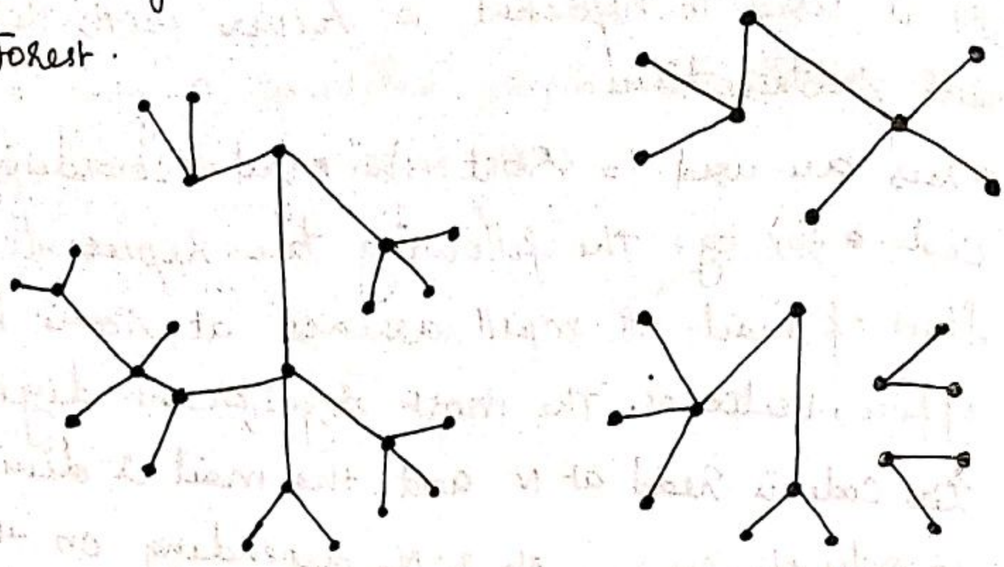
- Trees are used to represent the genealogy of a family
- It is used to represent a river with its tributaries and subtributaries
- Trees are used to sort the mail according to Zip code. for eg:- The following tree represents the flow of mail. All maill arrives at some local office, vertex N. The most significant digit in the Zip code is read at N and the mail is divided into 10 piles $N_1, N_2, \ldots N_9$ & N_0 depending on the most significant digit. Each pile is further divided into 10 piles according to the second most significant digit and so on till the mail is subdivided into 10^5 possible piles, each representing a unique 5-digit zip code.



Decision Tree.

Forest

A disconnected acyclic graph is called a forest or A disjoint collection of trees is called a forest.



Tree Forest.

Properties of Trees

Theorem - I

There is one and only one path between every pair of vertices in a tree T.

Proof:-

Since T is a connected graph, there must exist at least one path between every pair of vertices in T. Now suppose that between two vertices a and b of T there are two distinct paths. The union of these two paths will contain a circuit and T cannot be a Tree.

Theorem - II

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

Proof:

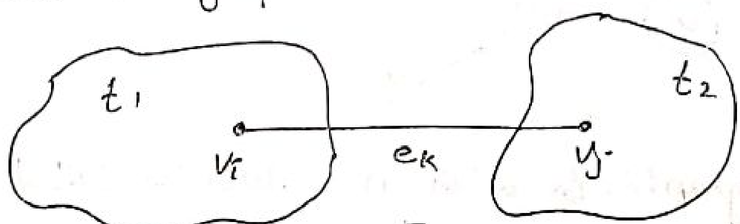
Existence of a path between every pair of vertices assures that G is connected. A circuit in a graph (with two or more vertices) implies that there is atleast one pair of vertices a, b such that there are two distinct paths between a and b. Since G has one and only one path between every pair of vertices, G can have no circuit. Therefore G is a tree.

Theorem - 3

A tree with n vertices has n-1 edges.

Proof:

The theorem will be proved by the induction of no: of vertices. Let us now consider a tree T with n vertices. In T let e_k be an edge with end vertices v_i and v_j. According to theorem-I there is no other path between v_i and v_j except e_k. Therefore deletion of e_k from T will disconnect the graph, as shown in the following figure



T.

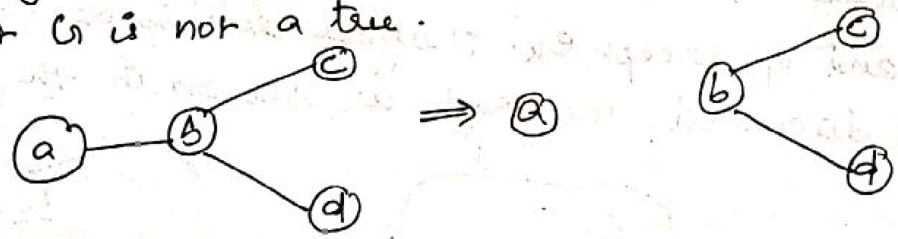
T-e_k consists of exactly two components and since there were no circuits in T to begin with each of these components is a tree. Both these trees t_1 and t_2, have fewer than n vertices each, and therefore by the induction hypothesis, each contains one less edge than the number of vertices in it. Thus T-e_k consists of n-2 edges. Hence T has exactly n-1 edges.

Theorem - 4.

A graph is a tree if and only if it is minimal connected.

Proof:-

In a tree, its vertices are connected together with the minimum number of edges. A connected graph is said to be minimally connected if removal of any one edge from it disconnects the graph. A minimally connected graph cannot have a circuit, otherwise we could remove one of the edges in the circuit and still leave the graph connected. Thus a minimally connected graph is a tree. Conversely if a connected graph G is not minimally connected, there must exist an edge e_i in G such that $G-e_i$ is connected. Therefore e_i is in some circuit, which implies that G is not a tree.



A graph G with n vertices is called a tree if

1. G is connected and is circuitless or
2. G is connected and has n-1 edges
3. G is circuitless and has n-1 edge
4. There is exactly one path between every pair of vertices in G
5. G is a minimally connected graph.

pendant Vertices in a tree

A pendant vertex was defined as a vertex of degree one. Each of the tree has several pendant vertices because in a tree of n vertices we have $n-1$ edges and hence $2(n-1)$ degrees to be divided among n vertices since no vertex can be of zero degree, we must have atleast two vertices of degree one in a tree.

Theorem -5

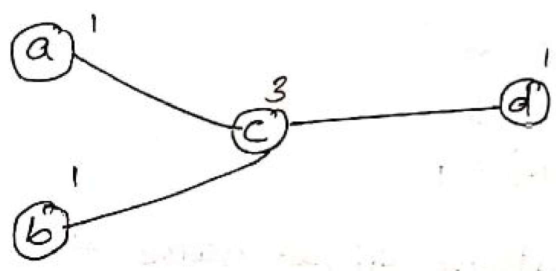
In any tree, there are at least two pendant vertices

proof:-

Let the number of vertices in a given tree T is n and $n \geq 2$. Therefore the no: of edges is $n-1$ using above theorems.

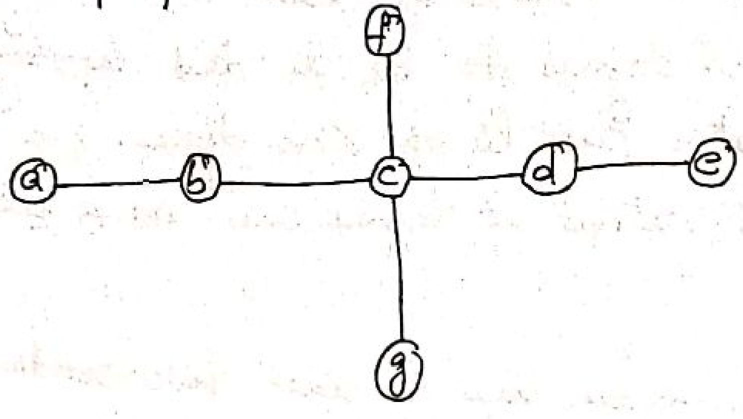
Summation of $\deg(v_i) = 2 * e = 2 * (n-1) = 2n-2$

the degree sum is to be divided among n vertices since a tree T is a connected graph, it cannot have a vertex of degree zero. Each vertex contributes at least one to the above sum. Thus there must be at least two vertices of degree 1. Hence every tree with atleast two vertices have atleast two pendant vertices



Distance and centers in a tree

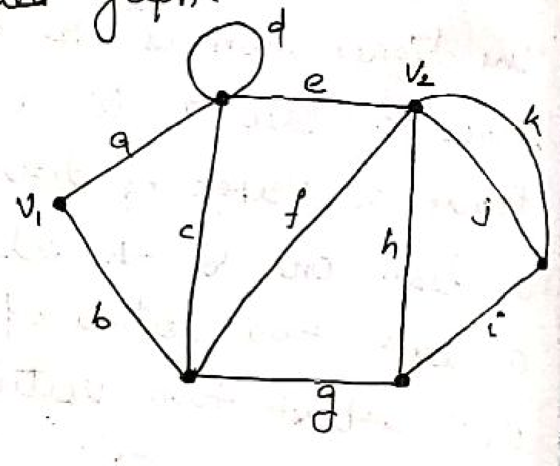
The following given tree has 7 vertices, and it seems that vertex c is located more "centrally" than any of the other three vertices



In a connected graph, G the distance $d(V_i, V_j)$ between two of its vertices V_i and V_j is the length of the shortest path (i.e. the number of edges in the shortest path) between them.

The definition of distance between any two vertices is valid for any connected graph.

Example:-



Path between vertices V_1 & V_2 are.

(a, e) - 2 (no: of edges)
(a, c, f) - 3
(b, c, e) - 3
(b, f) - 2
(b, g, h) - 3
(b, g, i, k) - 4

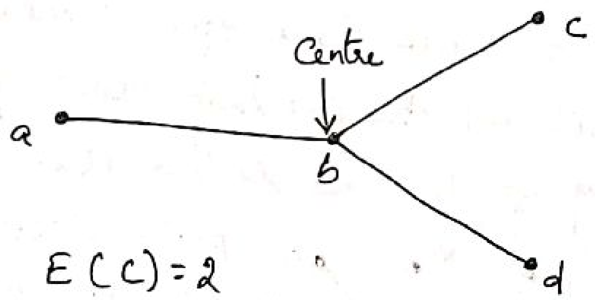
Of the above given paths the shortest paths are (a, e) or (b, f) $\therefore d(V_1, V_2) = 2$.

Eccentricity

The Eccentricity (also known as associated number or Seperation) $E(v)$ of a vertex is a graph G is the distance from v to the vertex farthest from v in G, that is

$$E(v) = \max_{v_i \in G} d(v, v_i)$$

Eg:



$E(a) = 2$ $E(c) = 2$
$E(b) = 1$ $E(d) = 2$

A vertex with minimum eccentricity in a graph G is called a centre of G. Hence vertex b is the centre of the above tree.

A vertex with maximum eccentricity in a graph G is called the diameter of a tree.

Consider the following Tree.

$E(A) = 3$ $E(C) = 2$ $E(E) = 3$
$E(B) = 3$ $E(D) = 2$ $E(F) = 3$

Here two vertices have minimum eccentricity value. $E(C) = E(D) = 2$. So this tree has two centres and the tree can be called as bicentres

Theorem-6

Every tree has either one or two centres

Proof:-

The maximum distance max $d(v, v_i)$ from a given vertex v to any other vertex v_i occurs only when v_i is a pendant vertex.

Let us start with a tree T having more than two vertices. Tree T must have two or more pendant vertices. Delete all the pendant vertices from T. The resulting graph T' is still a tree.

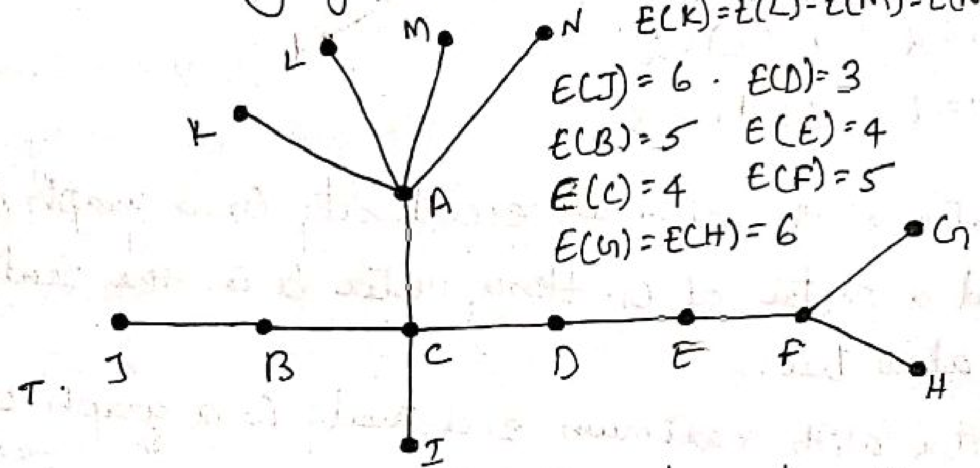
$E(K) = E(L) = E(M) = E(N) = 6$

$E(J) = 6$. $E(D) = 3$

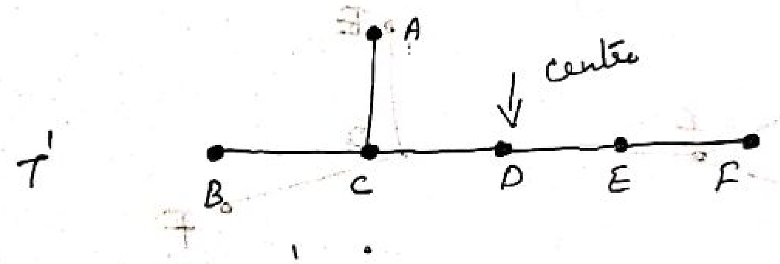
$E(B) = 5$ $E(E) = 4$

$E(C) = 4$ $E(F) = 5$

$E(G) = E(H) = 6$



Remove all pendant vertex from above tree T

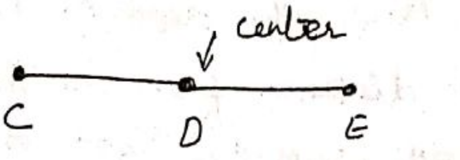


Eccentricity of T' is

$E(A) = 4$ $E(C) = 3$ $E(E) = 3$

$E(B) = 4$ $E(D) = 2$ $E(F) = 4$.

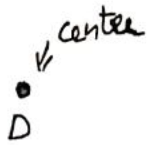
Remove all pendant vertex from T'



T''

$E(C) = 2$
$E(D) = 1$
$E(E) = 2$

Remove pendant vertex from T''



$E(D) = 0$.

The removal of all pendant vertices from T uniformly reduced the eccentricities of the remaining vertex of T' by one. All vertices that T had as centers will still remain centers in T'. From T' we get another tree T''. This process is continued we left with either a vertex or an edge of T. Thus the theorem proved.

Rooted Trees

A tree in which one vertex is distinguished from all the others is called a rooted tree. A root vertex is generally marked distinctly and the root node is enclosed in a small triangle. All rooted trees with four vertices are shown in the following figure.

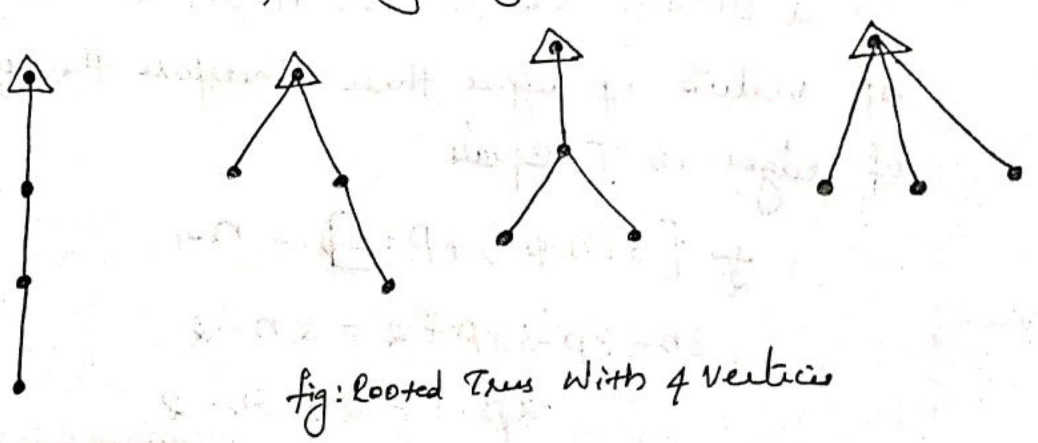


fig: Rooted Trees With 4 vertices

In a rooted tree, the depth or level of a vertex v is its distance from the root, i.e. length of the unique path from the root to v. Thus the root has depth 0.

The height of a rooted tree is the length of a longest path from the root (the greatest depth in the tree).

Binary Trees.

A binary tree is defined as a tree in which there is exactly one vertex of degree two, and each of the remaining vertices is of degree one or three. The vertex of degree two is distinct from all other vertices, this vertex is serves as a root. Thus every binary tree is a rooted tree.

Two properties of binary trees are.

1. The number of vertices n in a binary tree is always odd. This is because there is exactly one vertex of even degree and the remaining $n-1$ vertices are of odd degrees.

2. Let p be the number of pendant vertices in a binary tree T. Then $n-p-1$ is the no: of vertices of degree three. Therefore the number of edges in T equals.

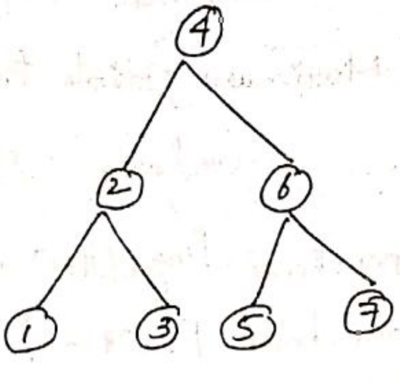
$$\frac{1}{2}\left[3(n-p-1)+p+2\right] = n-1$$

$$3n-3p-3+p+2 = 2n-2$$

$$-2p-1 = 2n-3n-2$$

$$-2p = -n-2+1$$
$$-2p = -n-1$$
$$2p = n+1$$
$$p = \frac{n+1}{2}$$

In the graph given



No: of vertices $n = 7$

No: of edges $= n-1 = 6$

No: of pendant vertices

$$p = \frac{n+1}{2} = \frac{7+1}{2} = 4 \quad (1,3,5,7)$$

No: of vertices of degree 3

$$= n-p-1 = 7-4-1 = 2 \quad (2,6)$$

A nonpendant vertex in a tree is called an internal vertex: The no: of internal vertices in a binary tree is one less than the number of pendant vertices.

In a binary tree a vertex v_i is said to be at level l_i if v_i is at a distance of l_i from the root. Thus the root is at level 0.

Eg:-

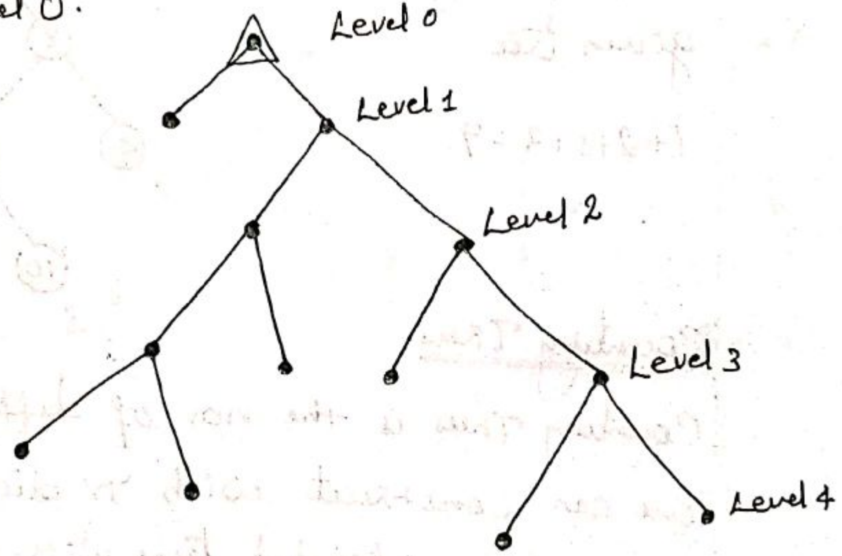


fig: 13 vertex, 4-level binary tree

The maximum level, l_{max} of any vertex in a binary tree is called the Height of the tree.

- Minimum possible height of a binary tree

$$\min l_{max} = \lceil \log_2(n+1) - 1 \rceil$$

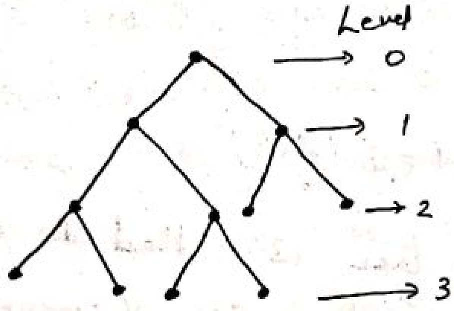
- Maximum possible height of a binary tree

$$\max l_{max} = \frac{n-1}{2}$$

$$\min l_{max} = \lceil \log_2(11+1) - 1 \rceil$$

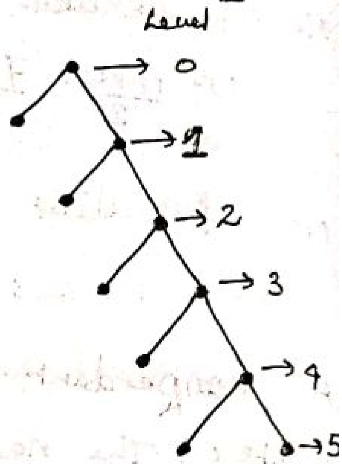
$$\min l_{max} = \lceil 3.58 - 1 \rceil$$

$$\min l_{max} = \lceil 2.58 \rceil = 3$$



$$\max l_{max} = \frac{n-1}{2}$$

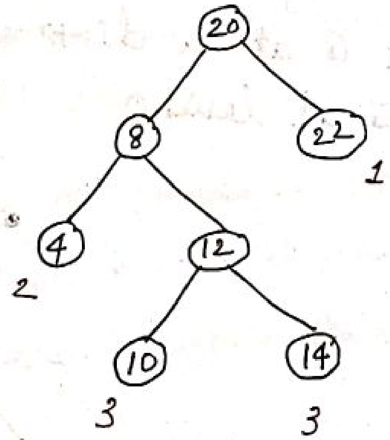
$$\max l_{max} = \frac{11-1}{2} = 5$$



Sum of the path lengths from the root to all pendant vertices is called path length or External path Length.

The path Length of a given tree

$1+2+3+3 = 9$

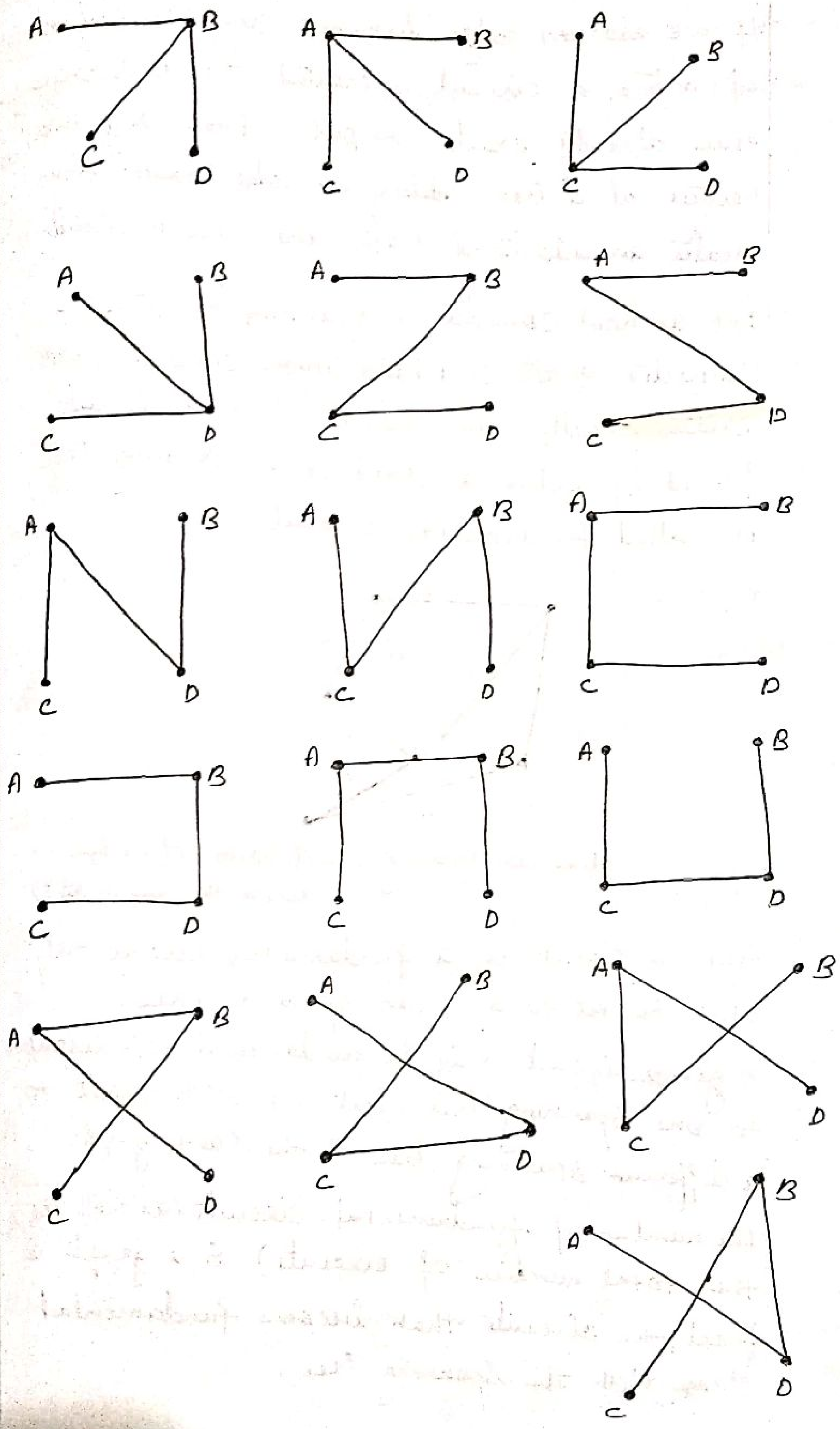


Counting Trees

Counting Trees is the no. of different trees that one can construct with 'n' distinct vertices.

The number of labeled trees with n vertices is n^{n-2}.

