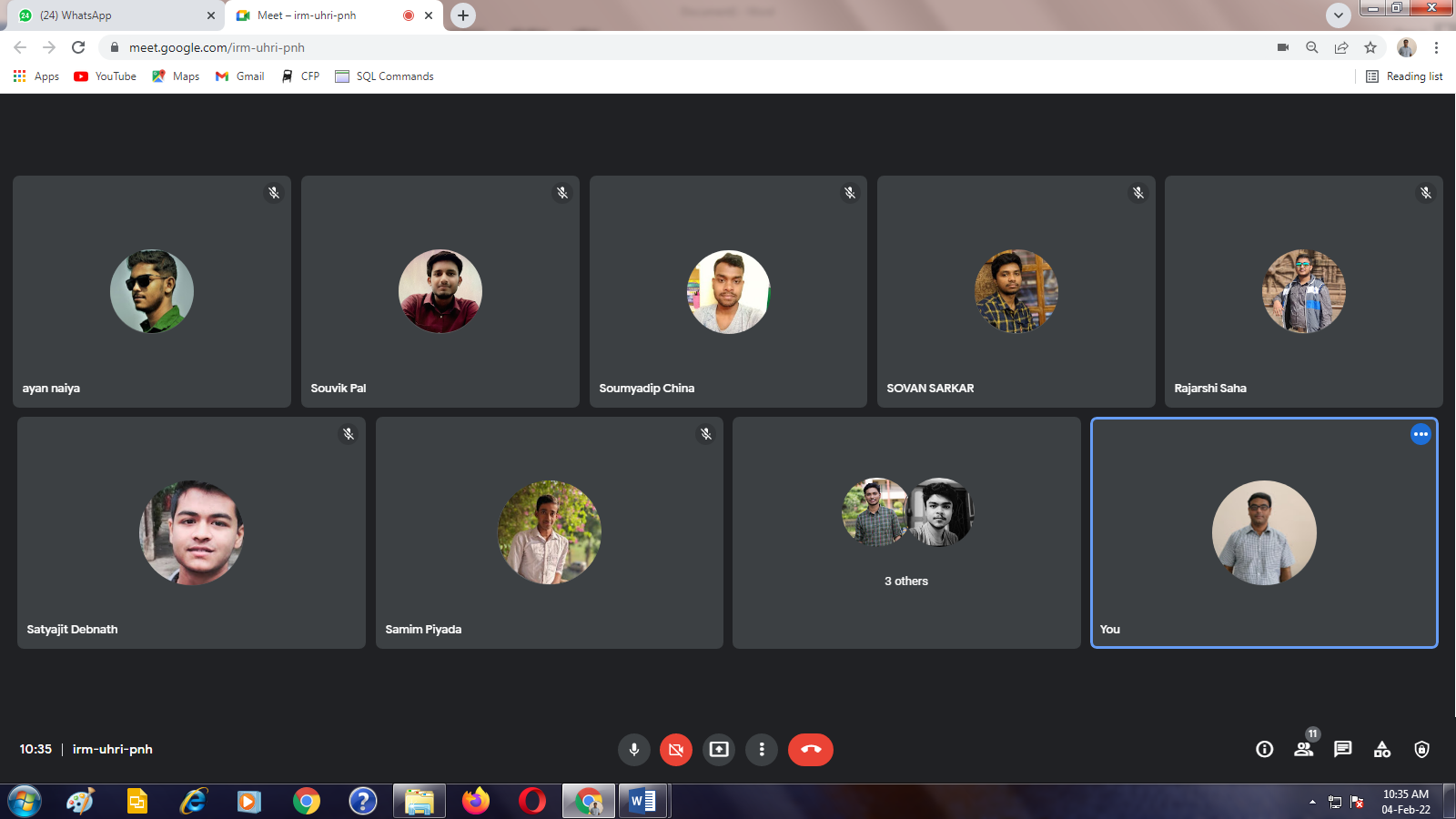
**CLASS 4/2/2022**

**UG SEMSTER-3**

**GRAPH THEORY**



**Spanning tree**

**A spanning tree T of a connected graph G , is a sub graph of G such that T contains all the vertices of G. T must be a tree i.e. it must be a connected sub-graph and also acyclic.**

**THEOREM 3-11**