If $n=4$. we have the following different trees

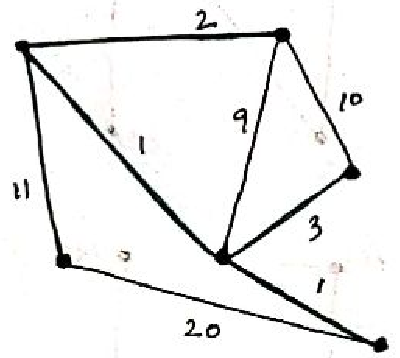


Fundamental circuits

If we add an edge between any two vertices of a tree a circuit is created. This is because there already exists one path between any two vertices of a tree, adding an edge between them creates an additional path, and hence a circuit

Let us now consider a spanning tree T in a connected graph G. Adding anyone chord to T will create exactly one circuit. Such a circuit formed by adding a chord to a spanning tree is called fundamental circuit

Eg:-



Minimum spanning Trees (Green color edges denote the edges in MST)

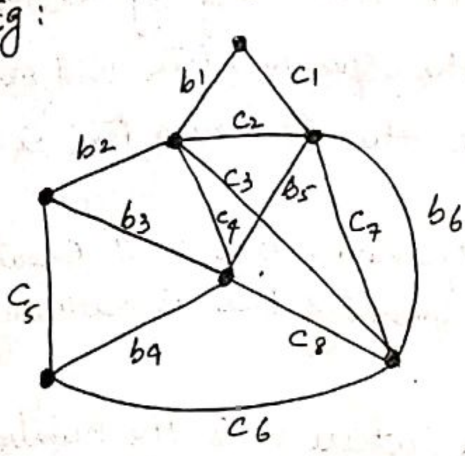
First a circuit is a fundamental circuit only. With respect to a given spanning tree. A given circuit may be fundamental with respect to one spanning tree, but not with respect to a different spanning tree of the same graph.

The number of fundamental circuits (as well as the number of circuits) in a graph is the total number of circuits) in a graph is fixed, the circuits that become fundamental change with the spanning tree.

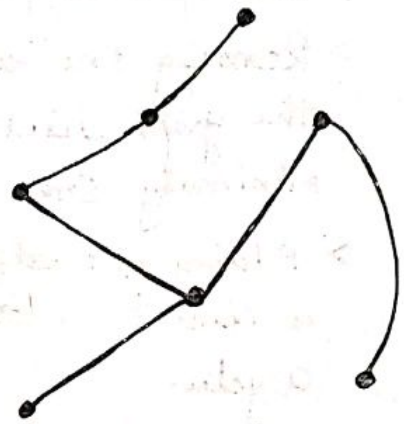
Spanning Trees

A tree T is said to be a spanning tree of a connected graph G if T is a subgraph of G and T contains all vertices of G.

Eg:



Graph G



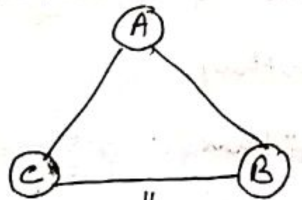
Spanning Tree of G

Since the vertices of G are barely hanging together in a spanning tree, it is a kind of skeleton of the original graph G. So the spanning trees are sometimes called as skeleton or scaffolding. of G.

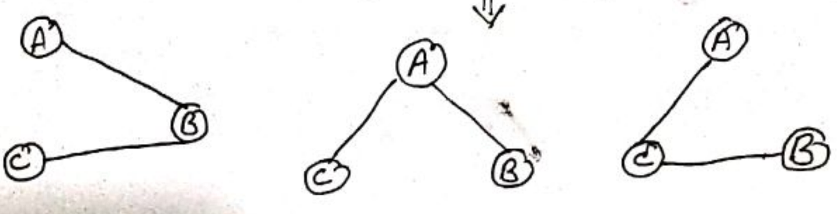
Spanning Trees are the largest trees among all trees in G, it is also called as maximal tree subgraph or maximal tree of G.

A connected graph G can have more than one spanning tree.

Eg:



Graph G



spanning tree of G

Properties of Spanning Tree.

> A Connected graph G can have more than one Spanning tree.

> All possible spanning trees of graph G, have the Same number of edges and Vertices

> The spanning tree does not have any cycle (loop).

> Removing one edge from the spanning tree will make the graph disconnected, ie the Spanning tree is minimally Connected.

> Adding one edge to the spanning tree will create a circuit or loop, ie the Spanning tree is maximally acyclic

> Spanning tree has n-1 edges, where n is the number of nodes (vertices).

> From a complete graph, by removing maximum e-n+1 edges, we can construct a spanning tree

> A complete graph can have maximum n^{n-2} number of spanning trees.

Minimum Spanning Tree (MST)

In a weighted graph, a minimum spanning tree is a spanning tree that has minimum weight than all other spanning trees of the same graph.

There are mainly two minimum spanning tree algorithm used to find MST of a connected graph.

→ Kruskal's Algorithm

→ Prim's Algorithm

Primi Spanning Tree Algorithm

Primi algorithm is used to find minimum cost spanning tree uses the greedy approach.

Prim's algorithm treats the nodes as a single tree and keeps on adding new nodes to the spanning tree from the given graph.

Steps in Prims Algorithm.

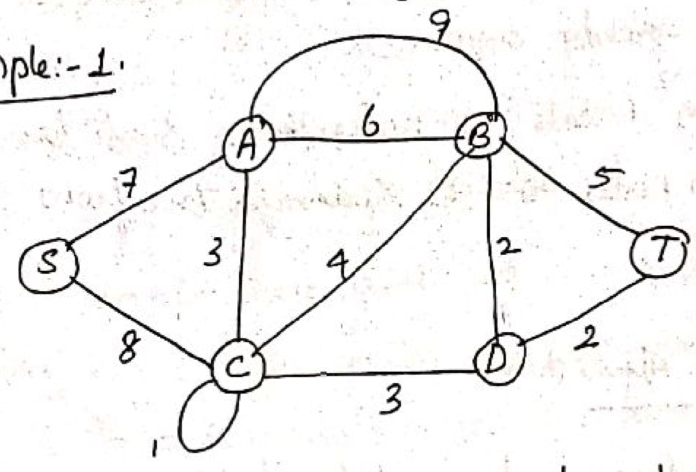
1. Remove all loops and parallel edges.
2. Choose a vertex v at random from the graph $G=(V,E)$.
3. Choose a minimum-weight edge (u,v) connecting a vertex v in the set A to the vertex u outside of set A. in each iteration.
4. Step 3 is repeated until a spanning tree is formed.

Algorithm.

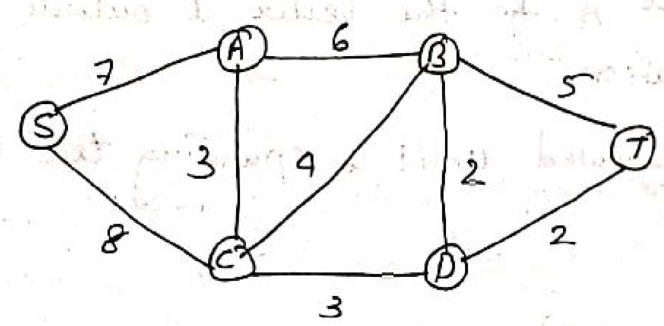
```
MST-PRIME(G,w,r)
for each u ∈ V[G]
    do key[u] ← ∞
       π[u] ← NIL
key[r] ← 0
Q ← V[G]
while Q ≠ φ
   do u ← extract-min(Q)
      for each v ∈ Adj[u]
          do if v ∈ Q and w(u,v) < key[v]
               then π[v] ← u
                    key[v] ← w(u,v).
```

Complexity of prim's Algorithm is $O(E \log V)$

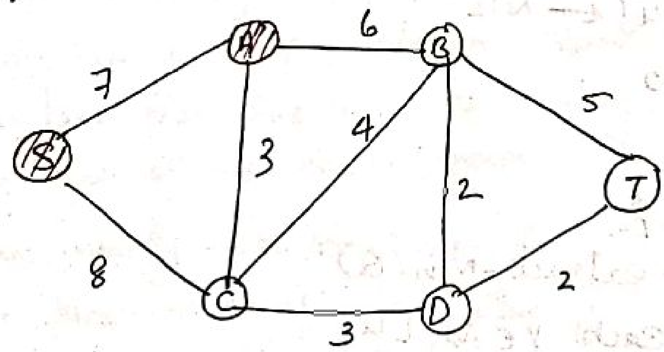
Example:- 1.



Remove all loops and parallel edges. In case of parallel edges, keep the one which has the least cost associated and remove all others.

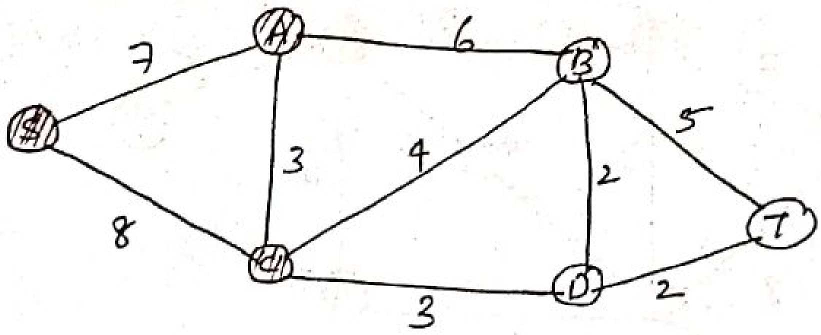


Choose any random node as root node. We can choose 'S' as root node. Check out the outgoing edges from 's' and select the one with least cost



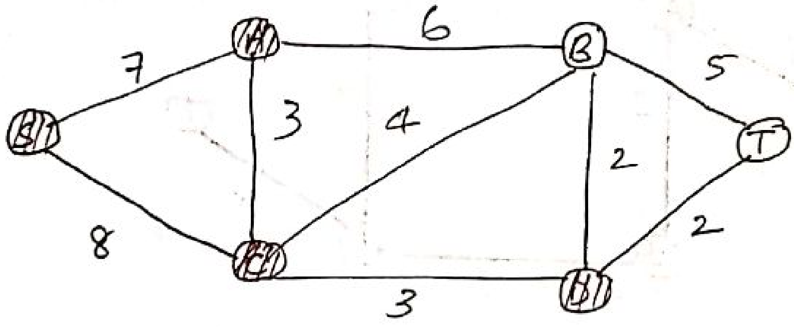
$A = \{S\}$

Now the tree S-7-A is treated as one node and we check for all edges going out from it and select the one with lowest cost and include it in the tree.



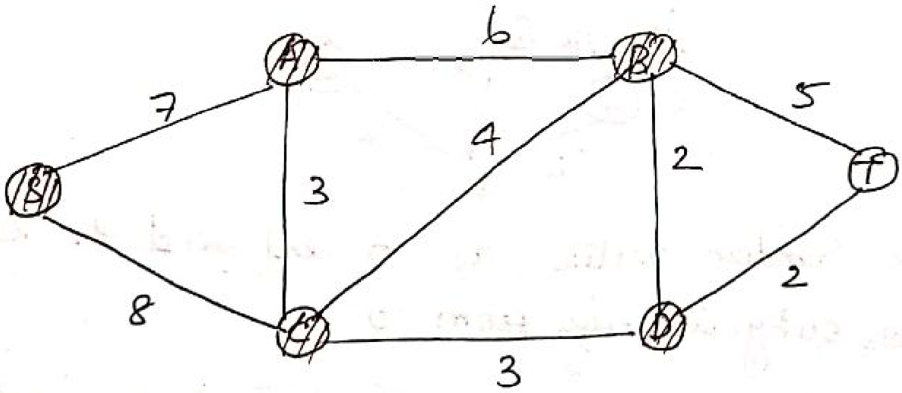
A: { S, A, C }

Now the tree is S-7-A-3-C and check all the edges again. The edges are 8, 6, 4, 3. choose the least cost edge is C-3-D and add vertex C to set A



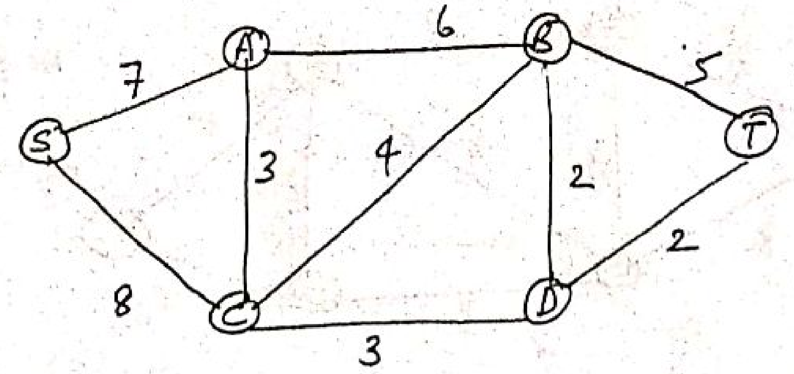
A = { S, A, C, D }

Now the tree is S-7-A-3-C-3-D and check all the edges again. The edges are 8, 6, 4, 2, 2. Two least cost edges are present D-2-B and D-2-T. We can choose any of this edge to the spanning tree. Now we are choosing D-2-B.



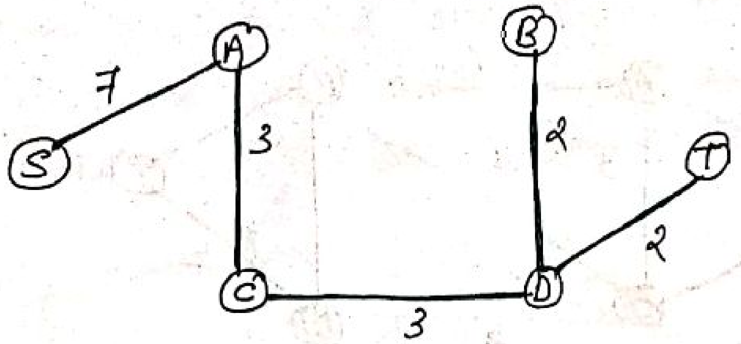
A = { S, A, C, D, B }

Now the tree is S-7-A-3-C-3-D-2-B and check all the edges again. The edges are 8, 6, 4, 5, 2. out of this 2 is least cost. So we choose D-2-T and add T to vertex set.



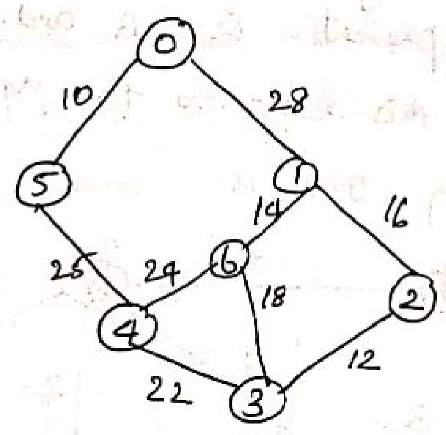
A = {S, A, C, D, B, T}

Since all vertices are added to the set A. we get the final spanning tree as.

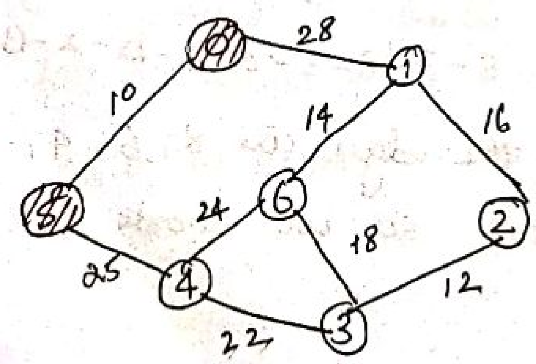


Minimum cost $= 7+3+3+2+2+2 = \underline{\underline{19}}$

Example 2:-

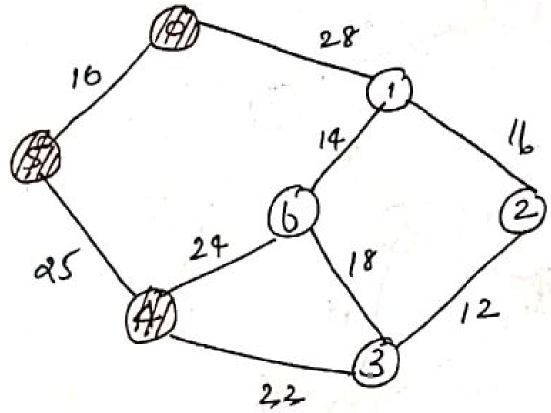


Choose random vertex as '0' and find the least weight outgoing edge from '0'



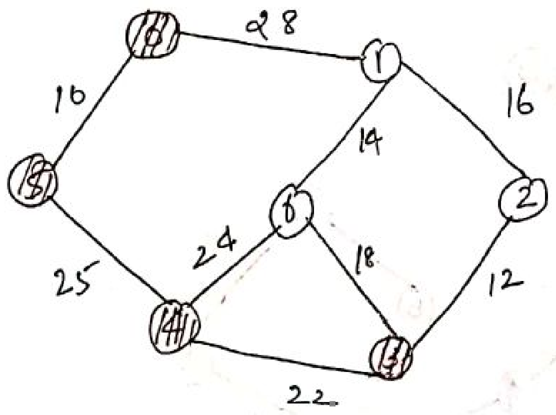
A = {0, 5}

Now consider the tree 0-10-5 and find the least cost edge It is 5-25-4



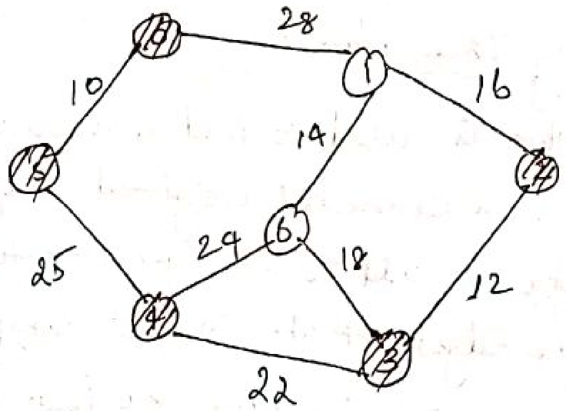
A={0,5,4}

Now consider the tree 0-10-5-25-4 and find the least cost edge. It is 4-22-3.



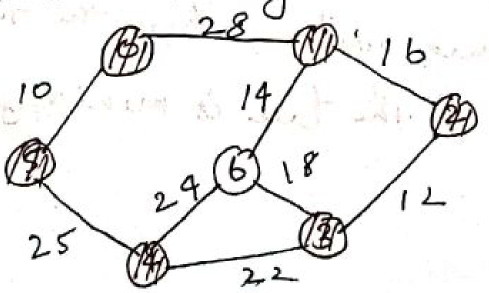
A={0,5,4,3}

Now consider the tree 0-10-5-25-4-22-3. and find the least cost edge it is 3-12-2.



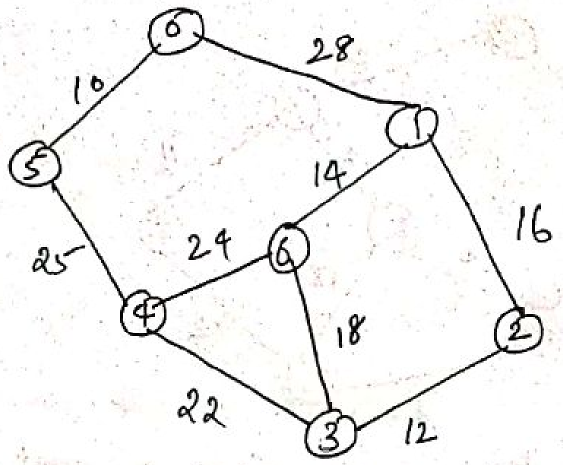
A={0,5,4,3,2}

Now consider the tree 0-10-5-25-4-22-3-12-2 and find the least cost edge it 2-16-1

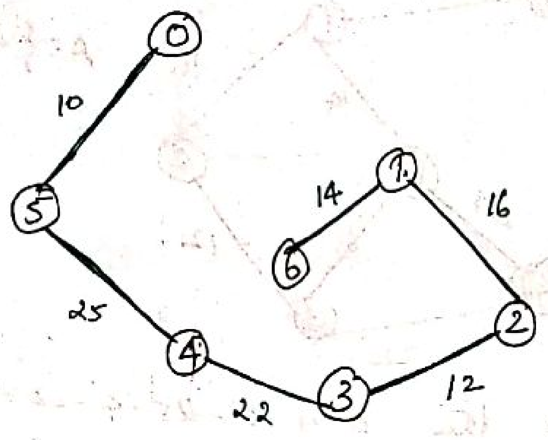


A={0,5,4,3,2,1}

The next least cost edge is : 1-14-6.



So the final spanning Tree is



Minimum cost = 10 + 25 + 22 + 12 + 16 + 14 = 99

Kruskals Spanning Tree Algorithm.

Kruskals algorithm is used to find a minimum Spanning tree for a connected weighted graph. It finds a safe edge to add to the growing forest by finding of all the edges that connect any two trees in the forest, an edge (u,v) of least weight. It finds a subset of the edges that forms a tree that includes every vertex, where the total weight of all the edges in the tree is minimized.

Note:-

If the graph is not connected, then it finds a minimum spanning forest.

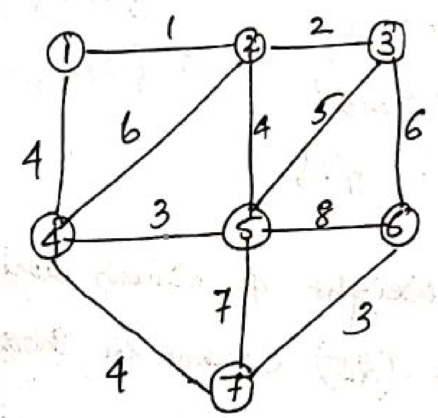
MST-KRUSKAL(G, w)

1. $A \leftarrow \phi$
2. for each vertex $v \in V[G]$
3. do MAKE-SET (v)
4. Sort the edges of E into nondecreasing order by weight w
5. for each edge $(u, v) \in E$, taken in nondecreasing order by weight
6. do if FIND-SET(u) $\neq$ FIND-SET(v)
7. then $A \leftarrow A \cup \{(u, v)\}$
8. UNION (u, v)
9. Return A.

Steps to find MST using kruskal's algorithm.

1. Sort all the edges in non-decreasing order of their weight
2. Pick the smallest edge. check if it forms a cycle with the spanning tree formed so far. If cycle is not formed, include this edge. Else discard it.
3. Repeat step 2 until there are (v-1) edges in the spanning tree

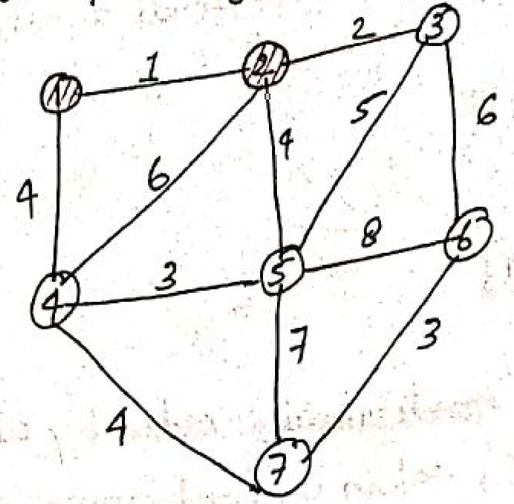
Example 1:



Sort edges by weight.

1: {1,2}
2: {2,3}
3: {4,5}
3: {6,7}
4: {1,4}
4: {2,5}
4: {4,7}
5: {3,5}
6: {2,4}
6: {3,6}
7: {5,7}
8: {5,6}

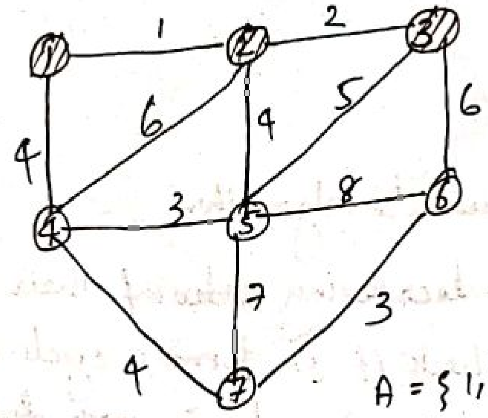
Add first edge to X if no cycle ...



1: {1,2} ✓
2: {2,3}
3: {4,5}
3: {6,7}
4: {1,4}
~~4: {2,5}~~
4: {4,7}
5: {3,5}
6: {2,4}
6: {3,6}
7: {5,7}
8: {5,6}

A = {1,2}

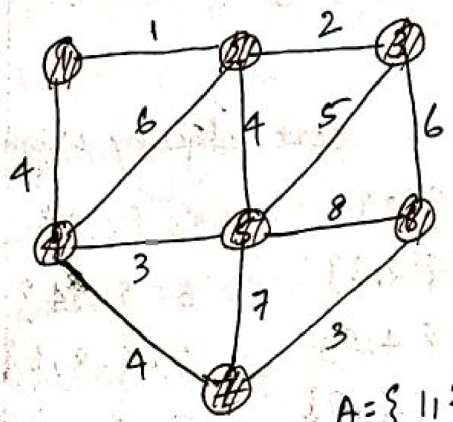
Next choose the edge with 2 weight if it is not form a cycle



1: {1,2} ✓
2: {2,3} ✓
3: {4,5}
3: {6,7}
4: {1,4}
4: {2,5}
4: {4,7}
5: {3,5}
6: {2,4}
6: {3,6}
7: {5,7}
8: {5,6}

A = {1,2,3}

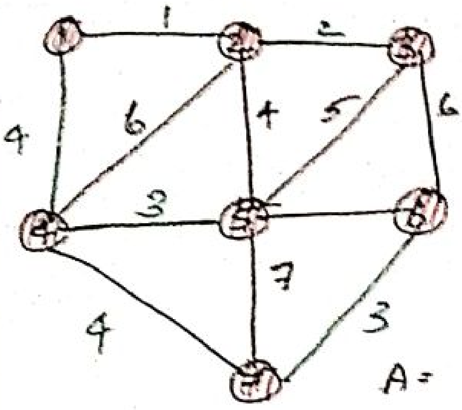
Next choose edge with weight 3 if it is not form a cycle.



1: {1,2} ✓
2: {2,3} ✓
3: {4,5} ✓
3: {6,7} ✓
4: {1,4}
4: {2,5}
4: {4,7}
5: {3,5}
6: {2,4}
6: {3,6}
7: {5,7}
8: {5,6}

A = {1,2,3,4,7,5,6}

Next choose the edge with weight 4 which does not form a cycle in the tree. (2,5) cannot be choose since it form a cycle.

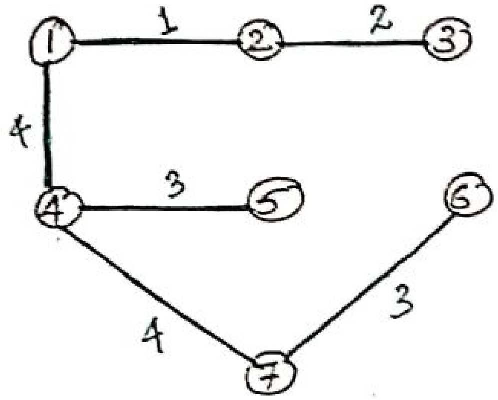


A =

1 : {1,2} ✓
2 : {2,3} ✓
3 : {4,5} ✓
3 : {6,7} ✓
4 : {1,4} ✓
4 : {2,5} ✗
4 : {4,7} ✓
5 : {3,5}
6 : {2,4}
6 : {3,6}
7 : {5,7}
8 : {5,6}

Since all vertices in graph are visited the adding of edge can be stopped.

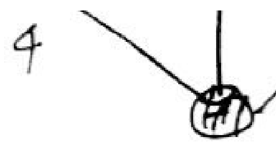
The final spanning Tree.



Total cost =

4+1+2+3+4+3

= 17



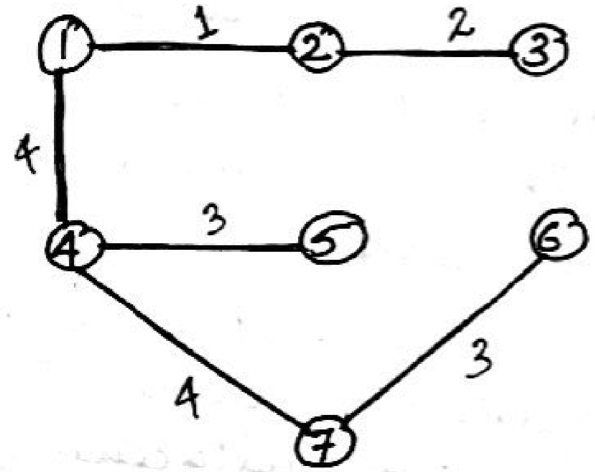
A =

4 : {2,5} ×   0 : {5,6}

4 : {4,7} ✓

Since all vertices in graph are visited the adding of edge can be stopped.

The final spanning Tree



Total cost =

4+1+2+3+4+3

= 17

Dijikstra's shortest path Algorithm

Dijikstra's Algorithm is used for solving the single source shortest path problem. It computes the shortest path from one particular source node to all other remaining nodes of the graph.

Conditions for Dijikstra's Algorithm.

→ It work only for connected graphs.

→ Dijikstra's Algorithm works only for those graphs that do not contain any negative weight edge.

→ It works for directed as well as undirected graphs

Dijikstra's Algorithm.

Step 1: Create a set sptset (Shortest path tree set) that keeps track of vertices included in Shortest path tree, ie whose minimum distance from source is calculated and finalized. Initially this set is empty.

Step 2: Assign a distance value to all vertices in the input graph. Initialize all distance values as INFINITE. Assign distance value as 0 for the source vertex so that it is picked first.

Step 3: While sptset doesn't include all vertices

a) pick a vertex 'u' which is not there in sptset and has minimum distance value.

b) Include u to sptset.

c) update distance value of all adjacent vertices of u. To update the distance values, iterate through all adjacent vertices For every adjacent vertex v, if sum of distance value of u (from source) and. weight of edge u-v is less than the distance value of v, then update the distance value of v

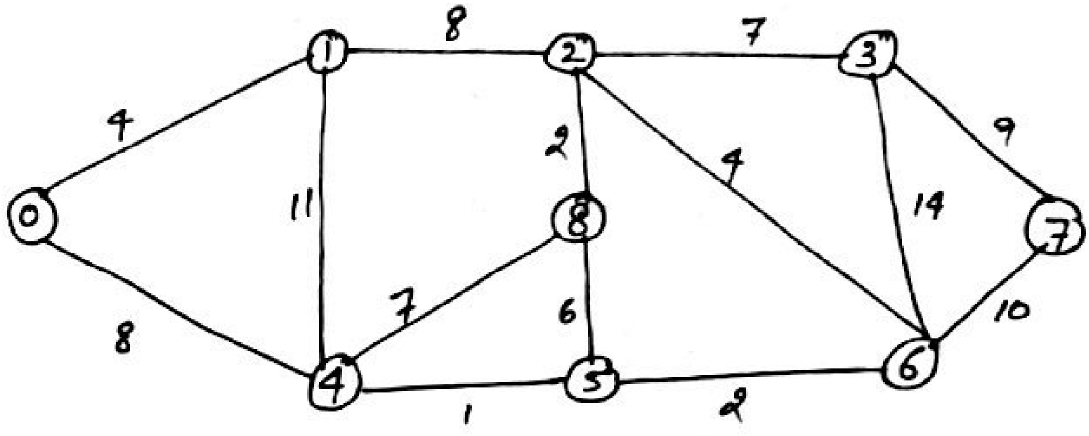
To update the distance value from source to all other vertices following formula is used. If we are considering two vertices u and v then

if $d(u) + c(u,v) < d(v)$

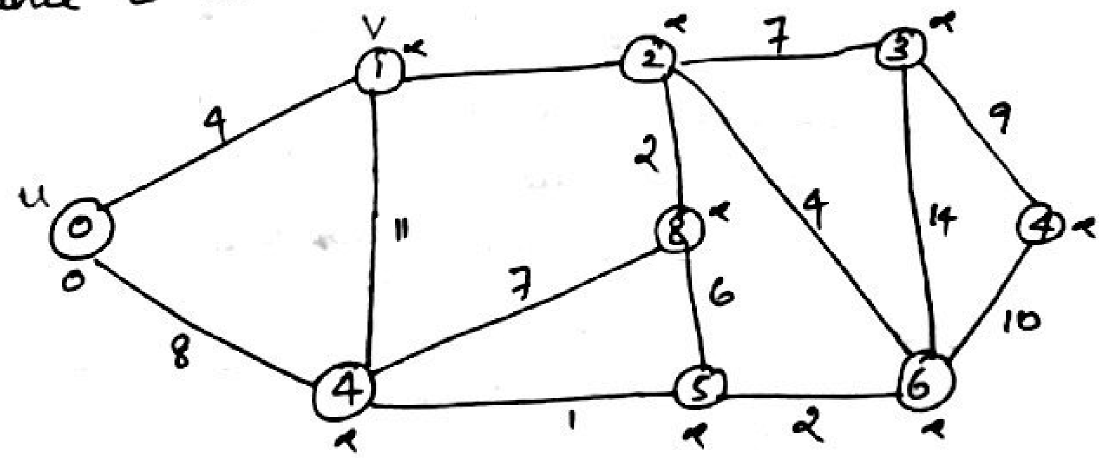
then $d(v) = d(u) + c(u,v)$.

Example 1:-

find the shortest path from the source vertex to all other vertices in the following graph using Dijikstra's Algorithm.



From the source vertex we have to find shortest path to all nodes.
Consider the source vertex as '0'
Distance from source vertex to itself is zero and in the above graph distance from vertex '0' to itself = 0.
Initially, from source vertex to all other vertex the distance is considered as '∞'



find the smallest distance from the source node '0' and replace '∞' with the smallest distance

The formula is

$d(u) + c(u,v) < d(v)$

then $d(v) = d(u) + c(u,v)$.

From source '0' vertex two edges (0,1) and (0,4)

The smallest distance from '0' to these two edges are

$0+4 < \infty$ toward (0,1)

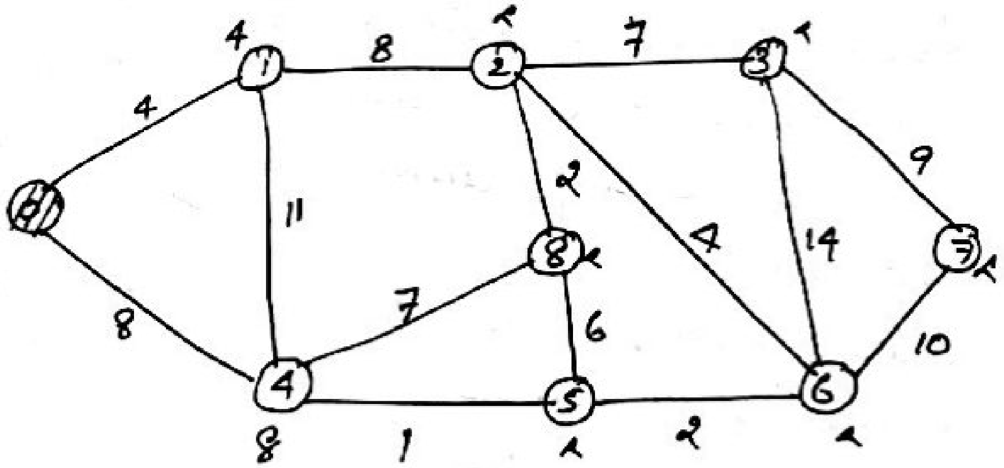
$4 < \infty$.

$\therefore dis(1) = 4$.

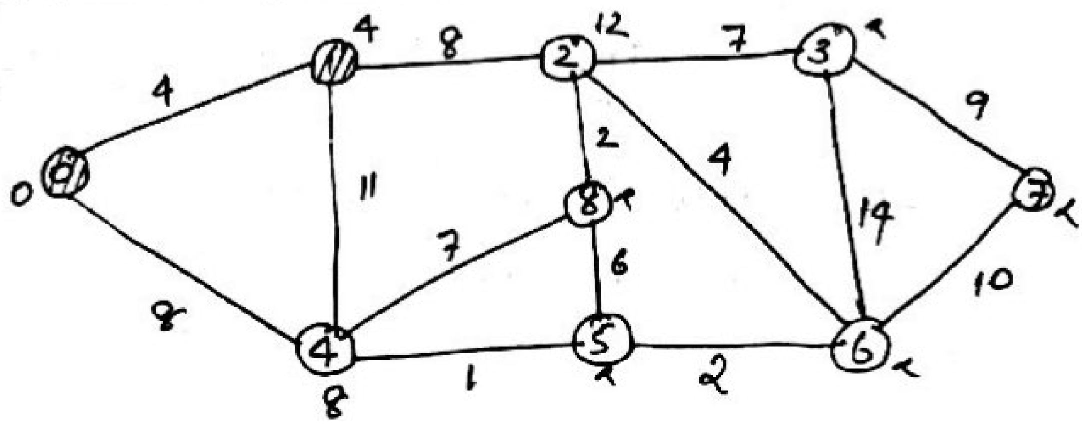
$0+8 < \infty$ toward (0,4)

$8 < \infty$.

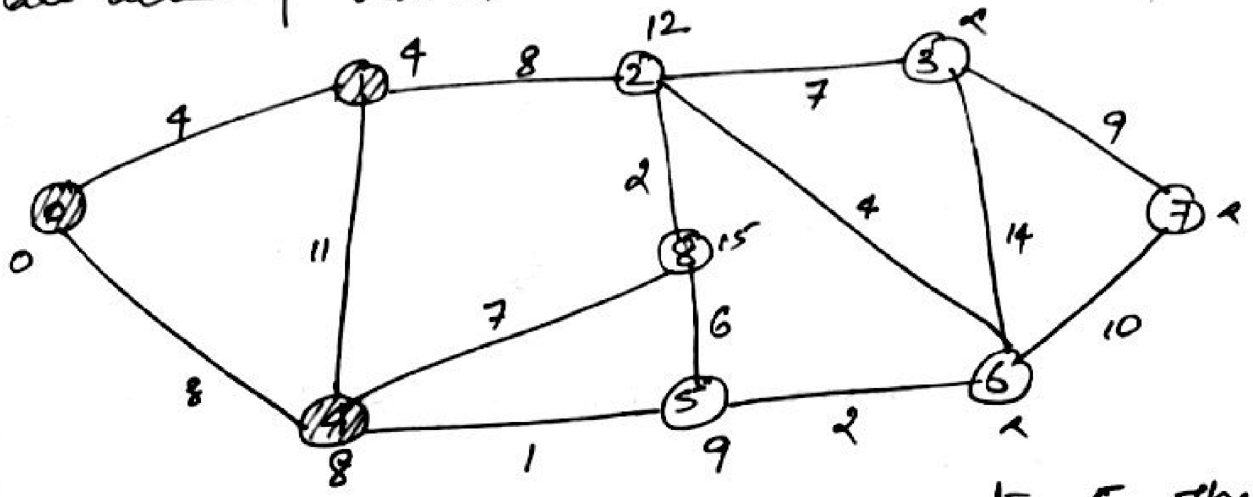
$\therefore dis(4) = 8$



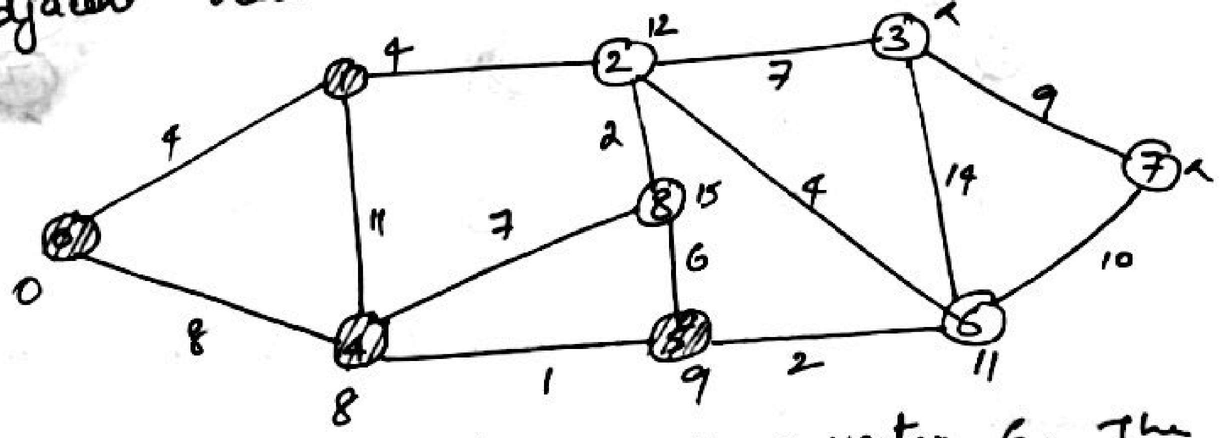
find the shortest distance from vertex 1. The adjacent vertex are 1-2, 1-4, 1-0 of which 1-0 is already visited. Then find the shortest distance to other 2 vertices



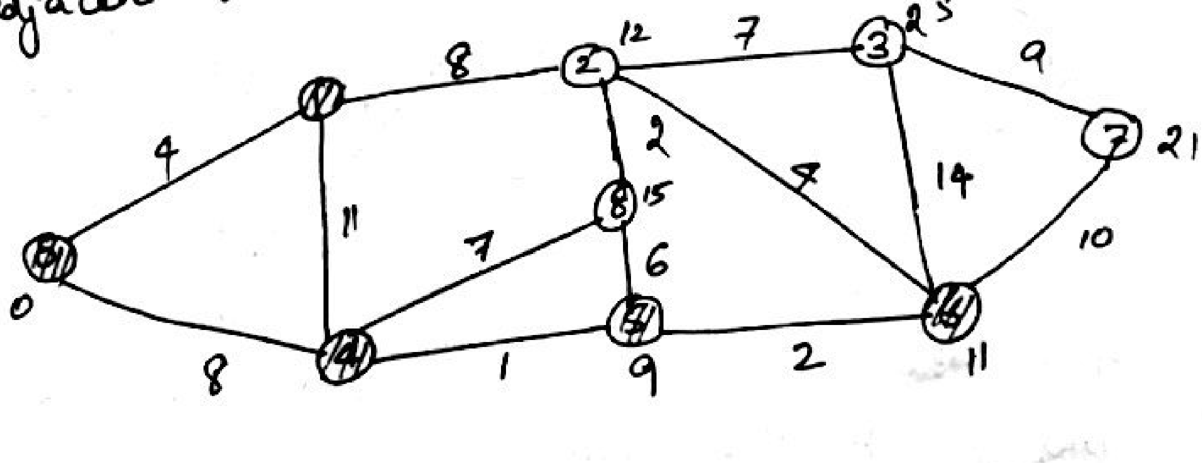
Then find the shortest distance from vertex 4 since it is the next shortest distance. Its edges are 4-5, 4-8, 4-1, 4-0. of which 4-0 & 4-1 are already visited



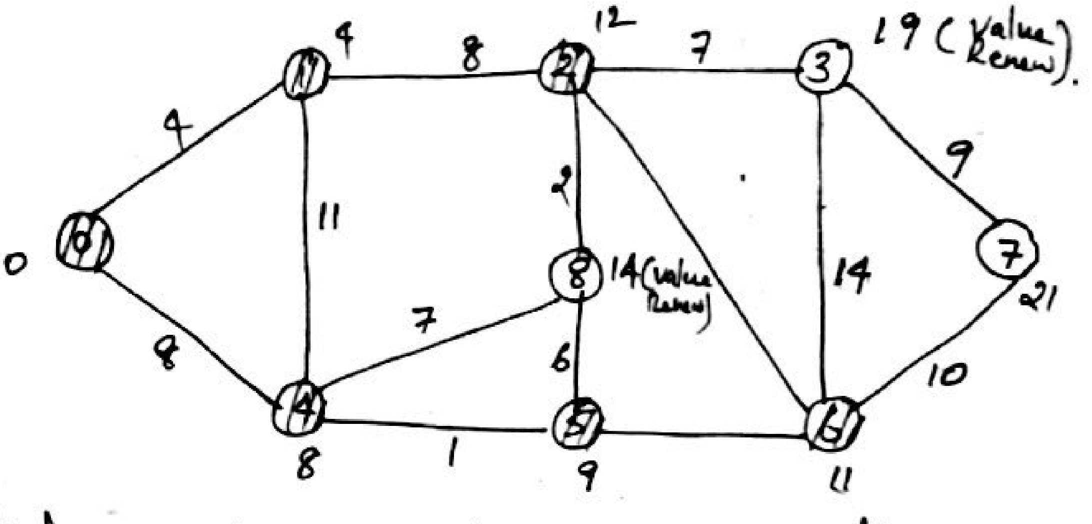
find the shortest distance from vertex 5. the adjacent vertex are 5-6, 5-8. 5-4.



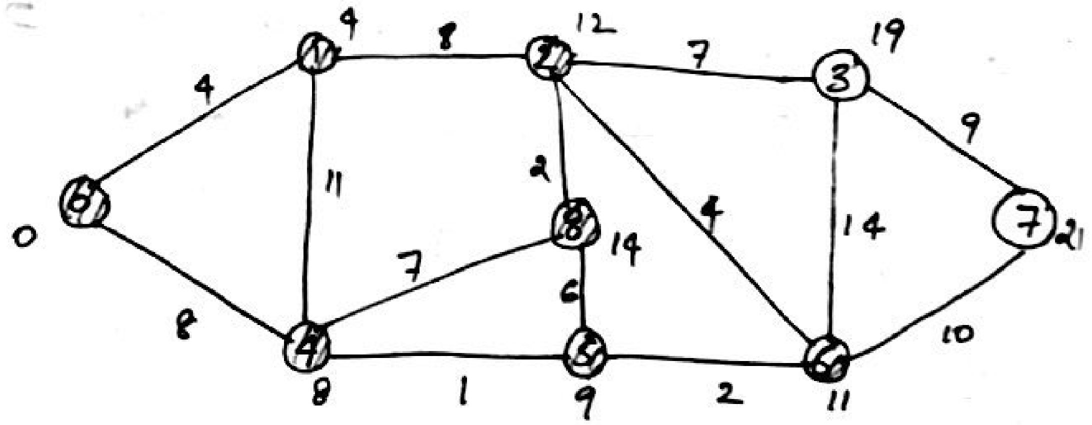
find the shortest distance from vertex 6. The adjacent vertex are 6-7, 6-3, 6-2



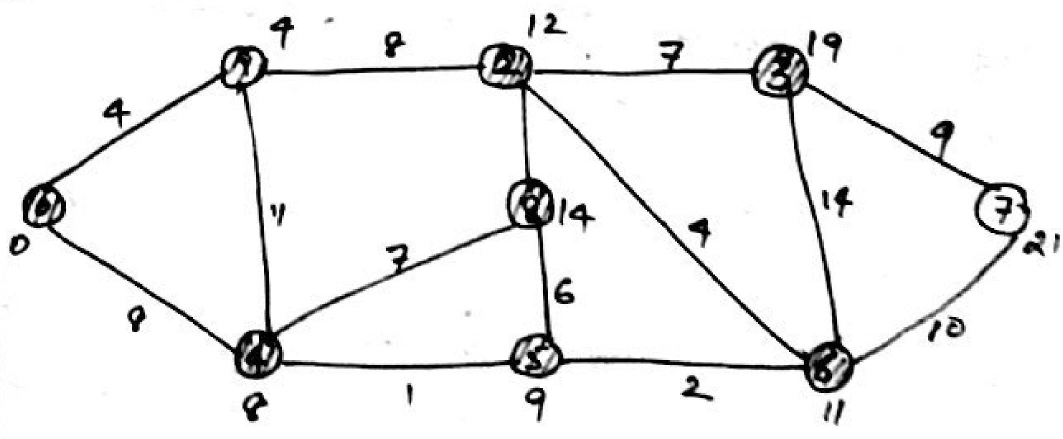
find the shortest distance from vertex 2 and select that vertex.



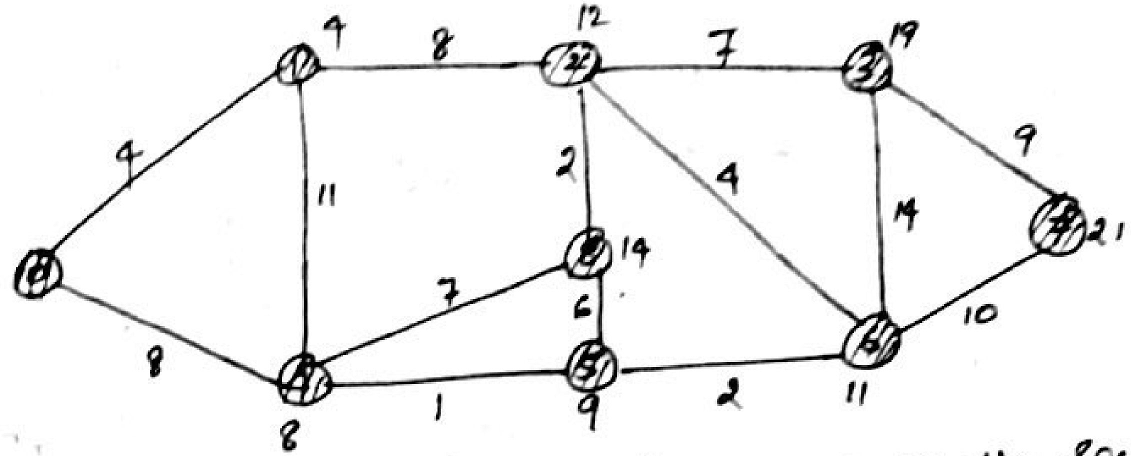
find the shortest distance from vertex 8 and select that vertex.



find the shortest distance from vertex 3 and select that vertex.



find the shortest distance from vertex 7 and select that vertex.

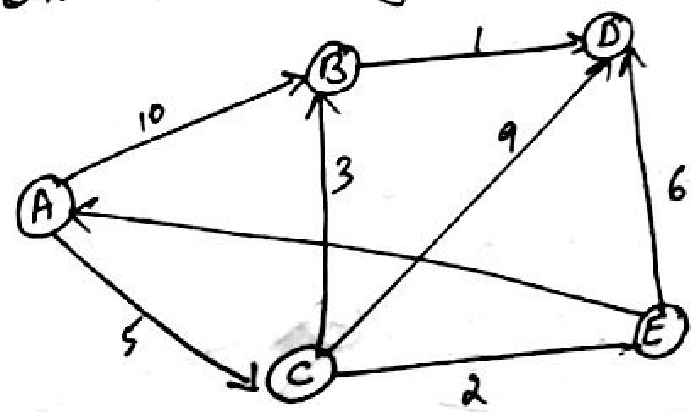


following are the shortest distance from the source vertex '0'.

0-1 = 4
0-4 = 8
0-2 = 12
0-8 = 14
0-5 = 9
0-3 = 19
0-6 = 11
0-7 = 21

Example 2:-

Find the shortest distance from the source vertex to all other vertex using Dijkstra's Algm.

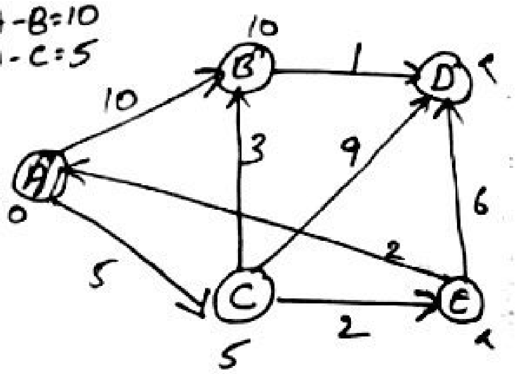


Consider the source vertex as 'A'

	A	B	C	D	E
A	$\boxed{0}$	∞	∞	∞	∞

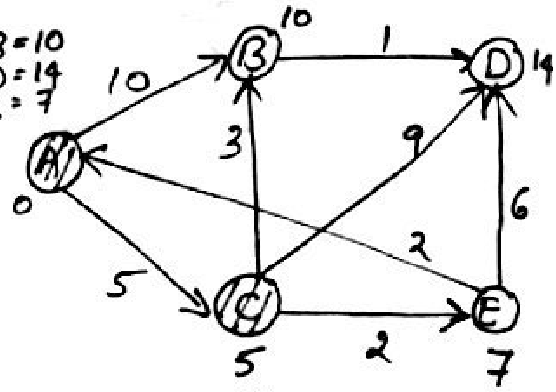
	A	B	C	D	E
A	$\boxed{0}$	∞	∞	∞	∞
C		10	$\boxed{5}$	∞	∞

A-B=10
A-C=5



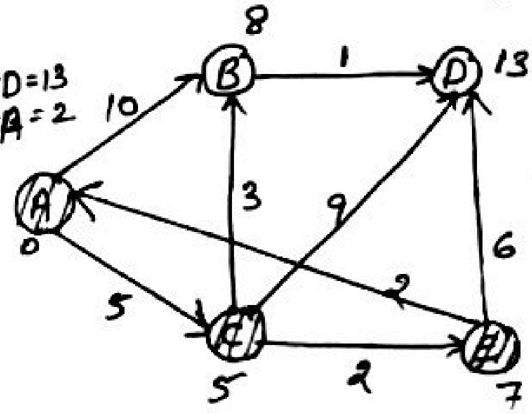
	A	B	C	D	E
A	$\boxed{0}$	∞	∞	∞	∞
C		10	$\boxed{5}$	∞	∞
E		8		14	$\boxed{7}$

C-B=10
C-D=14
C-E=7



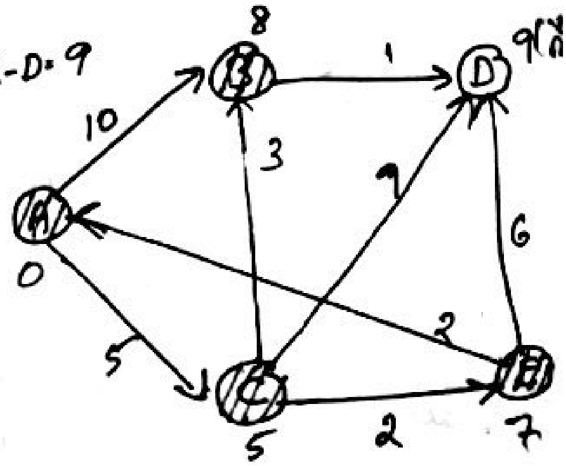
	A	B	C	D	E
A	$\boxed{0}$	∞	∞	∞	∞
C		10	$\boxed{5}$	∞	∞
E		8		14	$\boxed{7}$
B		$\boxed{8}$		13	

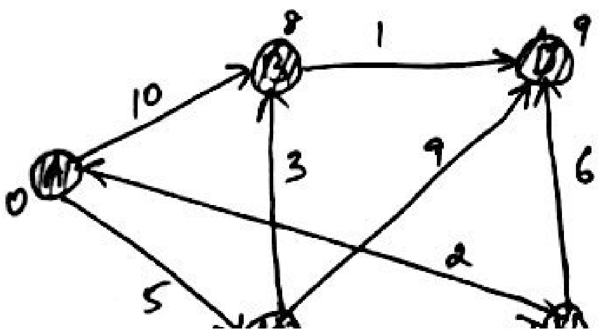
E-D=13
E-~~A~~=2



	A	B	C	D	E
A	$\boxed{0}$	∞	∞	∞	∞
C		10	$\boxed{5}$	∞	∞
E		8		14	$\boxed{7}$
B		$\boxed{8}$		13	
D				$\boxed{9}$	

B-D=9





Shortest Distance from A.

A-B = 8

A-C = 5

A-D = 9

A-E = 7

Shortest path from vertex A-D = DBCA

Shortest path from vertex A-E = ECA.

Shortest path from vertex A-B = BCA.

Floyd Warshall Algorithm.

Floyd Warshell algorithm is used to find all pair shortest path problem from a given weighted path. This algorithm will generate a matrix, which will represent the minimum distance from any node to all other node in the graph.

Algorithm:-

Step 1: Initialize the shortest paths between any two vertices with infinity

Step 2: Find all pair shortest paths that use 0 intermediate vertices, then find the shortest paths that use 1 intermediate vertex and so on until using all N vertices as intermediate Node.

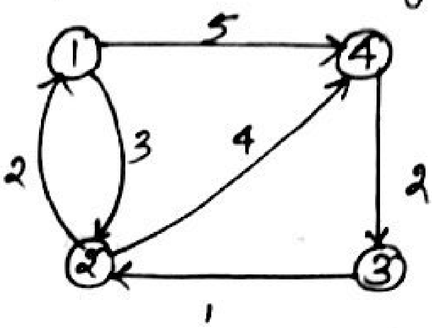
Step 3: Minimize the shortest paths between any 2 pairs in the previous operation.

Step 4: For any 2 vertices (i,j) one should actually minimize the distance between this pair using the first K nodes, so the shortest path will be min(dist[i][K] + d[K][j], dist[i][j]).
dist[i][K] represents the shortest path that only uses the first K vertices.

dist[k][j] Represents the shortest path between the pair k, j. As the shortest path will be a concatenation of the shortest path from i to k, then from k to j.

Example 1:-

Find the shortest path of the following graph using Floyd Warshall Algm.



$$A^0 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 3 & \infty & 5 \\ 2 & 2 & 0 & \infty & 4 \\ 3 & \infty & 1 & 0 & \infty \\ 4 & \infty & \infty & 2 & 0 \end{array}$$

In A^0 matrix all diagonal elements are equal to 0. and Directed edge weight value is Represented

$$A^1 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 3 & \infty & 5 \\ 2 & 2 & 0 & & \\ 3 & \infty & & 0 & \\ 4 & \infty & & & 0 \end{array} = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 3 & \infty & 5 \\ 2 & 2 & 0 & \infty & 4 \\ 3 & \infty & 1 & 0 & \infty \\ 4 & \infty & \infty & 2 & 0 \end{array}$$

$$A^2 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 3 & & \\ 2 & 2 & 0 & \infty & 4 \\ 3 & & 1 & 0 & \\ 4 & & \infty & & 0 \end{array} = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 3 & \infty & 5 \\ 2 & 2 & 0 & \infty & 4 \\ 3 & \underline{3} & 1 & 0 & \underline{5} \\ 4 & \infty & \infty & 2 & 0 \end{array}$$

$$A^2[3,1] = A^1[3,2] + A^1[2,1]$$
$$= 1 + 2 = 3$$

$$A^2[3,4] = A^1[3,2] + A^1[2,4]$$
$$= 1 + 4 = 5$$

$$A^3 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & & \infty & \\ 2 & & 0 & \infty & \\ 3 & 3 & 1 & 0 & 5 \\ 4 & & & 2 & 0 \end{array} = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 3 & \infty & 5 \\ 2 & 2 & 0 & \infty & 4 \\ 3 & 3 & 1 & 0 & 5 \\ 4 & \underline{5} & \underline{3} & 2 & 0 \end{array}$$

$$A^3[4,1] = A^2[4,3] + A^2[3,1]$$
$$= 2 + 3 = \underline{\underline{5}}$$

$$A^3[4,2] = A^2[4,3] + A^2[3,2]$$
$$= 2 + 1 = \underline{3}$$

$$A^4 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & & & 5 \\ 2 & & 0 & & 4 \\ 3 & & & 0 & 5 \\ 4 & 5 & 3 & 2 & 0 \end{array} = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 3 & \underline{7} & 5 \\ 2 & 2 & 0 & \underline{6} & 4 \\ 3 & 3 & 1 & 0 & 5 \\ 4 & 5 & 3 & 2 & 0 \end{array}$$

$$A^4[1,3] = A^3[1,4] + A^3[4,3]$$
$$5 + 2 = \underline{\underline{7}}$$

$$A^4[2,3] = A^3[2,4] + A^3[4,3]$$
$$= 4 + 2 = \underline{6}$$

The final matrix is

$$A^4 = \begin{bmatrix} 0 & 3 & 7 & 5 \\ 2 & 0 & 6 & 4 \\ 3 & 1 & 0 & 5 \\ 5 & 3 & 2 & 0 \end{bmatrix}$$

MOD-IV

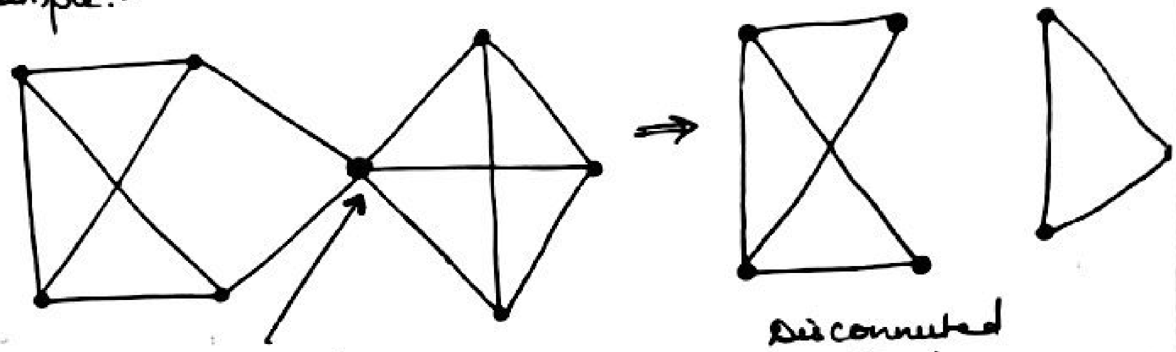
Connectivity and planar graph.

The connectivity of a connected graph G is the minimum number of vertices or edges whose removal makes G disconnects or reduces to a trival graph.

Vertex Connectivity

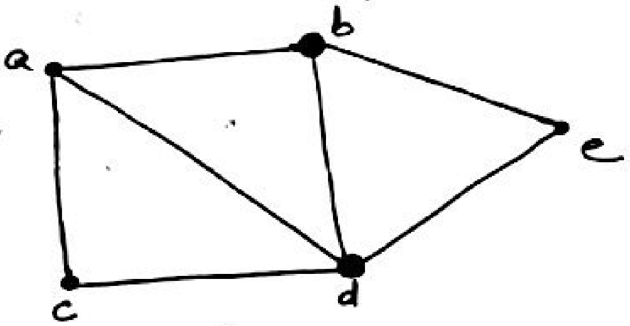
The vertex connectivity of a connected graph G is defined as the minimum number of vertices whose removal from G leaves the remaining graph disconnected.

Example:-

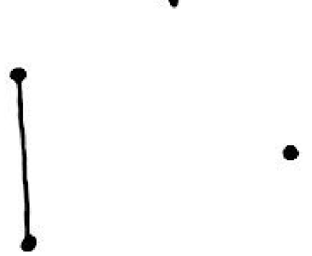


Vertex connectivity = 1

Disconnected graph.



Vertex connectivity = 2 {b, d}



Disconnected graph.

Note:- Vertex connectivity of a tree is 1.

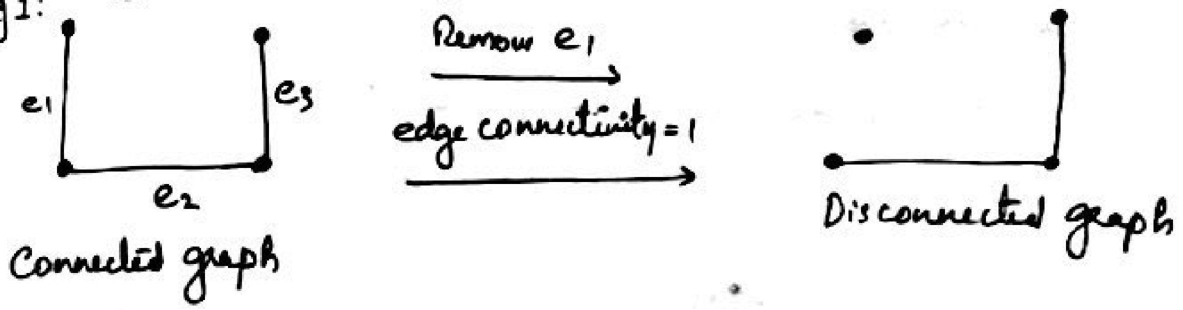
Edge Connectivity

Each cut-set of a connected graph G consists of a certain number of edges. The number of edges in the smallest cut-set is defined as the edge connectivity of G.

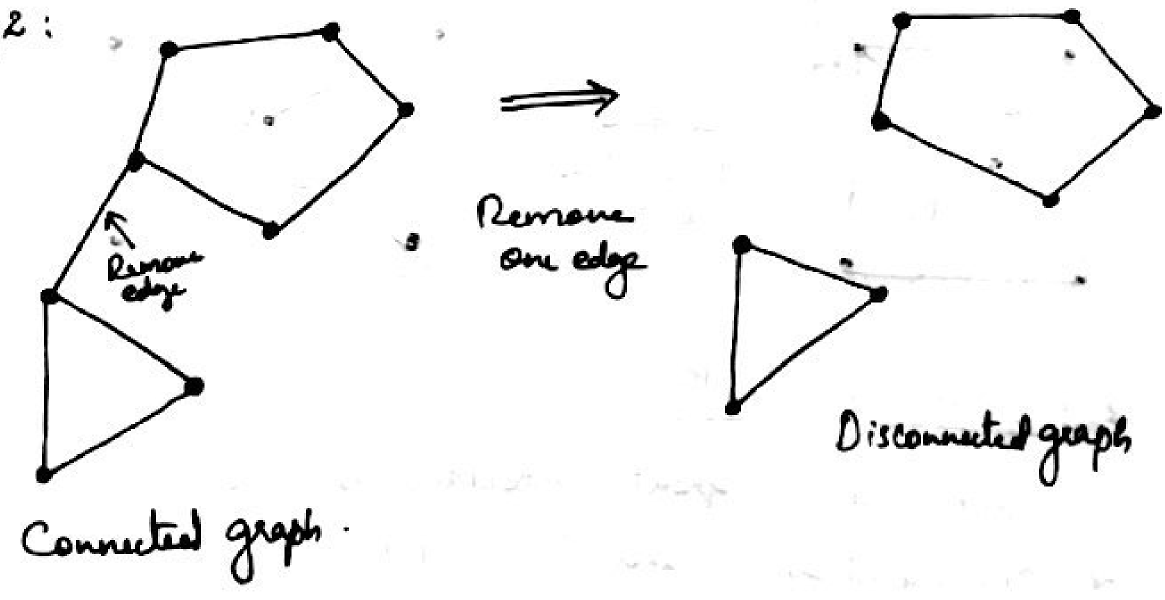
Edge connectivity of a connected graph can be defined as the minimum number of edges whose removal (deletion) leaves the remaining graph disconnected or reduces the rank of the graph by one.

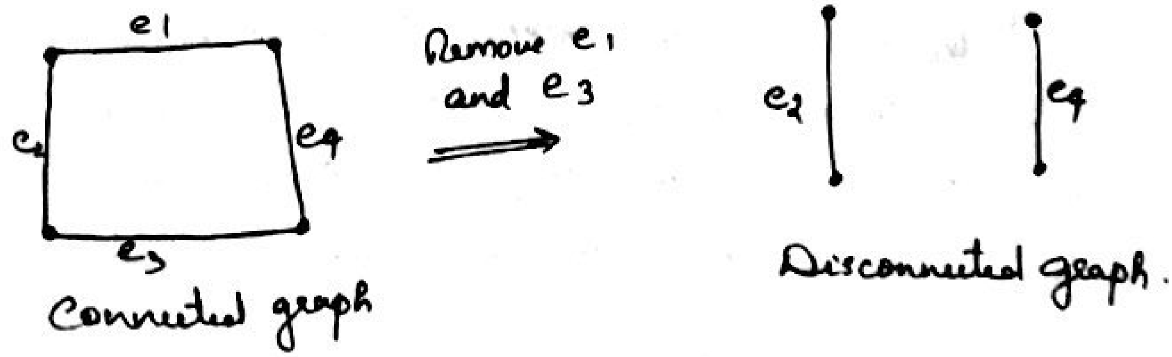
The edge connectivity of a tree is one.

Eg 1:



Eg 2:



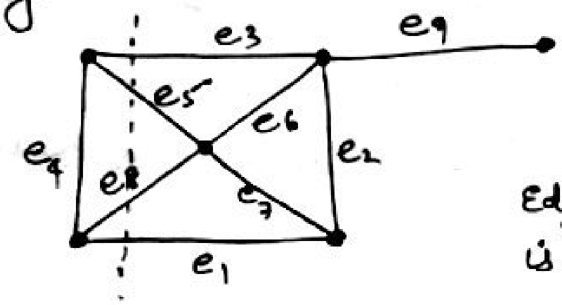


Cut Sets

In a connected graph G, a set of edges whose removal disconnects the graph is called a cut set.

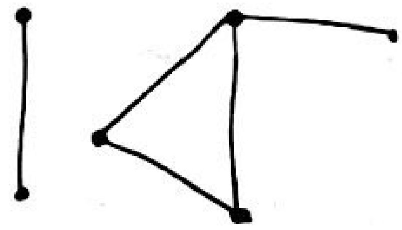
cut set is also called as cocycle or minimal cut set

Eg:-



Connected graph.

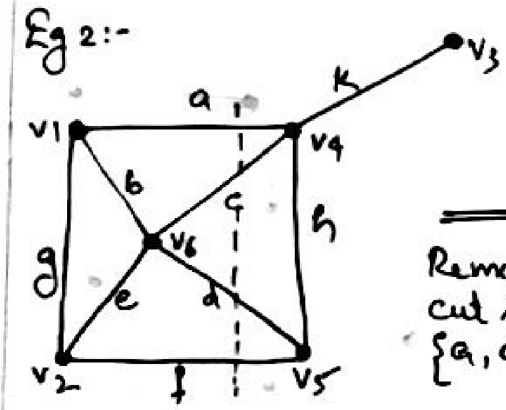
Edges {e3, e5, e8, e1} is a cut set.



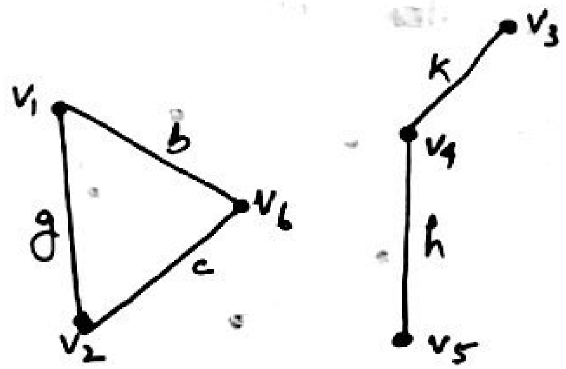
Disconnected graph.

Other cut-sets are {e3, e5, e4}, {e4, e8, e1}, {e3, e6, e7, e1}
{e3, e6, e2}, {e2, e7, e1}, {e9}.

Eg 2:-



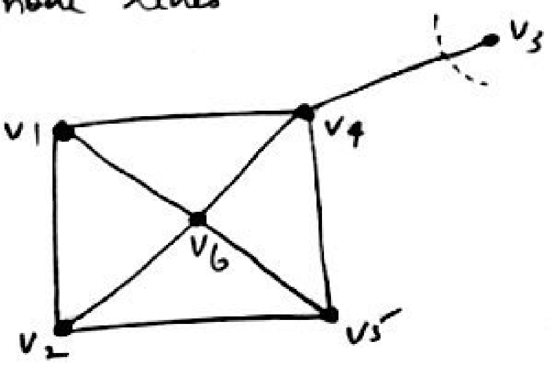
Removal of cut set. {a, c, d, f}



Application of cutsets

Cut set are of great importance in studying properties of communication and transportation network.

for eg. In the following graphs 6 vertices represents the 6 cities connected by telephone line. we need to find out if there are any weak spot in the network that needs strengthening by means of additional telephone lines



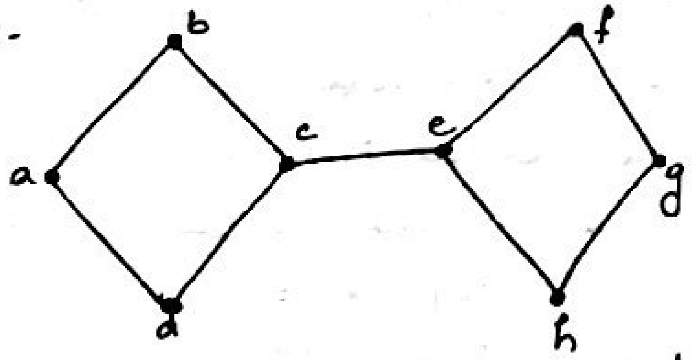
Cut Vertex

Let 'G' be a connected graph. A vertex $V \in G$ is called a cut vertex of 'G'. If 'G-v' results in a disconnected graph. Removing a cut vertex from a graph breaks it in to two or more graphs.

Note :- Removing a cut-vertex may render a graph disconnected

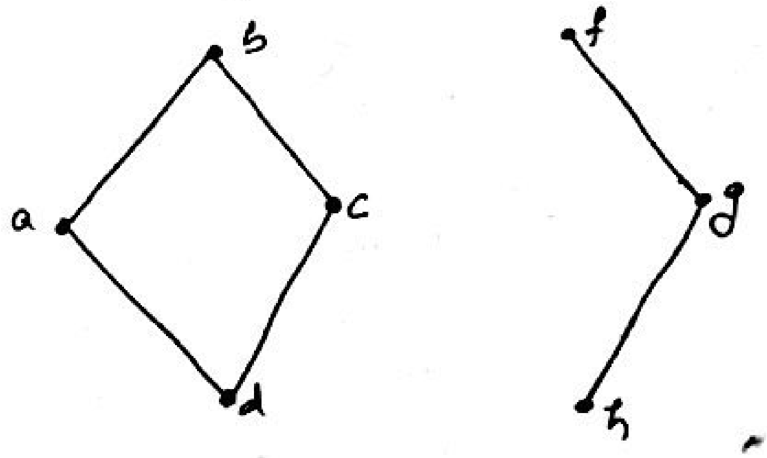
A connected graph 'G' may have at most (n-2) cut vertices

Eg :-



By removing vertex 'c' or vertex 'e' the graph will become a disconnected graph

⇓



Theorem -1 :-

The edge connectivity of a graph G cannot exceed the degree of the vertex with the smallest degree in G.

Proof :-

Let vertex V_i be the vertex with the smallest degree in G. Let $d(V_i)$ be the degree of V_i. Vertex V_i can be seperated from G by removing the $d(V_i)$ edges incident on vertex V_i. Hence the theorem

Theorem - 2 :-

The vertex connectivity of any graph G can never exceed the edge connectivity of G.

Proof :-

Let α denote the edge connectivity of G. Therefore there exists a cutset S in G with α edges. Let S partition the vertices of G into subsets V_1 and V_2. By removing atmost α vertices from V_1 (or V_2) on which the edges in S are incident, we can effect the removal of S (together with all other edges incident on these vertices) from G. Hence the theorem.

Theorem -3 :-

The maximum vertex connectivity one can achieve with a graph G of n vertices 'e' edges ($e \geq n-1$) is the integrated part of the no: $\frac{2e}{n}$ ie $\lfloor 2e/n \rfloor$

Proof :-

Every edge in G contributes 2 degrees. The total 2e degrees is divided among n vertices. ∴ there must be atleast one vertex in G whose degree is equal to or less than the no: $\frac{2e}{n}$.

The edge connectivity G cannot exceed this no: (based on theorem). Thus for a 'n' vertex regular graph.

vertex connectivity $\leq$ edge connectivity $\leq \frac{2e}{n}$

edge connectivity $\leq \frac{2e}{n}$ (degree) and

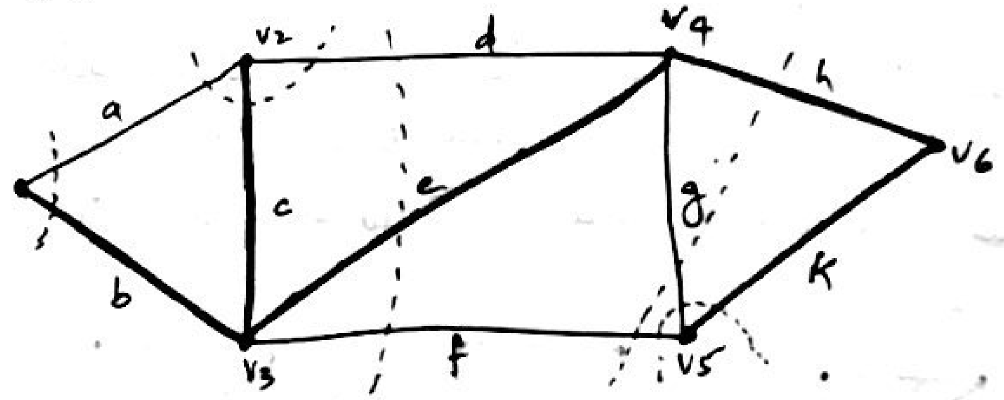
vertex connectivity $\leq$ edge connectivity based on previous theorem)

$\therefore$ maximum vertex connectivity $\leq \frac{2e}{n}$

$\therefore$ Vertex connectivity $= \lfloor \frac{2e}{n} \rfloor$

Fundamental cut set.

Fundamental cut set is a cut set 'S' will contain only one branch b of T, where 'T' is the spanning tree of G. and the rest of the edges in S are chords with respect to T. Such a cut set is called fundamental cut set or basic cut set.



Spanning Tree branches $\{b, c, e, h, k\}$

Chords of the graph $: \{a, d, f, g\}$

Fundamental cutset $: \{(a, b), (a, c, d), (d, e, f), (h, g, f), (f, g, k)\}$

Theorem - 4 :-

Every cut set in a connected graph G must contain atleast one branch of every spanning tree of G.

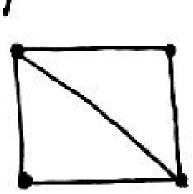
proof :-

In a given connected graph G, let Q be the minimal set of edges containing atleast one branch of every spanning tree of G. Consider G-Q the subgraph [illegible] that remains after removing the edges in Q from G. Since the subgraph G-Q contains no spanning tree of G, G-Q is disconnected (one component of which may just consist of an isolated vertex). Also, since Q is a minimal set of edges with this property. and edge e from Q returned to G-Q will create at atleast one spanning tree. Thus the subgraph G-Q+e will be a connected graph. Therefore Q is a minimal set of edges whose removal from G disconnect G. This by definition is a cut set. Hence the theorem.

Fundamental circuits :-

Fundamental circuit is a circuit formed by adding a chord of G to a spanning tree T such that it creates only one circuit is called a fundamental circuit

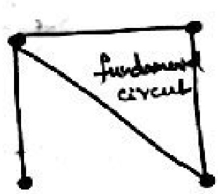
Example .



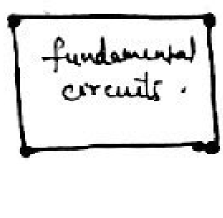
G .

T

⇒



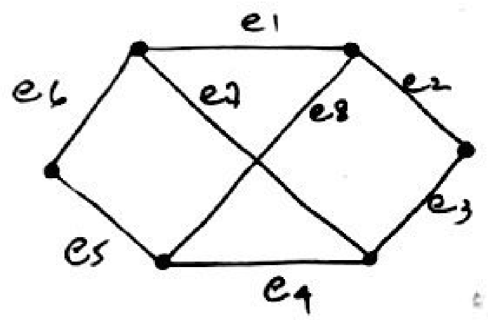
⇓



Note:-

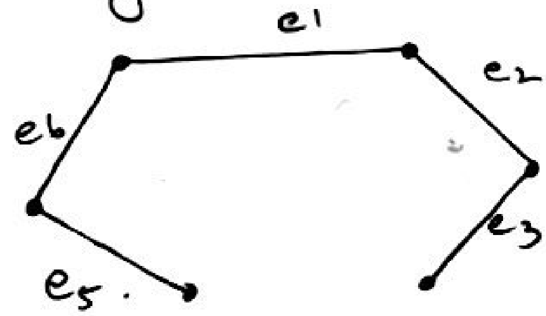
Fundamental circuits number of a graph is equal to the no: of chords in the graph.

Q. Determine fundamental circuits of the following given graph.



(G)

Spanning Tree

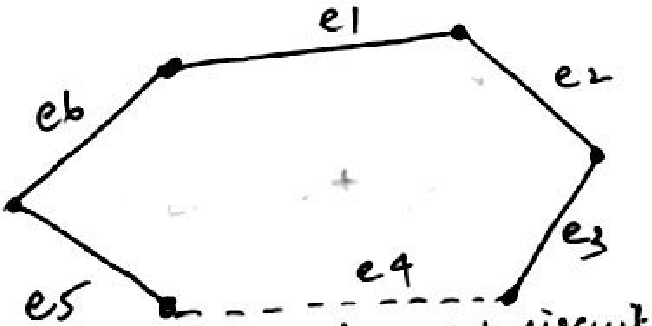


(T)

Chords of the given graph is {e4, e7, e8}.

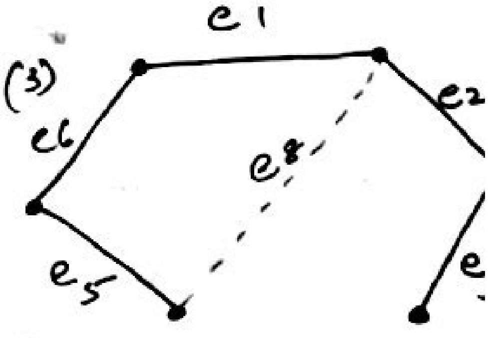
fundamental circuits are.

(1)



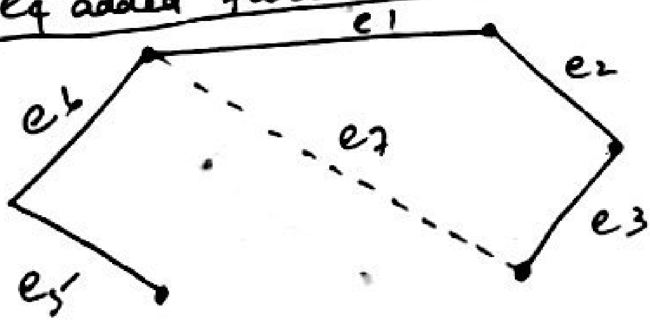
e4 added fundamental circuit

(3)



e8 added fundamental circuit

(2)

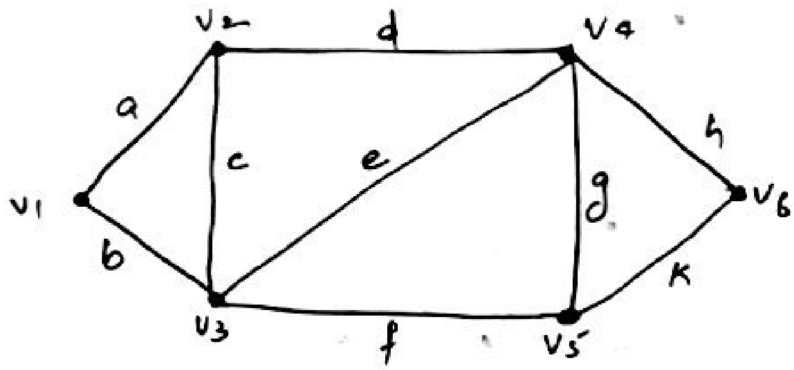


e7 added fundamental circuit

Theorem-5:-

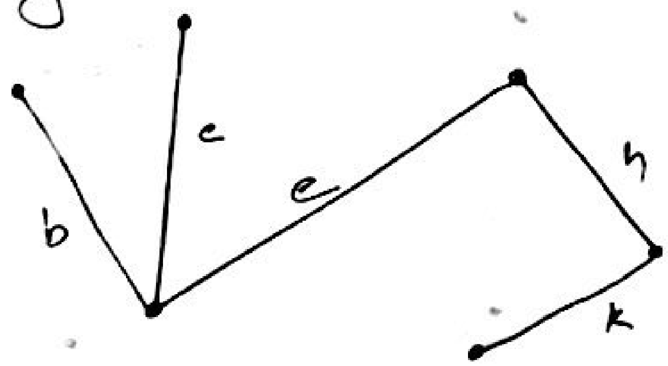
With respect to the given spanning tree T, a chord C_i that determines a fundamental circuit Γ occurs in every fundament cut set associated with the branches in Γ and in no other

As an example consider the spanning tree {b, c, e, h, k} ~~sto~~. of the following given graph.



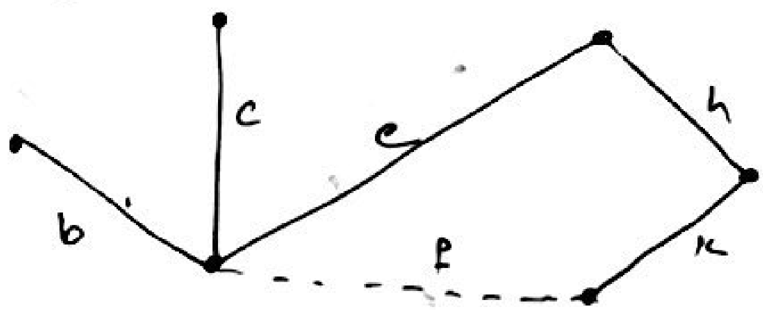
(G)

Spanning Tree.



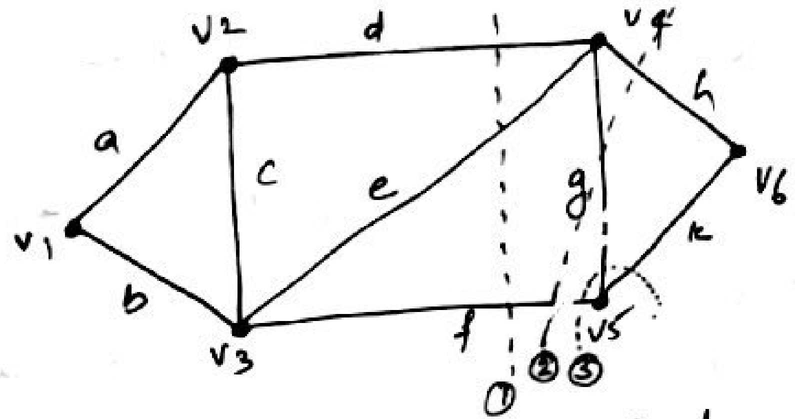
(T)

The fundamental circuit made by the chord f is.
{f, e, h, k}.



The fundamental cut-sets determined by the three branches e, h and k are

determined by branch e : {d, e, f} ①
determined by branch h : {f, g, h} ②
determined by branch k : {f, g, k} ③

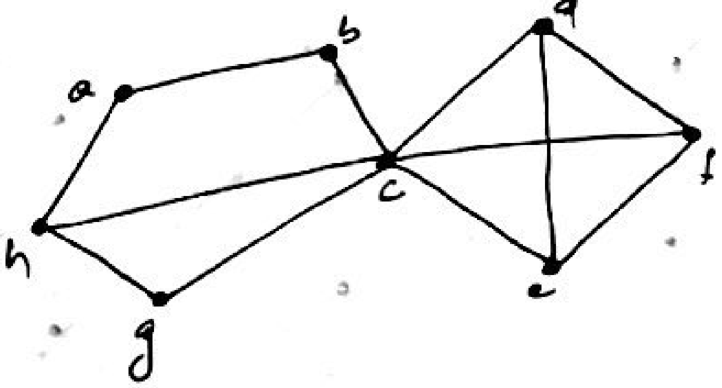


Chord f occurs in each of these fundamental cut-sets and there is no other fundamental Cut set that Contains f. The converse of the Theorem is also true.

Seperable graph

A Connected graph is said to be seperable if its vertex Connectivity is one. All other connected graphs are Called Non seperable.

Eg:-



vertex connectivity = 1
{c}

Planar Graphs

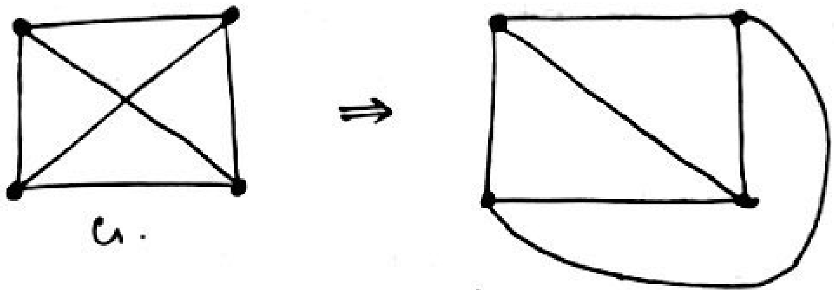
A graph G is said to be planar if there exists some geometric representation of G which can be drawn on a plane such that no two of its edges intersect with eachother.

The geometric representation of planar graph is also called embedding.

An embedding of a planar graph G on a plane is called a plane representation of G.

A graph that cannot be drawn on a plane without a crossover between edges is called non planar

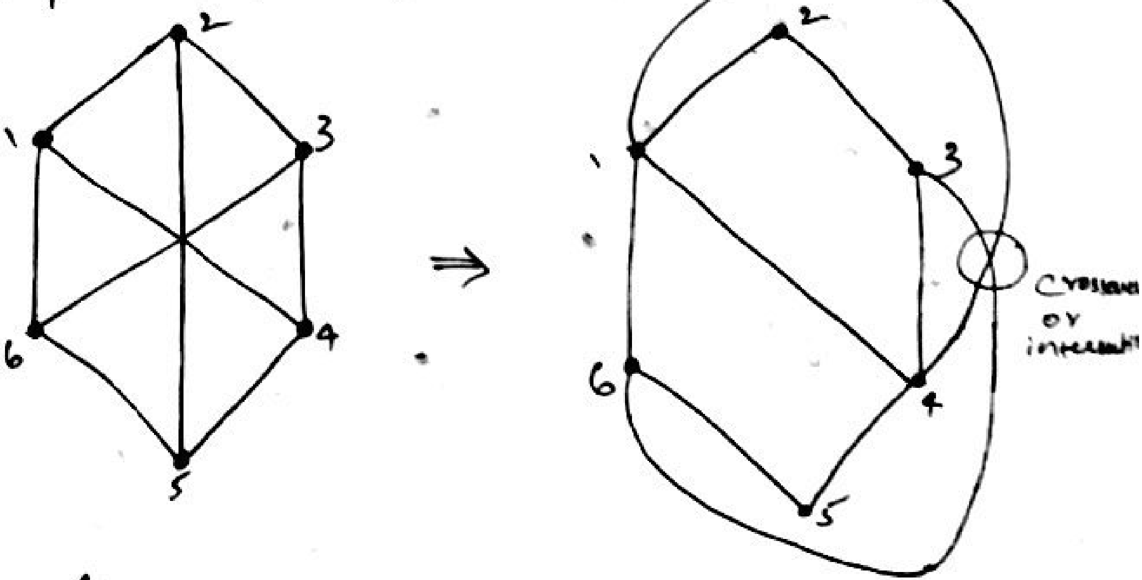
Eg:



G.

planar representation of G

Non planar Graph example.



G

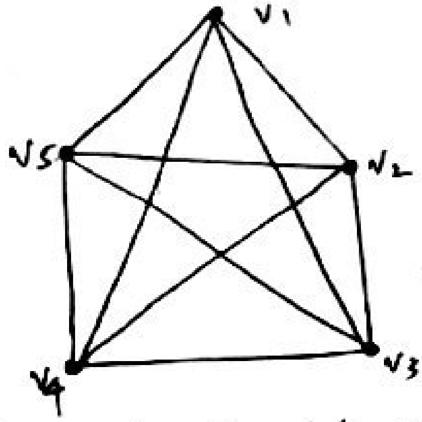
planar representation of G

Kuratowski's Graph

Kuratowski's Graph Consists of two specific non planar graph.

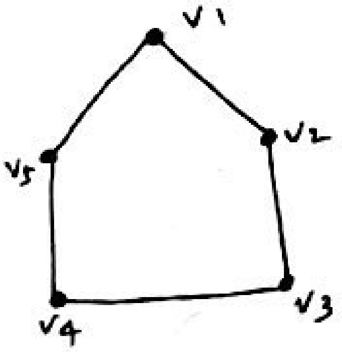
Kuratowski's First Graph.

The first graph of Kuratowski is a complete graph with 5 vertices and 10 edges it is denoted by K_5



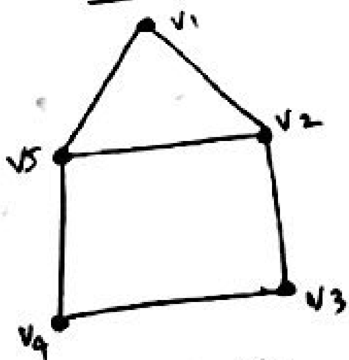
Q Prove that the Complete graph of 5 vertices is non-planar or P.T Kuratowski's first graph is non-planar or P.T K_5 is non-planar

Step 1



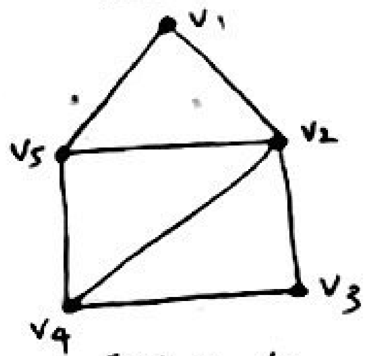
5 vertices.

step 2



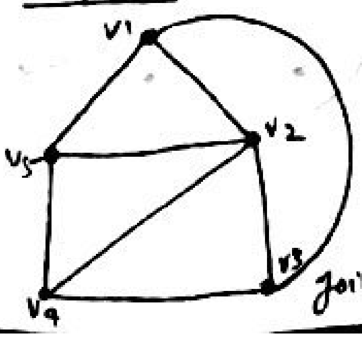
Join v2 - v5

Step - 3



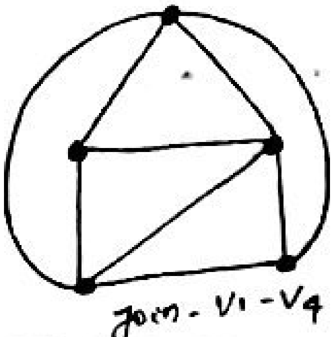
Join v2 - v4

Step 4



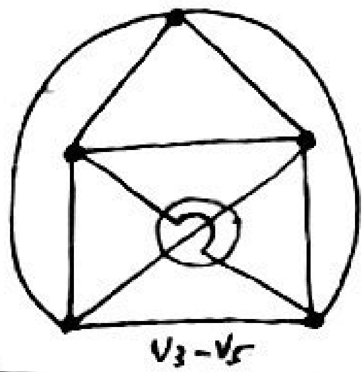
Join v1 - v3

Step-5



Join - v1 - v4

Step - 6



V3 - V5

The edge $v_3 - v_5$ Cannot be drawn Without Crossover.

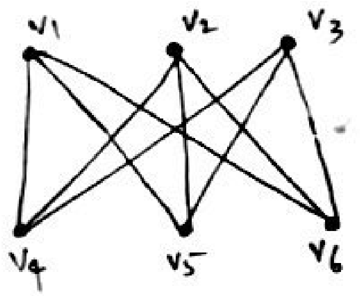
$\therefore$ The graph Cannot be embedded in a plane.

$\therefore$ the complete graph of 5 vertices, ie K_5 is non-planar.

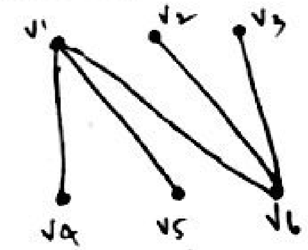
<u>Kuratowski second graph.</u>

The second graph of Kuratowski is a <u>Regular Connected graph with 6 vertices and 9 edges.</u>

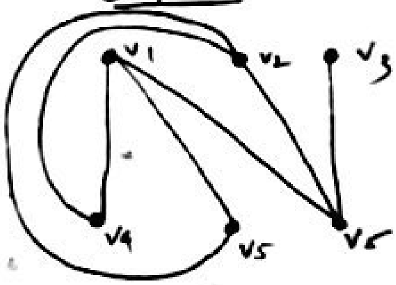
It is denoted by $K_{3,3}$



<u>Step 1</u>

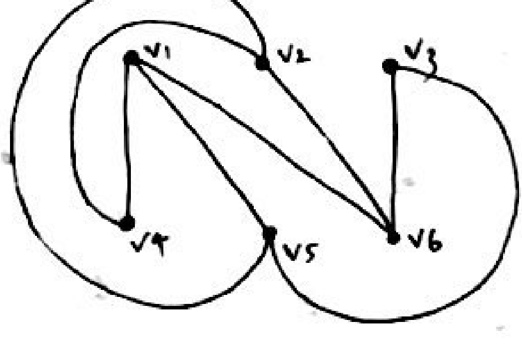


<u>Step-2</u>



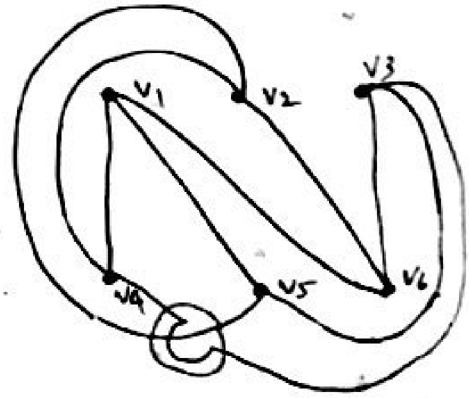
Join $v_2 - v_4$ & $v_2 - v_5$

<u>Step - 3</u>



Join $v_3 - v_5$

<u>Step - 4</u>



$v_3 - v_4$ Cannot be drawn Without Crossover

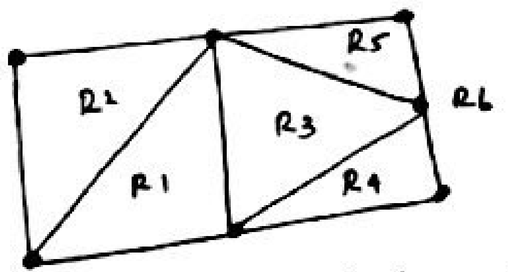
So the graph cannot be embedded on a plane ∴ the kuratowski's second graph is non-planar.

Properties of kuratowski graph.

* Both are Regular graph
* Both are non planar
* Removal of one edge or one vertex makes each a planar graph.
* kuratowski's first graph is the non planar graph with the smallest number of vertices and
* kuratowski's second graph is the non planar graph with the smallest number of edges. Thus both are the simplest non planar graphs.

Regions

Every planar graph divides the plane into connected areas called Regions.



R6 is called Infinite Region - Region lying outside a graph.

Degree of a bounded Region r : deg(r) = No: of edges enclosing the Regions r.

Example : in the above graph.

Deg(R1) = 3
Deg(R2) = 3
Deg(R3) = 3
Deg(R4) = 3
Deg(R5) = 3
Deg(R6) = 7

Sum of degrees of all Region = 3+3+3+3+3+7 = 22

In a planar graph with n regions, sum of degrees of region is $2|E|$

No: of edges = 11.

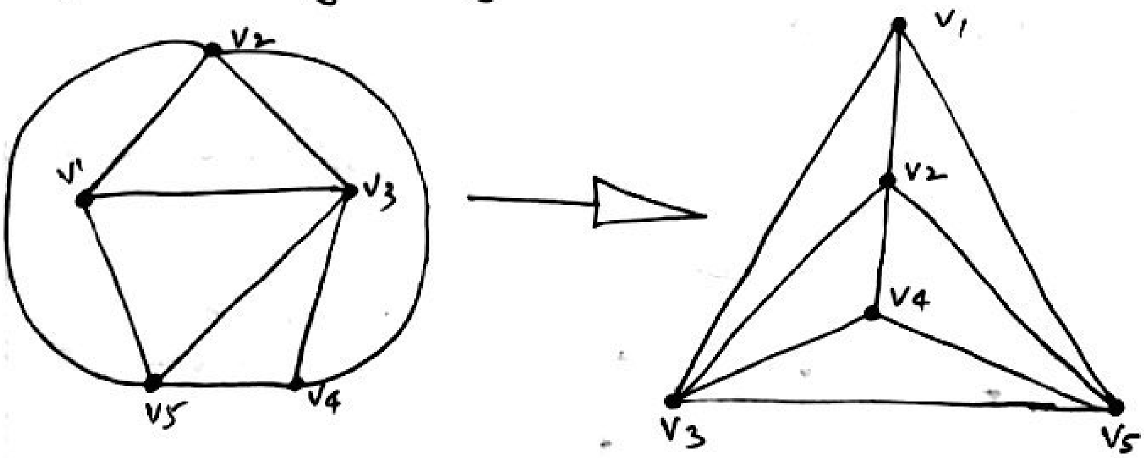
$\therefore$ Sum of degrees of Region = $2 \times 11 = \underline{22}$

Different Representation of planar graph

Straight Line Representation :-

Any simple planar graphs can be embedded in a plane such that every edge is drawn as a straight line segment.

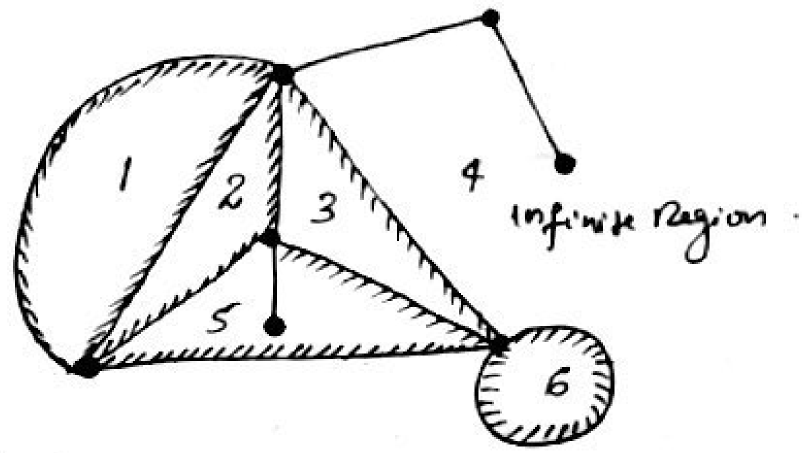
This representation is only used for simple graph because self loop and parallel edges cannot be represented by straight line



Plane Representation :-

A plane representation of a graph divides the plane into regions (also called windows, faces or meshes). A region is characterized by the set of edges (or the set of vertices) forming its boundary.

Infinite Region : The portion of the plane lying outside a graph embedded in a plane. Such a region is called the infinite, unbounded, outer or exterior region for that particular plane representation.



Euler's Theorem

A connected planar graph with n vertices and e edges has $e-n+2$ regions.

We can disregard a self loop or parallel edges because it simply adds one region to the graph and simultaneously increases the value of e by one. We can also disregard all edges that do not form boundaries of any region. Adding of any such edge increase (or decrease) e by one and increases n by one, keeping the quantity $e-n$ unaltered.

Since any simple planar graph can have a plane representation such that each edge is a straight line, any planar graph can be drawn such that each region is a polygon (a polygonal net).

Let the polynomial net representing the given graph consist of f regions or faces and let k_p be the number of p-sided regions

If we add self loop or parallel edges it added region

Calculating the no: of edges.

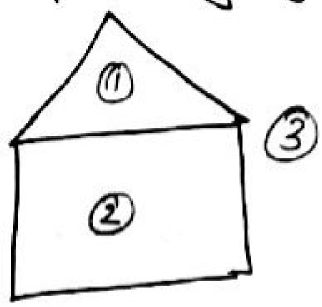
Let k_3 = no: of triangles.
k_4 = no: of quadrilaterals
k_5 = no: of pentagons
⋮
k_r = no: of r-sided polygons.

Hence total no: of edges in the entire net of polygon would be

$$3k_3 + 4k_4 + 5k_5 \cdots + rk_r + p = 2 * e$$

Where 'e' is the total no: of edges in the polyhedron

Eg. Consider the following graph.



It consists of 3 Regions and the no: of edges.

$$= 3k_3 + 4k_4 + p$$
$$= 3\times1 + 4\times1 + 5 = 12 = 2\times6 = 12$$

Calculating the no: of faces.

Total no: of faces/Regions would be $k_3 + k_4 + k_5 + \cdots k_r + 1 = f$.

The outer Region/infinite Region is also to be counted along with all interior Regions, hence the '1'

'f' is the total no: of Regions/faces

Sum of all interior angles of the polyhedron.

Sum of all interior angles of a p-sided polygon is $(p-2)\pi$

Taking the sum of interior angles for each polygon inside the polyhedron,

$k_3(3-2)\pi + k_4(4-2)\pi + k_5(5-2)\pi + \cdots + k_r(r-2)\pi$

Sum of all exterior angles of the polyhedron

Sum of all exterior angle of p-sided polygon is $(p+2)\pi$

Total angle sum of the polyhedron

At each vertex we have an angle of $360°$

Since we have n vertices Total angle sum $= n * 360°$

$= n * 2\pi = \underline{2\pi n}$

$\boxed{\text{Sum of interior angles + Sum of exterior angles = total angle sum of polyhedron.}}$

$$k_3(3-2)\pi + k_4(4-2)\pi + k_5(5-2)\pi + \cdots + k_r(r-2)\pi + (p+2)\pi = 2\pi n$$

$$\pi\left[(3k_3 - 2k_3) + (4k_4 - 2k_4) + (5k_5 - 2k_5) + \cdots rk_r - 2k_r\right] + (p+2)\pi = 2\pi n$$

$$\pi(3k_3 + 4k_4 + 5k_5 + \cdots + rk_r) - 2k_3\pi - 2k_4\pi - 2k_5\pi \cdots - 2k_r\pi + p\pi + 2\pi = 2\pi n$$

$$\pi(2e - p) - 2\pi(k_3 + k_4 + k_5 + \cdots k_r) + p\pi + 2\pi = 2\pi n$$

$$\pi(2e - p) - 2\pi(f-1) + \pi(p+2) = 2\pi n$$

Cancelling π on both sides.

$$(2e - p) - 2(f-1) + (p+2) = 2n$$

$$2e - \cancel{p} - 2f + 2 + \cancel{p} + 2 = 2n$$

$$2e - 2f + 4 = 2n$$

Cancelling 2 on both sides

$$e - f + 2 = n \qquad \therefore f = \underline{e - n + 2}$$

Theorem 6:-

In any simple graph, connected planar graph with f regions, n vertices and e edges ($e > 2$) the following inequality holds.

$$e \geq \frac{3}{2} f$$

$$e \leq 3n - 6.$$

Proof:-

Since each region is bounded by atleast 3 edges or sides each edge belongs to exactly 2 region

$$2e \geq 3f$$ → Total no. of faces/region

Total edge

$$e \geq 3/2 \, f \qquad f = e - n + 2$$

$$e \geq 3/2 (e - n + 2)$$

$$2e \geq 3e - 3n + 2$$

$$e \leq 3n - 6.$$

Q. Prove that Kuratowski's first graph K_5 is non-planar. using inequality equation

K_5 is complete graph with five vertices.

$$n = 5, \; e = 10, \; 3n - 6 = 9 < e$$

Thus the graph violates the inequality and hence it is not planar

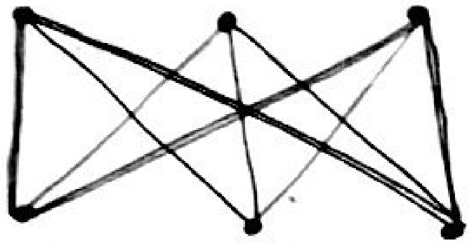
$$e \leq 3n - 6.$$

$$10 \leq 15 - 6$$

$10 \leq 9$ inequality does not exist

Q. Prove that Kuratowski's second graph $K_{3,3}$ is non-planar using inequality equation.

K_6 is regular connected graph with 6 vertices and 9 edges. To prove the non planarity of kuratowski second graph $K_{3,3}$ we can make use of additional fact that no regions in this graph can be bounded with fewer than four edges.



Hence if this graph were planar, we would have

$$2e \geq 4f$$

$$2e \geq 4(e-n+2)$$

$$2*9 \geq 4(9-6+2)$$

$$18 \geq 4(5)$$

$18 \geq 20$ this inequality does not hold.

Geometric Dual

Graph is divided into Region and many regions is adjacent ie two regions share a common edge for each region we denote a point inside and outside of the region.

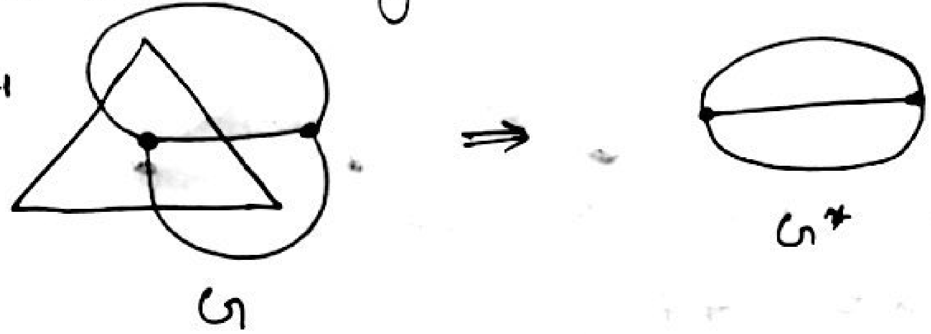
If two regions F_i and F_j are adjacent (ie they have a common edge). draw a line joining points p_i and p_j that intersects the common edge between F_i and F_j exactly once.

If there is more than one edge common between F_i and F_j draw one line between points p_i and p_j for each of the common edges.

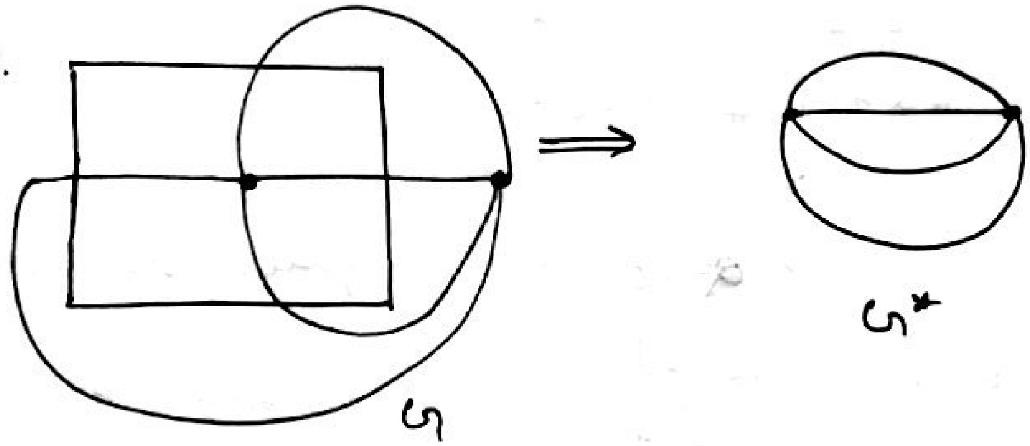
For an edge e lying entirely in one region, draw a self loop at point p_k intersecting e exactly once.

By this procedure we obtain a new graph G^* consisting of vertices and edges joining these vertices. Such a graph G^* is called a dual or geometric dual of G.

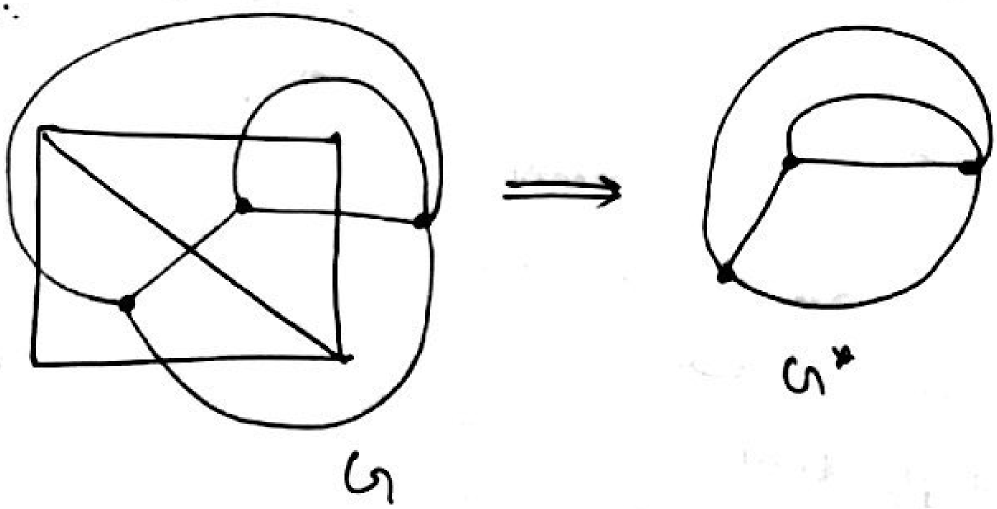
Eg:1

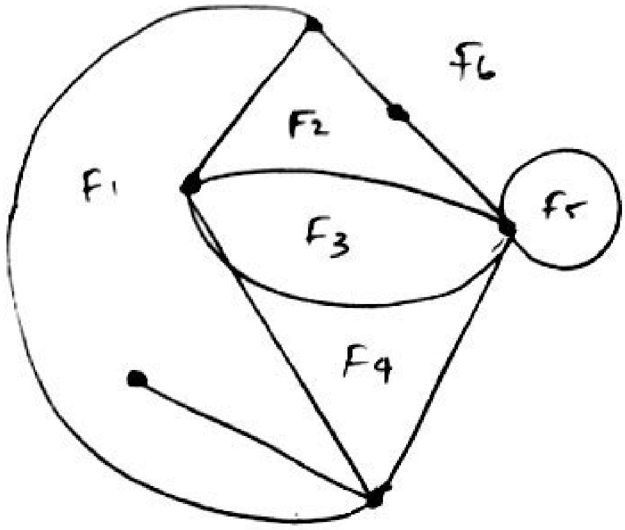


Eg 2.

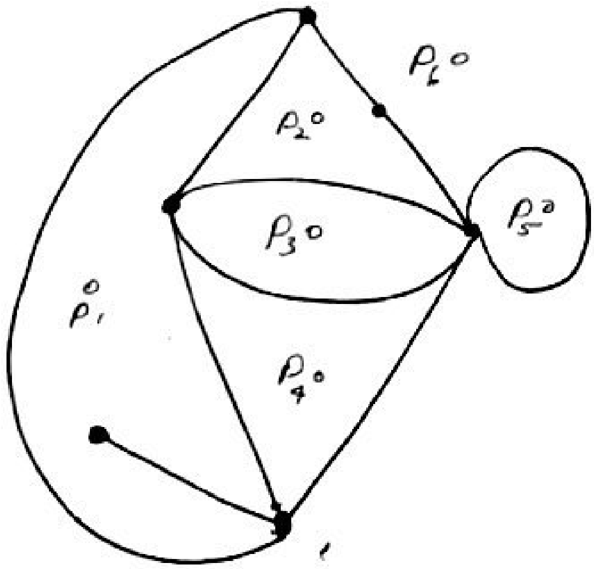


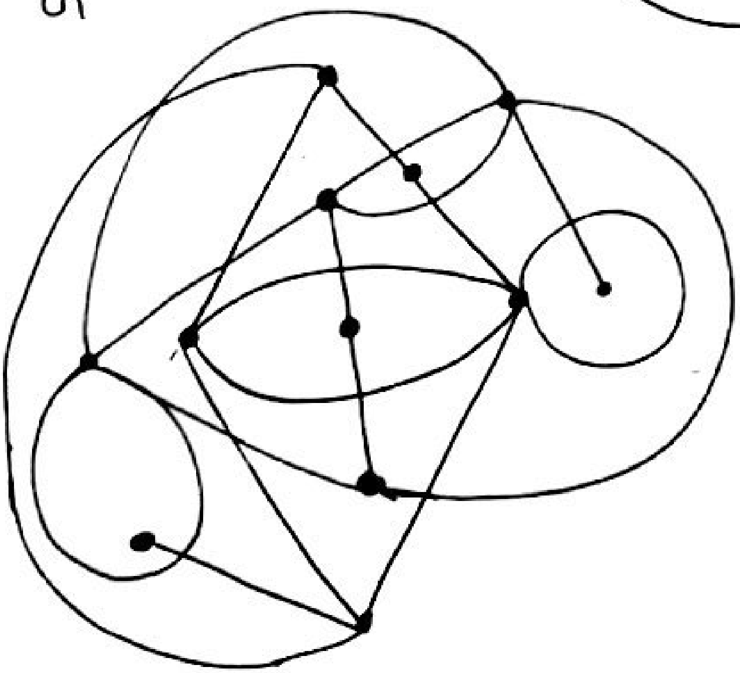
Eg 3.

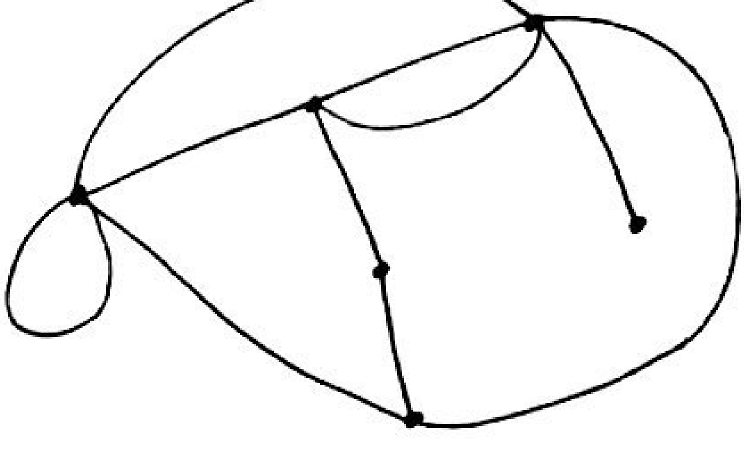




G







G*

Graph Representation and Vertex coloring

Matrix Representation of Graph

A matrix is a convenient and useful way of representing a graph. Expressing the graph in matrix form are widely used in many applications like electrical Network analysis, operation research etc.

Incidence matrix

Let G be a graph with 'n' vertices 'e' edges and no self loops, define n x e matrix

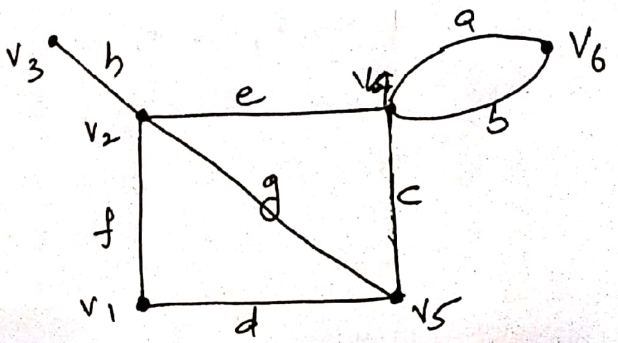
$$A = [a_{ij}]$$

where 'n' rows corresponds to 'n' vertices and 'e' columns corresponds to e edges as

$$a_{ij} = \begin{cases} 1 & \text{if } j^{th} \text{ edge is incident with } i^{th} \text{ vertex } v_i \\ 0 & \text{otherwise.} \end{cases}$$

Such a matrix is called incidence matrix
This incidence matrix contain only two elements 0 and 1. Such a matrix is also called a binary matrix or (0,1) matrix

Incidence matrix for a graph G is denoted by A(G)
A graph and its incidence matrix is shown below



$$A(G) = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{array} \begin{array}{c} \begin{array}{cccccccc} a & b & c & d & e & f & g & h \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

Observations

* Since every edge is incident on exactly 2 vertices, each column of $A(G)$ has exactly 2 ones
* The no: of 1's in each row is equal to the degree of the corresponding vertex.
* A row with all 0's represent an isolated vertex
* Parallel edges in a graph produce identical columns in its incidence matrix ie colum 1 and 2.

Adjacency Matrix

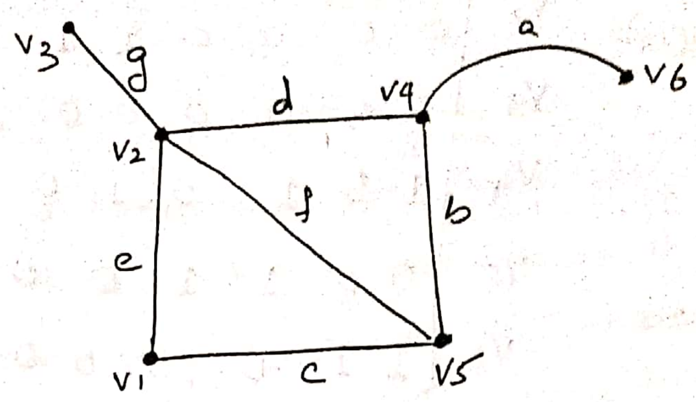
The adjacency matrix of a graph G with n vertices, e edges and no parallel edges is an $n \times n$ symmetric binary matrix.

$X = [x_{ij}]$, where

$$x_{ij} = \begin{cases} 1, & \text{if there is an edge between } v_i \text{ \& } v_j \\ 0, & \text{if there is no edge between } v_i \text{ \& } v_j \end{cases}$$

The adjacency matrix for a graph G is denoted by $X(G)$.

A graph G and its adjacency matrix is shown below



$$
\begin{array}{c}
 \\ V_1 \\ V_2 \\ V_3 \\ V_4 \\ V_5 \\ V_6
\end{array}
\begin{array}{c}
\begin{array}{cccccc} V_1 & V_2 & V_3 & V_4 & V_5 & V_6 \end{array} \\
\begin{bmatrix}
0 & 1 & 0 & 0 & 1 & 0 \\
1 & 0 & 1 & 1 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 1 \\
1 & 1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0
\end{bmatrix}
\end{array}
$$

Observations

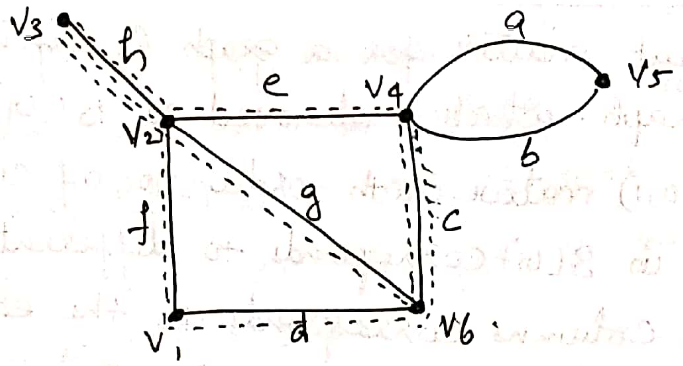
* The entries along the principle diagonal of $X(G)$ are all 0's iff the graph has no self loops.
  A self loop at the i^{th} vertex correspond to $X_{ii} = 1$
* If the graph has no self loops, the degree of a vertex equals the no: of 1's in the corresponding row or column of x.
* The definition of adjacency matrix makes no provision for parallel edges. This is why the adjacency matrix x is defined for graphs without parallel edges.

Path Matrix

A path matrix is defined for a specific pair of vertices in a graph i.e. (x, y) and it is written as $p(x, y)$.
The rows in $p(x, y)$ corresponds to different paths between vertices x and y.
The path matrix of the (x, y) vertices is $p(x, y) = [p_{ij}]$ where

$$p_{ij} = \begin{cases} 1 & \text{if the } j^{th} \text{ edge lies in } i^{th} \text{ path} \\ 0 & \text{otherwise} \end{cases}$$

It is (0,1) matrix in the order no: of path x no: of edges.
A graph and one of its path matrix is shown below.



Consider all paths between vertices V_3 & V_4, there are 3 different paths they are $\{h, e\}$, $\{h, g, c\}$, $\{h, f, d, c\}$.
It is denoted by path 1, 2 and 3.

So

$$p(V_3, V_4) = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{cccccccc} a & b & c & d & e & f & g & h \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \end{array}$$

Observation

* A colum of all 0's corresponds to an edge that does not lie in any path x and y.
* A column of all 1's corresponds to an edge that lie in every path between x and y.
* There is no row with all 0's
* The no: of 1's in a row is equal to the no: of edges in the corresponding path.
* The ring sum of any two rows in $p(x,y)$ corresponds to a circuit or an edge disjoint of circuits.

Circuit Matrix

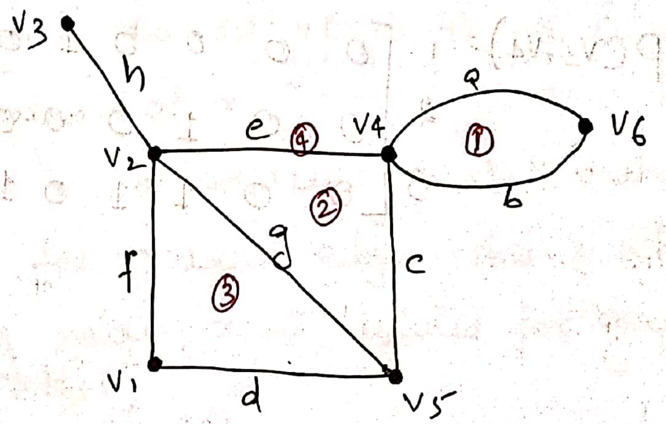
A circuit matrix for a graph G defines no: of circuits in a graph which is denoted as $B(G)$

It is (0,1) matrix with order no: of circuits x no: of edges ie row in $B(G)$ corresponds to different circuits in G and the columns corresponds to the edges in G.

Circuit matrix for G is $B(G) = [b_{ij}]$

where $b_{ij} = \begin{cases} 1 & \text{if } j^{th} \text{ edge lie in the } i^{th} \text{ circuit} \\ 0 & \text{otherwise} \end{cases}$

A graph G and its circuit matrix is shown below



The above graph consists of 4 circuits namely $\{a,b\}$, $\{e,c,g\}$, $\{f,g,d\}$, $\{c,d,f,e\}$ and it can be numbered as 1, 2, 3 and 4 respectively

The circuit matrix is $\underset{\text{Circuit}}{4} \times \underset{\text{edges}}{8}$ matrix

$$B(G) = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccccccc} a & b & c & d & e & f & g & h \end{array} \\ \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \end{bmatrix} \end{array}$$

Observations

* A column of all 0's corresponds to a non-circuit edge ie one edge that does not belong to any circuit.
* Each row of $B(G)$ is a circuit vector
* The no: of 1's in a row is equal to the no: of edges in the corresponding circuit.
* The circuit matrix is capable of representing a self loop. the corresponding row will have a single one.
* If a graph is disconnected and consists of 2 components g_1 and g_2. The circuit matrix $B(G)$ can be written as

$$B(G) = \left[\begin{array}{c:c} B(g_1) & 0 \\ \hdashline 0 & B(g_2) \end{array}\right]$$

where $B(g_1)$ & $B(g_2)$ are circuit matrix of g_1 and g_2.

* Permutation of any 2 rows or columns simply corresponds to relabelling the circuits and edges.
* Two graph G_1 and G_2 will have the same circuit matrix iff G_1 and G_2 are isomorphic

Graph Colouring

We are given a graph G with n vertices and are asked to color its vertices such that no two adjacent vertices have the same color. The coloring problem constitutes the minimum number of colors required to color a given graph.

After coloring the vertices, we can group the vertices into different sets - each set consisting of individual colors. This is a partitioning problem.

Graph color Applications

Some important applications of graph coloring are as follows.

- Map coloring
- Scheduling the tasks
- Preparing Time Table
- Assignment.
- Conflict Resolution
- Sudoko.

Chromatic Number.

Painting all the vertices of a graph with colors such that no two adjacent vertices have the same color is called the proper coloring of a graph. A graph in which every vertex has been assigned a color according to proper coloring is called a properly colored graph. Usually a given graph can be properly colored in many different ways. The following figure shows three different proper coloring of a graph.

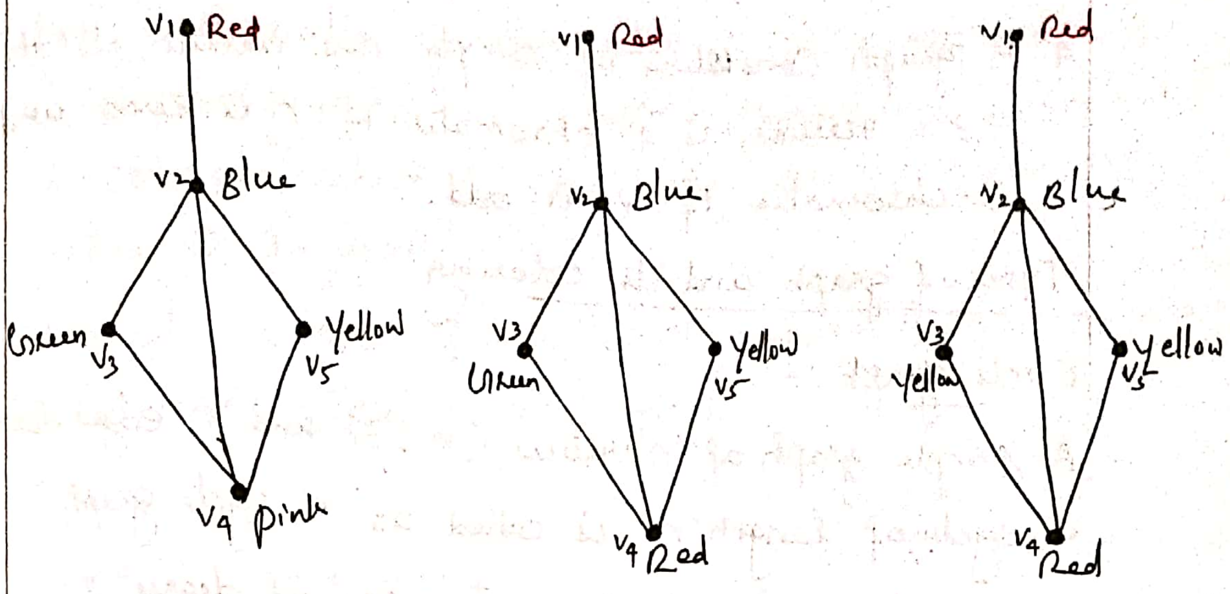


fig: proper coloring of a graph.

The proper coloring which is of interest to us is one that requires the minimum number of colors. A graph G that requires k different colors for its proper coloring, and no less is called k-chromatic graph and the number k is called the chromatic number of G.

The above given graph is 3-chromatic

Note:-
For coloring problems only simple, connected graphs are to be considered.

Observations in coloring a graph.

1. A graph consisting of only isolated vertices is 1-chromatic
2. A graph with one or more edges (not self loop) is atleast 2-chromatic
3. A complete graph of n vertices is 'n' chromatic as all its vertices are adjacent. Hence a graph containing a complete graph of 'r' vertices is atleast r-chromatic
Every graph having a triangle is at least 3-chromatic

4. A graph consisting of simply one circuit with $n \geq 3$ vertices is 2 chromatic if n is even and 3-chromatic if n is odd.

Types of graph and its coloring

Cycle graph:-

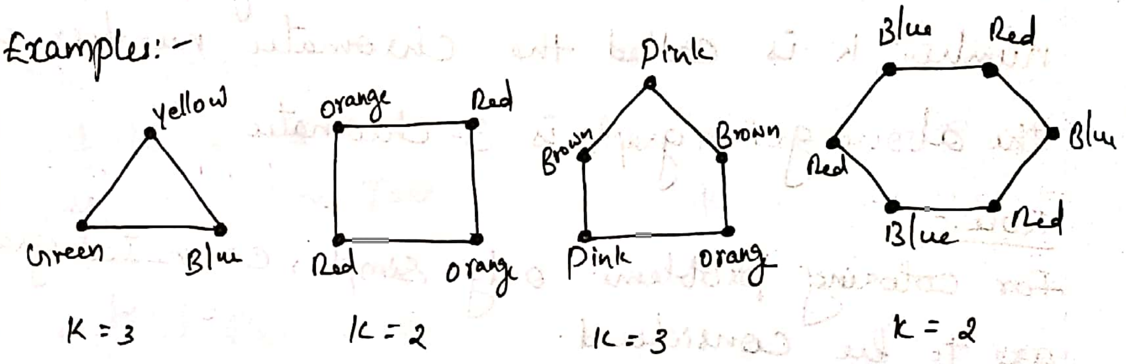
A simple graph of 'n' vertices ($n \geq 3$) and 'n' edges forming a cycle of length 'n' is called as a cycle graph.

In a cycle graph, all the vertices are of degree 2.

If the number of vertices in cycle graph is even, then its chromatic number is 2.

If the number of vertices in cycle graph is odd, then its chromatic number is 3.

Examples:-



Planar Graph:-

A planar graph is a graph that can be drawn in a plane such that none of its edges cross each other.

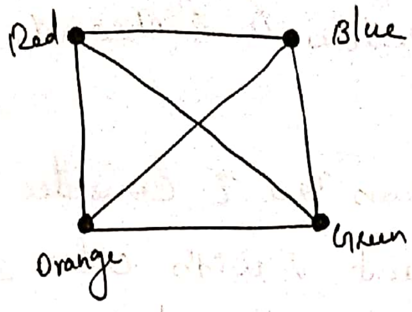
Chromatic number of any planar graph equal to or less than the no: of edges cross each other

Complete graph:-

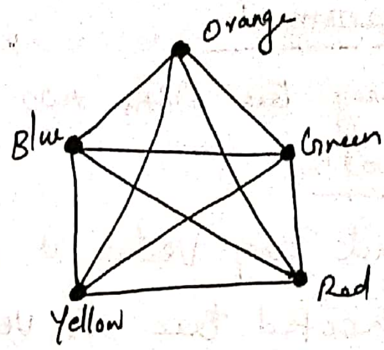
A complete graph is a graph in which every two distinct vertices are joined by exactly one edge. In a complete

graph, each vertex is connected with every other vertex.
So to properly color a complete graph the number of colored needed is equal to the number of vertices in the graph.
Chromatic Number of any complete graph is equal to the number of vertices in that graph.

Eg:-



K=4

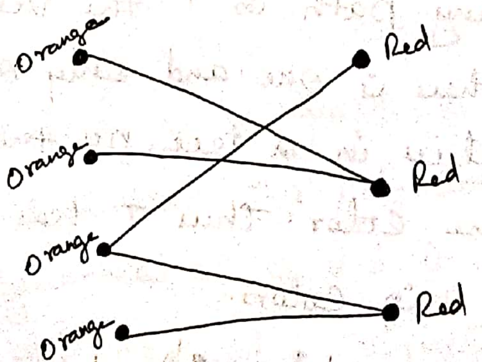


K=5

Bipartite Graphs

A Bipartite graph consists of two sets of vertices X and Y. The edges only join vertices in X to vertices in Y, not vertices within a set.
Chromatic Number of any Bipartite Graph = 2

Eg:



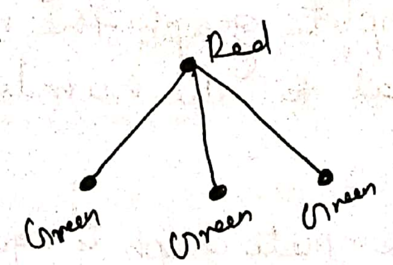
K=2.

Trees:-

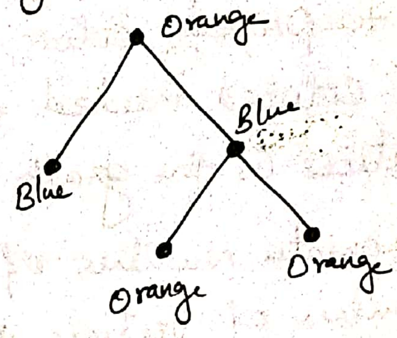
A Tree is a special type of connected graph in which there are no circuits.
Every tree is a bipartite graph.

So Chromatic number of any tree = 2.

Eg:



$K=2$



$K=2$.

<u>Theorem - I</u>

Every tree with two or more vertices is 2-chromatic

<u>Proof:-</u>

Select any vertex v in the given tree T. Consider T as a rooted tree at vertex v. paint v with color 1. paint all vertices adjacent to v with color 2. Next paint the vertices adjacent to then using color 1. Continue this process till every vertex in T has been painted. Now in T, we find that all vertices at odd distances from v have color 2 while v and vertices at even distances from v have color 1.

Now along any path in T the vertices are of altering colors. Since there is one and only one path between any two vertices in a tree, no two adjacent vertices have the same color. Thus T has been properly colored with two colors.

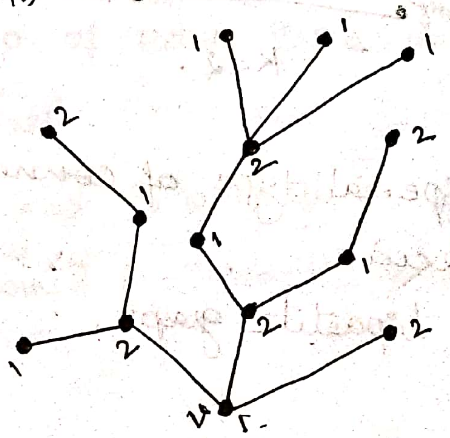


fig: proper coloring of tree.

Theorem - II

A graph with atleast one edge is 2 chromatic if and only if it has no circuit of odd length.

Proof:-

Let G be a connected graph with circuits of only even lengths. consider a spanning tree T in G. Using the color procedure and the chromatic color of a tree theorem the proper coloring of T can be done with two colors. Now add the chords to T one by one. Since G has no circuits of odd length, the end vertices of every chord being replaced are differently colored in T. Thus G is colored with two colors, with no adjacent vertices having the same color. That is G is 2-chromatic.

Conversely if G has a circuit of odd length, we would need atleast three colors just for that circuit.

Thus the theorem.

Chromatic polynomial

A given graph G of n vertices can be properly colored in many different ways using a sufficiently large number of colors. This property of a graph is expressed elegantly by means of a polynomial. This polynomial is called the chromatic polynomial of G and is defined as follows.

The value of the chromatic polynomial $P_n(\lambda)$ of a graph with n vertices gives the number of ways of properly coloring the graph, using λ or fewer colors.

Let C_i be the different ways of properly coloring G using exactly i different colors. Since i colors can be chosen out of λ colors is

$\binom{\lambda}{i}$ different ways.

there are $c_i\binom{\lambda}{i}$ different ways of properly coloring G using exactly i colors out of λ colors. Since i can be any positive integer from 1 to n, (it is not possible to use more than n colors on n vertices), the chromatic polynomial is a sum of these terms, that is

$$P_n(\lambda) = \sum_{i=1}^{n} c_i\binom{\lambda}{i} = \frac{c_1\,\lambda!}{(\lambda-1)!\,1!} + \frac{c_2\,\lambda!}{(\lambda-2)!\,2!} + \frac{c_3\,\lambda!}{(\lambda-3)!\,3!} + \cdots$$

$$= c_1\frac{\lambda}{1!} + \frac{c_2\lambda(\lambda-1)}{2!} + \frac{c_3\,\lambda(\lambda-1)(\lambda-2)}{3!} + \cdots$$

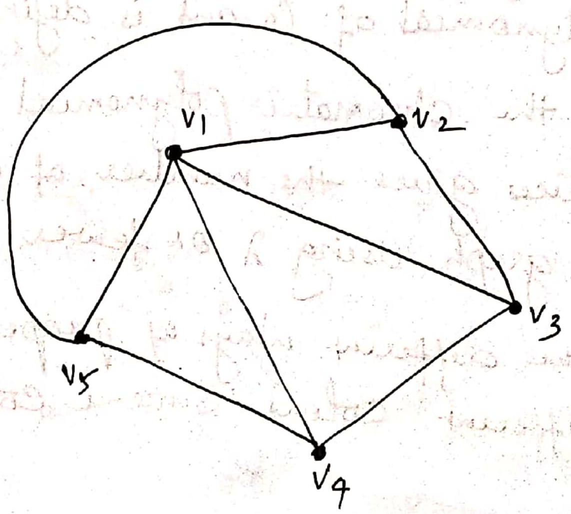
$$\cdots + \frac{c_n\lambda(\lambda-1)(\lambda-2)\cdots(\lambda-n+1)}{n!}$$

Each c_i has to be evaluated individually for the given graph.

Eg: Any graph with even one edge requires at least two colors for proper coloring and therefore $c_1 = 0$

A graph with n vertices and using different colors can be properly colored in $n!$ ways ie $c_n = n!$

Example:-



Since the graph consists of 5 vertices the chromatic polynomial to be find is $P_5(\lambda)$

$$P_5(\lambda) = C_1\binom{\lambda}{1} + C_2\binom{\lambda}{2} + C_3\binom{\lambda}{3} + C_4\binom{\lambda}{4} + C_5\binom{\lambda}{5}$$

Since the graph has a triangle, it will require at least three different colors for proper coloring. Therefore:

$$C_1 = C_2 = 0 \text{ and } C_5 = 5!$$

To evaluate C_3, Suppose that we have 3 colors $x, y \& z$. These three colours can be assigned properly to vertices V_1, V_2, and V_3 in $3! = 6$ different ways. We have no more choices left for vertex V_5, it must have the same colour as V_3 and V_4. $\therefore C_3 = 6$

With 4 colours V_1, V_2 and V_3 can be properly colored in $4 \cdot 6 = 24$ different ways - the fourth colour can be assigned to V_4 or V_5 thus providing 2 choices. The fifth vertex provides no additional choices. $\therefore C_4 = 24 \cdot 2 = 48$

Substituting these coefficients in $P_5(\lambda)$

$$P_5(\lambda) = 0 + 0 + \frac{6\lambda!}{(\lambda-3)!\,3!} + \frac{48\lambda!}{(\lambda-4)!\,4!} + \frac{5!\,\lambda!}{(\lambda-5)!\,5!}$$

$$= \frac{\lambda!}{(\lambda-3)!} + \frac{2\lambda!}{(\lambda-4)!} + \frac{\lambda!}{(\lambda-5)!}$$

$$= \lambda(\lambda-1)(\lambda-2) + 2\lambda(\lambda-1)(\lambda-2)(\lambda-3) + \frac{\lambda(\lambda-1)(\lambda-2)(\lambda-3)}{(\lambda-4)}$$

$$= \lambda(\lambda-1)(\lambda-2)[1 + 2(\lambda-3) + (\lambda-3)(\lambda-4)]$$

$$= \lambda(\lambda-1)(\lambda-2)[1 + 2\lambda - 6 + \lambda^2 - 4\lambda - 3\lambda + 12]$$

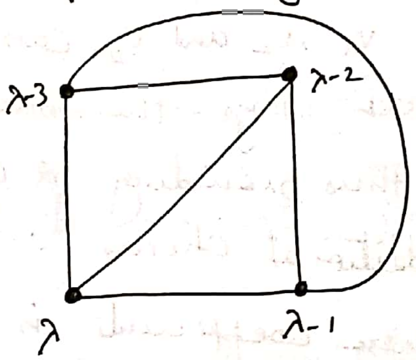
$$= \lambda(\lambda-1)(\lambda-2)[\lambda^2 - 5\lambda + 7]$$

Theorem - III

A graph of n vertices is a complete graph iff its Chromatic polynomial is $P_n(\lambda) = \lambda(\lambda-1)(\lambda-2)\cdots(\lambda-n+1)$

Proof:-

With λ colors, there are λ different ways of coloring any selected vertex of a graph. A second vertex can be colored properly is exactly $\lambda-1$ ways, the third is $\lambda-2$ ways, the fourth is $\lambda-3$ ways .. and the n^{th} is $\lambda-n+1$ ways if and only if every vertex is adjacent to every other. That is if and only if the graph is complete.



Theorem - IV

An n-vertex graph is a tree if and only if its chromatic polynomial $P_n(\lambda) = \lambda(\lambda-1)^{n-1}$

Proof

Proof is done by induction for 'n' no: of vertices

when $n=1$ (isolated vertex) $P_n(\lambda) = P_1(\lambda) = \lambda$

$n=2$ $\quad P_n(\lambda) = P_2(\lambda) = \lambda(\lambda-1)^{2-1}$

$= \lambda(\lambda-1)$

we can color 2 vertex in $P_2(\lambda)$

Assume the result is true for all trees with no: of vertices = 'K'

$$P_n(\lambda) = P_K(\lambda) = \lambda(\lambda-1)^{K-1}$$

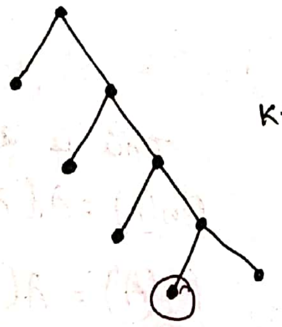
Consider a tree with K+1 vertices for a tree $n \geq 2$, it must have minimum of 2 pendant vertex

Eg:

For $n = K+1$ $\quad P_{K+1}(\lambda) = P_K(\lambda)+1 = \lambda(\lambda-1)^{K-1}(\lambda-1)^1$

$= \lambda(\lambda-1)^K$ ways.

Eg: Consider the following tree



When we remove pendant vertex from tree it become k vertex tree and can be colored in $\lambda(\lambda-1)^K$ ways. Then when we assigned back the removed vertex in its same position then it can only be colored by $(\lambda-1)$ color since its parent is already in one color so that color cannot be used to color the pendant vertex. So the total color required for that tree is

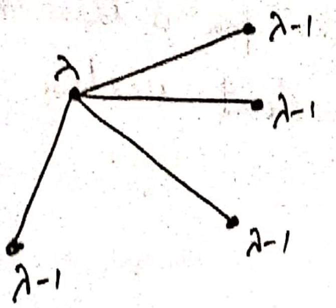
$$\lambda(\lambda-1)^{K-1}(\lambda-1) = \lambda(\lambda-1)^K \text{ ways}$$

∴ To color K+1 vertex we need $\lambda(\lambda-1)^K$ colors

Hence proved.

Example:-

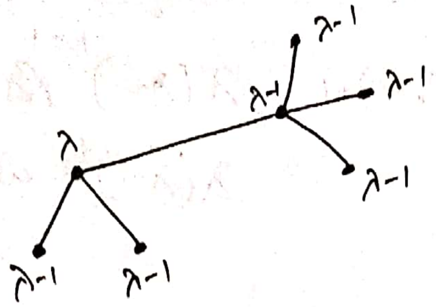
For the following given graph/Tree find the chromatic polynomial

① 

$P_n(\lambda) = \lambda(\lambda-1)^{n-1}$ $\quad n = 5$

$= \lambda(\lambda-1)^{5-1}$

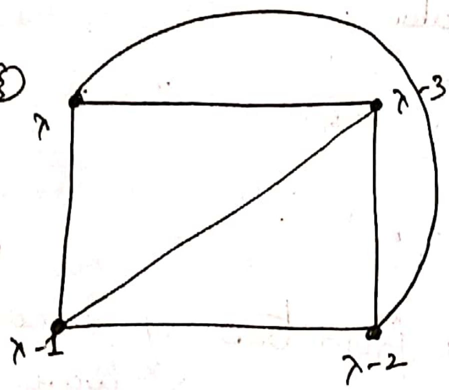
$= \lambda(\lambda-1)^{4}$

② 

$P_n(\lambda) = \lambda(\lambda-1)^{n-1}$

$= \lambda(\lambda-1)^{7-1}$

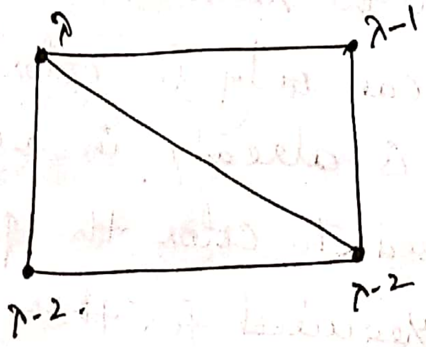
$= \lambda(\lambda-1)^{6}$

③ 

This is a complete graph.

$P_n(\lambda) = \lambda(\lambda-1)(\lambda-2)\cdots(\lambda-n+1)$

$P_4(\lambda) = \lambda(\lambda-1)(\lambda-2)(\lambda-3)$

④ 

$P_n(\lambda) = \lambda(\lambda-1)(\lambda-2)^{2}.$

$P_n(\lambda) = \sum_{i=1}^{n} C_i \binom{\lambda}{i}$

$= \sum_{i=1}^{4} C_i \binom{\lambda}{i}$

$= C_1\binom{\lambda}{1} + C_2\binom{\lambda}{2} + C_3\binom{\lambda}{3} + C_4\binom{\lambda}{4}$

$C_3 = 3! = 6$

$= 0 + 0 + C_3 \frac{\lambda!}{(\lambda-3)!\,3!} + 4!($

$= 0 + 0 + 6\frac{\lambda!}{(\lambda-3)!\,3!} + 4!\frac{\lambda!}{(\lambda-4)!\,4!}$

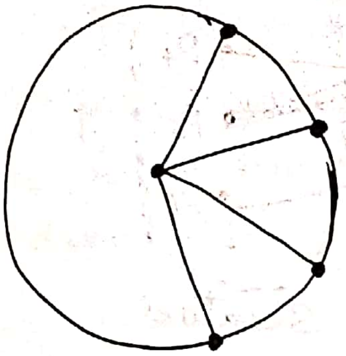
$$= 0 + 0 + \lambda(\lambda-1)(\lambda-2) + \lambda(\lambda-1)(\lambda-2)(\lambda-3)$$

$$= \lambda(\lambda-1)(\lambda-2)[1 + \lambda - 3]$$

$$= \lambda(\lambda-1)(\lambda-2)(\lambda-2)$$

$$= \lambda(\lambda-1)(\lambda-2)^2$$

(5)



$c_1 = 0$

$c_2 = 0$

$c_3 = 3!$

$c_4 = 4 \times 6 = 24 \times 2 = 48$

$c_5 = 5!$

$$P_5(\lambda) = c_1 + c_2 + c_3\binom{\lambda}{3} + c_4\binom{\lambda}{4} + c_5\binom{\lambda}{5}$$

$$= 0 + 0 + \frac{3!\,\lambda!}{(\lambda-3)!\,3!} + \frac{48\,\lambda!}{(\lambda-4)!\,4!} + \frac{5!\,\lambda!}{(\lambda-5)!\,5!}$$

$$= 0 + 0 + \lambda(\lambda-1)(\lambda-2) + 2\lambda(\lambda-1)(\lambda-2)(\lambda-3) + \lambda(\lambda-1)(\lambda-2)(\lambda-3)(\lambda-4)$$

$$= \lambda(\lambda-1)(\lambda-2)(\lambda^2 - 5\lambda + 7)$$

presence of factors $\lambda-1$ and $\lambda-2$ indicates that G is atleast 3 chromatic.

Matchings

Suppose that four applicants a_1, a_2, a_3 and a_4 are available to fill six vacant positions P_1, P_2, P_3, P_4, P_5 and P_6. Applicant a_1 is qualified to fill position P_2 or P_5. Applicant a_2 can fill P_2 or P_5. Applicant a_3

is qualified for p_1, p_2, p_3, p_4 or p_6. Applicant a_4 can fill jobs p_2 or p_5. This situation is represented by vertices.

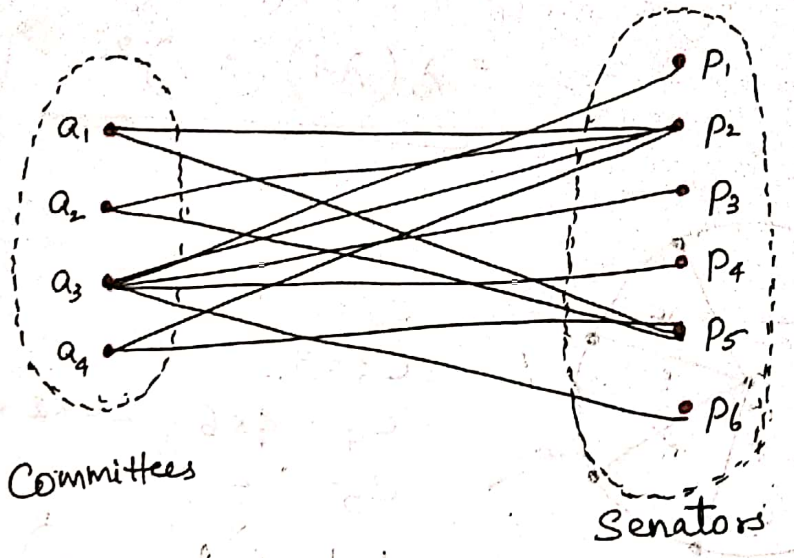
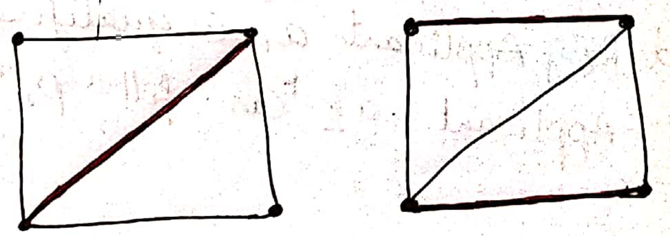


fig: Bipartite Graph.

The vacant positions and applicants are represented by vertices. The edges represent the qualifications of each applicant for filling different position. The graph clearly is bipartite, the vertices falling into two sets $V_1 = \{a_1, a_2, a_3, a_4\}$ and $V_2 = \{p_1, p_2, p_3, p_4, p_5, p_6\}$

Matching is the assignment of one set of vertices into another. Matching in a graph is a subset of edges in which no two edges are adjacent. A single edge in a graph is obviously a matching.

A <u>maximal matching</u> is a matching to which no edge in the graph can be added. For eg: in a complete graph of three vertices, any single edge is a maximal matchings



A graph may have many different maximal machings and of different size.

The maximal matchings with the largest number of edges are called <u>largest maximal matchings</u>

The number of edges in a largest maximal matching is called the <u>matching number</u> of the graph.

In a bipartite graph having a vertex partition V_1 and V_2 a complete matching of vertices in set V_1 into those in V_2 is a matching in which there is one edge incident with every vertex in V_1.

Every vertex in V_1 is matched against some vertex in V_2.

For a complete matching of set V_1 into set V_2, we must have atleast as many vertices in V_2 as there are in V_1. In other words there must be atleast as many vacant positions as the number of applicants if all the applicants are to be hired. In the above given fig Bipartite graph, although there are six positions and four applicants, a complete matching does not exist. of the three applicants a_1, a_2 and a_4 each qualifies for the same two positions P_2 and P_5 and therefore one of the three applicants cannot be matched.

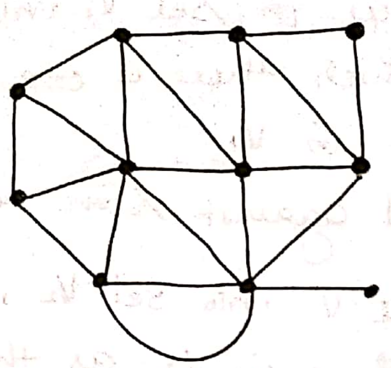
<u>Coverings</u>

In a graph G, a set g of edges is said to cover G if every vertex in G is incident on atleast one edge in g. A set of edges that covers a graph

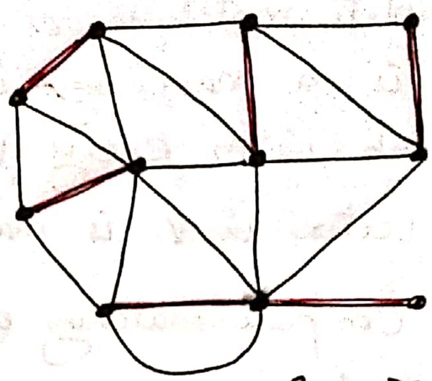
G is said to be an edge covering, a covering subgraph or covering of G. A graph G is trivially its own covering.

A spanning tree in a connected graph is another covering. A Hamiltonian circuit in a graph is also a covering.

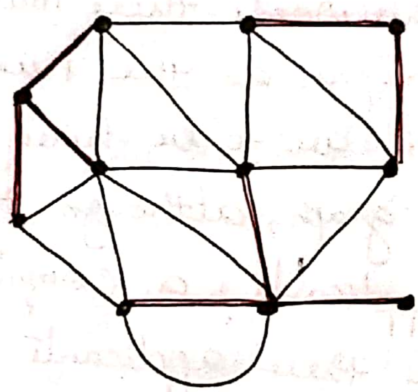
Example:-



(G)



Covering of G



Covering of G.

<u>Observations in Covering</u>

1. A covering exists for a graph if and only if the graph has no isolated vertex.

2. A covering of an n-vertex graph will have at least $\lceil n/2 \rceil$ edges. ($\lceil x \rceil$ denotes the smallest integer not less than x).

3. Every pendant edge in a graph is included in every covering of the graph.

4. Every covering contains a minimal covering.

5. If we denote the remaining edges of a graph by $(G-g)$, the set of edges g is a covering if and only if, for every vertex v, the degree of vertex in $(G-g) \leq$ (degree of vertex v in G) -1

6. No minimal covering can contain a circuit, for we can always remove an edge from a circuit without leaving any of the vertices in the circuit uncovered. Therefore a minimal covering of an n-vertex graph can contain no more than $n-1$ edges

7. A graph, in general, has many minimal coverings, and they may be of different sizes. The number of edges in a minimal covering of the smallest size is called the covering number of the graph.

Theorem - IV

A covering 'g' of a graph is minimal if and only if g contains no paths of length three or more.

Proof:-

Suppose that a covering g contains a path of length three and it is $v_1 e_1 v_2 e_2 v_3 e_3 v_4$

Edge e_2 can be removed leaving its end vertices v_2 and v_3 uncovered. Therefore g is not a minimal covering

Conversely if a covering 'g' contains no path of length three or more all its components must be star graphs. From a star graph no edge can be removed without leaving a vertex uncovered. That is g must be a minimal covering.

Following are the different star graphs

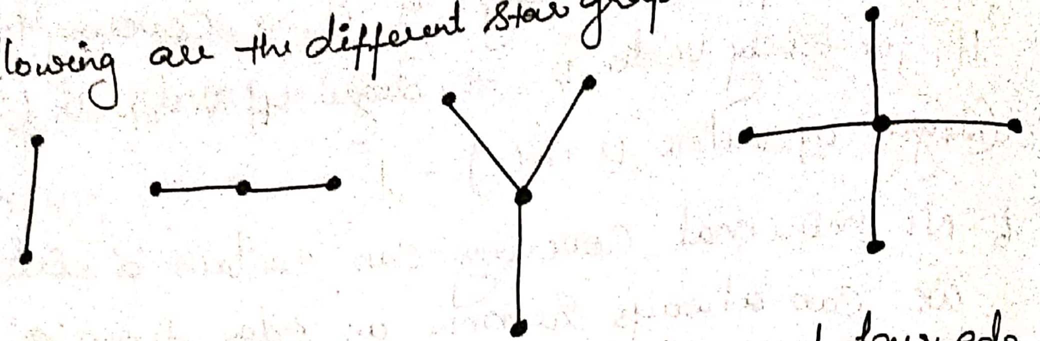
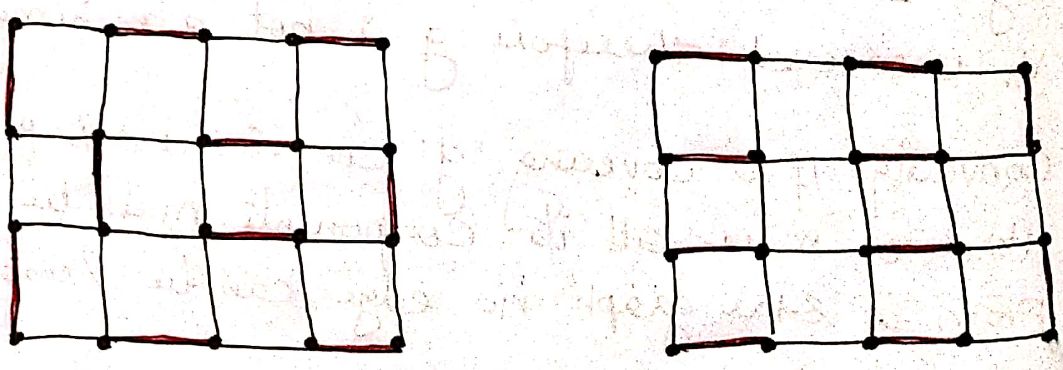


fig: star graphs of one, two, three and four edges

Dimer problem:-

A crystal is represented by a three dimensional lattice. Each vertex in the lattice represents an atom and an edge between vertices represents the bond between the two atoms.

Surface properties of crystals consisting of diatomic molecules called dimer. The problem is ~~equi~~ equivalent to finding all different coverings of a given graph such that every vertex in the covering is of degree one. Such covering in which every vertex is of degree one is called dimer covering or a 1-factor. A dimer covering is a matching because no two edges in it are adjacent. Moreover a dimer covering is a maximal matching. Dimer covering is referred to as a perfect matching. Two different dimer covering are as shown below.



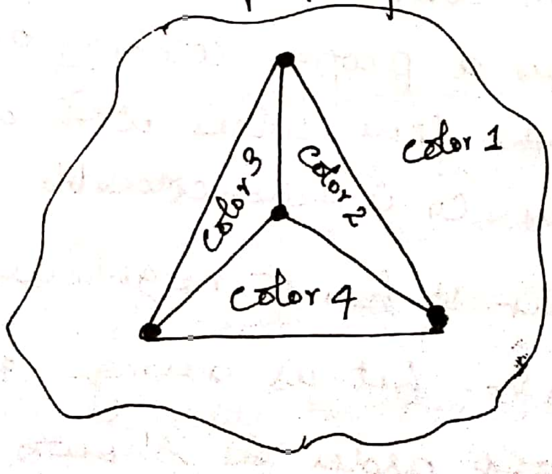
Note :-

A graph must have an even number of vertices to have a dimer covering.

Four Color problem

Consider the proper coloring of regions in a planar graph. Just as in coloring of vertices and edges, the region of a planar graph are said to be properly colored if no two contiguous or adjacent regions have the same color. (Two regions are said to be adjacent if they have a common edge between them. One or more vertices in common does not make two regions adjacent. The proper coloring of regions is also called map coloring referring to the fact that in an atlas different countries are colored such that countries with common boundaries are shown in different colors.

The regions are coloured using minimum number of colors. This leads to the most famous conjecture in graph theory. The conjecture is that every map (planar graph) can be properly colored with four colors.



Note :-

Every planar graph has a chromatic number of four or less

Five Color problem

Theorem - V

The vertices of every connected simple planar graph can be properly colored with five colors.

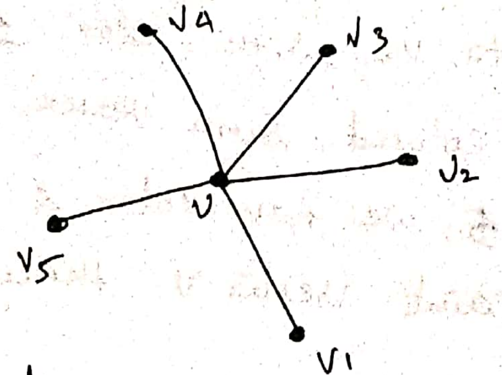
Proof:-

Let n be the number of vertices in a connected, simple planar graph. If $n \leq 5$, then the theorem is trivially true.

Assume that the theorem is true for all graph with $n \leq k$. Consider a graph G with $k+1$ vertices. Then, by Euler's theorem, G contains a vertex v of degree at most 5. If we consider the graph $H = G - v$, obtained by deleting v from G, then H has k vertices. Therefore by assumption made, H is 5-colorable.

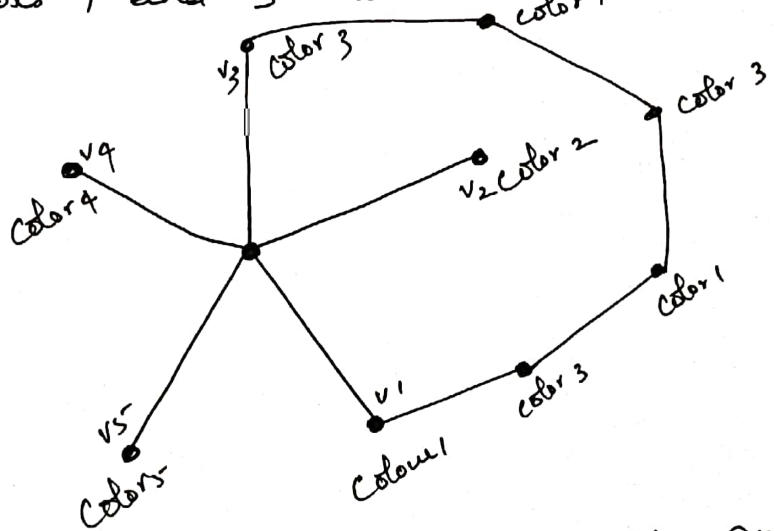
Since the degree of v is at most 5, v has atmost 5 neighbours in G. Suppose v has 4 or less number of neighbours. Then the neighbours can be colored with atmost four different colors and v can be colored with fifth color, all drawn from the colors used in H. Thus a proper coloring of G can be done by using the five colors with which H can be colored. Thus G is five-colorable.

Next, Suppose that v has 5 neighbours say v_1, v_2, v_3, v_4, v_5. Let us arrange them around v in anticlockwise order as shown in the following figure.



Since the vertices V_1, V_2, V_3, V_4 & V_5 are all mutually adjacent they constitute K_5 which is non-planar. This is not possible, because being a planar graph, G. cannot contain a non-planar graph as sub graph.

Suppose that there is a path exist in H between vertices V_3 and V_1 which can be colored alternately with colors 1 and 3 as shown in the following figure



then a similar path between V2 and V5 can be alternately colored with color 2 and 5 which cannot exists. Otherwise these two path will Intersect and cause H to be nonplanar.

If there is no paths between V2 and V_5 colored alternately with colors 2 and 5 starting from vertex V_2 we can interchange colors 2 and 5 of all vertices connected to V_2 with color ~~4~~ 5 and kept H as properly coloured. Since vertex V_5 is still ~~ab~~ with color 5, we have color 2 left over with which to paint vertex v. Hence the theorem.

CONTENT BEYOND THE SYLLABUS

Application of graphs in solving various Engineering problems

1. Introductions:

1.1. Graph theory is branch of mathematics that deals with the study of graph, that are considered to be the mathematical structure helpful to have mathematical model with pair wise relation between objectives.

Graph is made up of two things. Set of vertices and set of edges Graphs give us many techniques and flexibility while defining and solving real world problems. Graphs have many features such as,

- Establishes relation between objects.
- Helps in modeling
- Helps in decision making.

As an example in networking Engineering, Network is system of points with distances between them. A network can represent a road, pipeline, cables, etc. The problem with network involves finding the shortest path between one point to another point in the network**. [1]**

2. Applications of graph in the various engineering field

2.1. Network Engineering: In [1] and [5] author have explained an application of graph in networking. In addition to that, Graph theoretic concepts have many applications in network engineering, such as connectivity, data gathering, Energy efficiency, traffic analysis, Finding shortest path and many more, where the term Graph and Network are equivalent. In graph theory nodes and edges are used, in networking links and lines are used. The term graph is used in mathematics and Network is used in Engineering. Particularly in computer engineering.

Graph based representation for the network system makes the problem much easier and will provide much accurate results.

2.1.1. Illustration: Graph representation of circuit network

Circuit network showing the Resistors in series and parallel

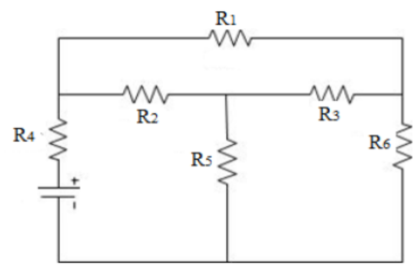


Figure 1

The logical truth table for the circuit can be shown as

	R_1	R_2	R_3	R_4	R_5	R_6
R_1	0	1	1	1	0	1
R_2	1	0	1	1	1	0
R_3	1	1	0	0	1	1
R_4	1	1	0	0	1	1
R_5	0	1	1	1	0	1
R_6	1	0	1	1	1	0

Graph model that is used to represent circuit network

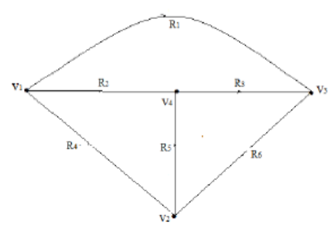


Figure 2

Again graph can be represented in one of the matrix form that is adjacency matrix taking the vertex set as $V = \{R_1, R_2, R_3, R_4, R_5, R_6\}$

$$A = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

From the adjacency matrix the energy of a graph, minimum dominating energy of a are defined. And the energy of a graph is defined as absolute sum of the Eigen values of adjacency matrix of a graph. The characteristic equation of adjacency matrix A is

$$(A - \lambda I) = \begin{bmatrix} \lambda & 1 & 1 & 1 & 0 & 1 \\ 1 & \lambda & 1 & 1 & 1 & 0 \\ 1 & 1 & \lambda & 0 & 1 & 1 \\ 1 & 1 & 0 & \lambda & 1 & 1 \\ 0 & 1 & 1 & 1 & \lambda & 1 \\ 1 & 0 & 1 & 1 & 1 & \lambda \end{bmatrix} = 0$$

That is, the characteristic equation is $\lambda^6 - 12\lambda^4 - 16\lambda^3 = 0$ and eigen values are

$\lambda = 0, 0,0, -2, -2, and\ 4$. Therefore energy of a graph is

$E(G) = |0|\{three\ times\} + |-2|\{two\ times\} + |4|(1) = \mathbf{8}$

2.2. Electrical Engineering: Electrical Circuits are closed loop formed by Source, Wires, Load and Switches. When switch is turned on electrical circuit is complete. Then current flows from negative terminal of source of power. Here we apply the concept of Graph Theory to solve Electrical Circuit Problems. In [4] author have discussed regarding an application of graph in electric circuits. Using the definition of a link and cycle matrix for the graph, we consider one more application of graph in electric field.

2.2.1. Link: A branch of a graph which does not belongs to particular tree under consideration.

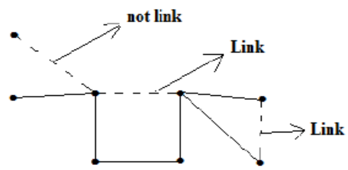


Figure 3

2.2.2. Cycle matrix (Loop matrix) : It is matrix with elements as $\boldsymbol{b_{ij}}$

$$b_{ij} = \begin{cases} 1 & Branch\ b_j'\ direction\ in\ the\ loop\ is\ same\ as\ direction\ of\ loop \\ -1 & Branch\ b_j'\ direction\ in\ the\ loop\ is\ opposite\ as\ direction\ of\ loop \\ 0 & When\ \ b_j\ is\ not\ in\ the\ loop \end{cases}$$

Loop matrix is a $i \times b$ matrix where i is the number of loops and b is he number of branches. A set of branches contained in a loop such that each loop contains one link and the remaining are the tree branches.

2.2.3. Illustration:

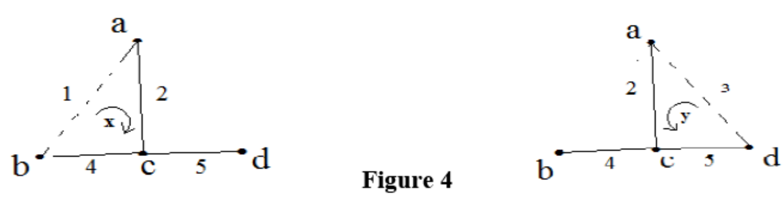


Figure 4

Selecting (2,4,5) as a tree and the co-tree is (1,3). For the above fig oriented cycle matrix (loop matrix) is $M = B = \begin{bmatrix} 1 & 1 & 0 & -1 & 0 \\ 0 & 1 & 1 & 0 & -1 \end{bmatrix}$

2.2.4. Impendence: Electric impendence is the measure of opposition that a circuit presents to a current when voltage is applied.

Consider an electric circuit shown bellow, the graph for the circuit is shown.

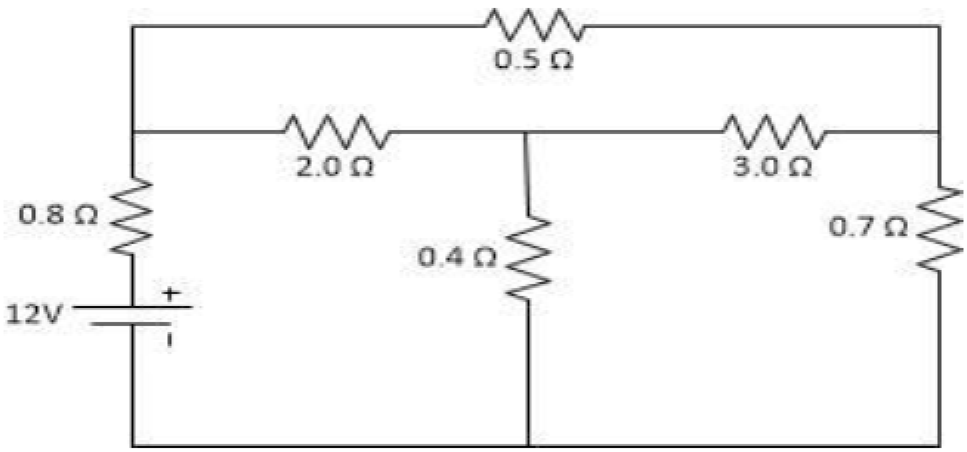
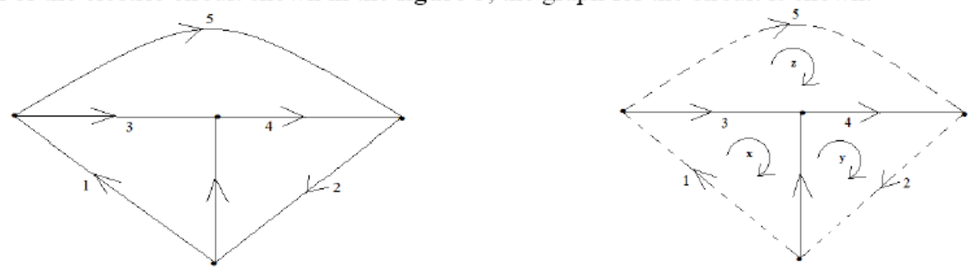


Figure 5

Now we discuss the graphical method of finding branch currents

For the electric circuit shown in the **figure 5**, the graph for the circuit is shown.



Graph representing circuit **Figure 6** Tree and Co-tre

The oriented cycle matrix is, $B = \begin{array}{c} x \\ y \\ z \end{array}\begin{bmatrix} 1 & 0 & 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & -1 & -1 & 1 & 0 \end{bmatrix}$

If branch impendence matrix is denoted by Z_b, then $(i,j)^{th}$ the elements of the matrix Z_b are defined as

$$Z_b = \begin{cases} R_i & for\ i = j \\ 0 & for\ i \neq j \end{cases}$$

$$\Rightarrow \text{ Branch impendence matrix, } Z_b = \begin{bmatrix} 0.8 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.7 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 2.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 3.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.5 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.4 \end{bmatrix}$$

Then branch source voltage vector is obtained as the product of oriented cycle matrix and branch impendence matrix, That is,

$$BZ_b = \begin{bmatrix} 0.8 & 0 & 2 & 0 & 0 & -4 \\ 0 & 0.7 & 0 & 3 & 0 & 4 \\ 0 & 0 & -2 & -2 & 0.5 & 0 \end{bmatrix} = E_{gb} \text{ (Say)}$$

And the mess (loop) impendence matrix is defined as $E_{gb}B'$

Therefore we get, $E_{gb}B' = BZ_bB' = \begin{bmatrix} 6.8 & -4 & -2 \\ -4 & 7.7 & -3 \\ -2 & -3 & 5.5 \end{bmatrix} = Z_l(say)$

Mess source voltage vector is given by BE_b which is obtained as the product of the matrix B

And E_b.That is $BE_b = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & -1 & -1 & 1 & 0 \end{bmatrix}\begin{bmatrix} 12 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 12 \\ 0 \\ 0 \end{bmatrix}$

If I_l represents the currents in the loops, then $I_l = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

Loop equations are $Z_l I_l = BE_b$

$$\Rightarrow \quad \begin{bmatrix} 6.8 & -4 & -2 \\ -4 & 7.7 & -3 \\ -2 & -3 & 5.5 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 12 \\ 0 \\ 0 \end{bmatrix}$$

$$\Rightarrow x = 6.672A, \quad y = 5.602A, \quad z = 5.482A.$$

Hence proposed graph theoretical method can be applied to solve electrical circuit problems to branch currents in the circuit.

2.3. Computer Science Engineering: Graph theory can be used in research areas of computer science**. In [2] [3]** uses of graph in computer engineering are explained. Along with those few more application are explained. Some of them are data structure, Image processing, Web designing, data mining, clustering, etc. Some of them are discussed here.

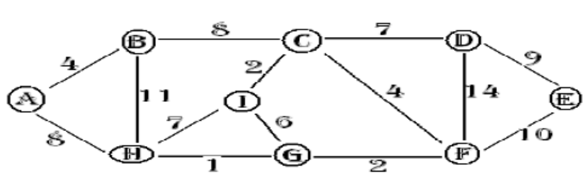
2.3.1. Data structure: It is a systematic method of organizing and storing the data. It is designed to suit a specific purpose so that it can be accessed and worked with appropriate ways. The selection of the model for the data depends on the information of the real world and the structure should be simple enough that can effectively process data whenever it is required.

2.3.2. Image processing: It is process of analysis and manipulation of digitized image, especially in order to improve the quality of an image. It is a method by which the information from the image is extracted. Using graph theory concept image processing method can be improved. The graph theory provides the calculation of alignment of the picture. It finds the mathematical constraints using **minimal spanning tree**.

2.3.3. Web designing: Web designing is also method of creating the web site that encompasses several different aspects including web page, layout, content production and graphic design. While the term web design and web development are often used interchangeably. Web design is a subset of web development. Here web pages are represented by the vertices and the links between the web pages are represented by the edges of the graph. In the web community the vertices are representing the classes of objects and each vertex represents one nature of the object and each vertex representing one type of object is connected to every vertex representing the other kind of the object. In graph theory same concept is explained by **complete bipartite graphs.**

The concepts of weighted graph in graph theory, where weights have been assigned to the edges of the graph are used to represent the structure, wherein the paired connections have numerical values. If the edges represent the roads between the places then problem is to fond shortest path connecting all places which is solved by minimum spanning tree. There are many methods of finding minimum spanning tree, the new and simple approach that we propose is an E**dge Elimination Method**" is illustrated here.

Considering the weighted graph shown in the following figure



The algorithm for the edge elimination method is

Step 1: The edge with highest weight from each smallest cycle is eliminated with care that graph is not disconnected.

Step 2: step 1 repeated for the cycles formed after the step 1.

Step 3: Continue the process of elimination of edges of highest weight from each smallest cycle obtained in the step 2 using step 1, until no cycle remains in the graph. So that there are exactly (n-1) edges, n is the number of vertices. The resulting graph is the minimum spanning tree as shown.

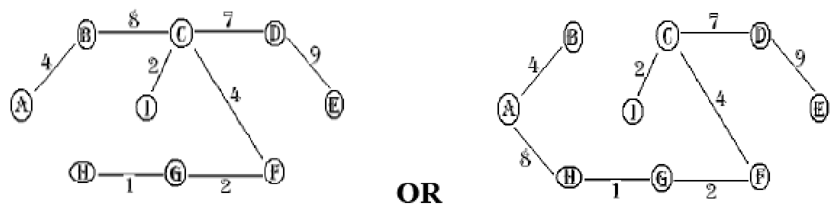


Figure 8 (Minimum spanning tree of 37)

APPLICATIONS OF GRAPH THEORY IN HUMAN LIFE

1. Introduction:

The origin of graph theory started with the problem of Koinsber bridge, in 1735. This problem lead to the concept of Eulerian Graph. Euler studied the problem of Koinsberg bridge and constructed a structure to solve the problem called Eulerian graph. In 1840, A.F Mobius gave the idea of complete graph and bipartite graph and Kuratowski proved that they are planar by means of recreational problems. The concept of tree, (a connected graph without cycles was implemented by Gustav Kirchhoff in 1845, and he employed graph theoretical ideas in the calculation of currents in electrical networks or circuits. In 1852, Thomas Gutherie found the famous four color problem. Then in 1856, Thomas. P. Kirkman and William R.Hamilton studied cycles on polyhydra and invented the concept called Hamiltonian graph by studying trips that visited certain sites exactly once. In 1913, H.Dudeney mentioned a puzzle problem. Eventhough the four color problem was invented it was solved only after a century by Kenneth Appel and Wolfgang Haken. This time is considered as the birth of Graph Theory. Caley studied particular analytical forms from differential calculus to study the trees. This had many implications in theoretical chemistry. This lead to the invention of enumerative graph theory. Any how the term "Graph" was introduced by Sylvester in 1878 where he drew an analogy between "Quantic invariants" and covariants of algebra and molecular diagrams. In 1941, Ramsey worked on colorations which lead to the identification of another branch of graph theory called extremel graph theory. In 1969, the four color problem was solved using computers by Heinrich. The study of asymptotic graph connectivity gave rise to random graph theory.

Applications of GSM and Time Table Scheduling :

Graph theoretical concepts are widely used to study and model various applications, in different areas. They include, study of molecules, construction of bonds in chemistry and the study of atoms. Similarly, graph theory is used in sociology for example to measure actors prestige or to explore diffusion mechanisms. Graph theory is used in biology and conservation efforts where a vertex represents regions where certain species exist and the edges represent migration path or movement between the regions. This information is important when looking at breeding patterns or tracking the spread of disease, parasites and to study the impact of migration that affect other species. Graph theoretical concepts are widely used in Operations Research. For example, the traveling salesman problem, the shortest spanning tree in a weighted graph, obtaining an optimal match of jobs and men and locating the shortest path between two vertices in a graph. It is also used in modeling transport networks, activity networks and theory of games. The network activity is used to solve large number of combinatorial problems. The most popular and successful applications of networks in OR is the planning and scheduling of large complicated projects. The best well known problems are PERT(Project Evaluation Review Technique) and CPM (Critical Path Method). Next, Game theory is applied to the problems in engineering, economics and war science to find optimal way to perform certain tasks in competitive environments. To represent the method of finite game a digraph is used. Here, the vertices represent the positions and the edges represent the moves.

Map coloring and GSM mobile phone networks:

Groups Special Mobile (GSM) is a mobile phone network where the geographical area of this network is divided into hexagonal regions or cells. Each cell has a communication tower which connects with mobile phones within the cell. All mobile phones connect to the GSM network by searching for cells in the neighbours. Since GSM operate only in four different frequency ranges, it is clear by the concept of graph theory that only four colors can be used to color the cellular regions. These four different colors are used for proper coloring of the regions. Therefore, the vertex coloring algorithm may be used to assign at most four different frequencies for any GSM mobile phone network. The authors have given the concept as follows:

Given a map drawn on the plane or on the surface of a sphere, the four color theorem assets that it is always possible to color the regions of a map properly using at most four distinct colors such that no two adjacent regions are assigned the same color. Now, a dual graph is constructed by putting a vertex inside each region of the map and connect two distinct vertices by an edge iff their respective regions share a whole segment of their boundaries in common. Then proper coloring of the dual graph gives proper coloring of the original map. Since, coloring the regions of a planar graph G is equivalent to coloring the vertices of its dual graph and vice versa. By coloring the map regions using four color theorem, the four frequencies can be assigned to the regions accordingly.

Time table scheduling: Allocation of classes and subjects to the Teachers is one of the major issues if the constraints are complex. Graph theory plays an important role in this problem. For „t" Teachers with „n" subjects the available number of „p" periods timetable has to be prepared. This is done as follows. A bipartite graph (or bigraph is a graph whose vertices can be divided into two disjoint sets *U* and *V* such that every edge connects a vertex in *U* to one in *V*; that is, *U* and *V* are independent sets) G where the vertices are the number of Faculty say t_1, t_2, t_3, t_4, ……. t_k and n number of subjects say n_1, n_2, n_3, n_4, ……. n_m such that the vertices are connected by „pi" edges. It is presumed that at any one period each Teacher can teach at most one subject and that each subject can be taught by maximum one Teacher. Consider the first period. The timetable for this single period corresponds to a matching in the graph and conversely, each matching corresponds to a possible assignment of Teacher to subjects taught during that period. So, the solution for the timetabling problem will be obtained by partitioning the edges of graph G into minimum number of matching. Also the edges have to be colored with minimum number of colors. This problem can also be solved by vertex coloring algorithm. " The line graph L(G) of G has equal number of vertices and edges of G and two vertices in L(G) are connected by an edge iff the corresponding edges of G have a vertex in common. The line graph L(G) is a simple graph and a proper vertex coloring of L(G) gives a proper edge coloring of G by the same number of colors. So, the problem can be solved by finding minimum proper vertex coloring of L(G)." For example, Consider there are 4 Teachers namely t_1, t_2, t_3, t_4,. and 5 subjects say n_1, n_2, n_3, n_4, n_5 to be taught. The teaching requirement matrix $p = [p_{ij}]$ is given as.

P	n1	n2	n3	n4	n5
t1	2	0	1	1	0
t2	0	1	0	1	0
t3	0	1	1	1	0
t4	0	0	0	1	1

Figure – a: The teaching requirement matrix for four Teachers and five subjects The bipartite graph is constructed as follows.

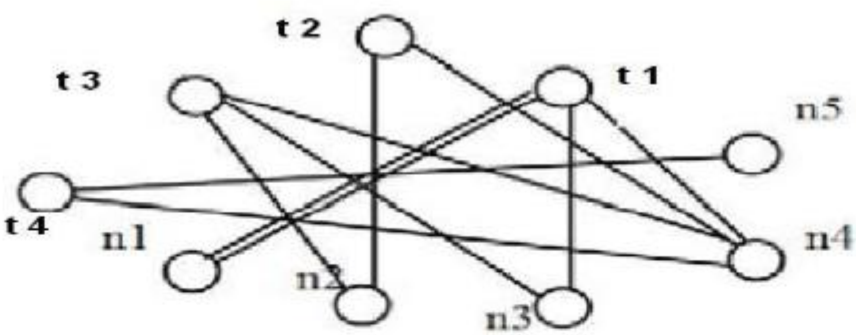


Figure – b: Bipartite graph with 4 Teachers and 5 subjects

Finally, the authors found that proper coloring of the above mentioned graph can be done by 4 colors using the vertex coloring algorithm which leads to the edge coloring of the bipartite multigraph G. Four colors are interpreted to four periods

….	1	2	3	4
t1	n1	n2	n3	n4

Figure – c: The schedule for the four subjects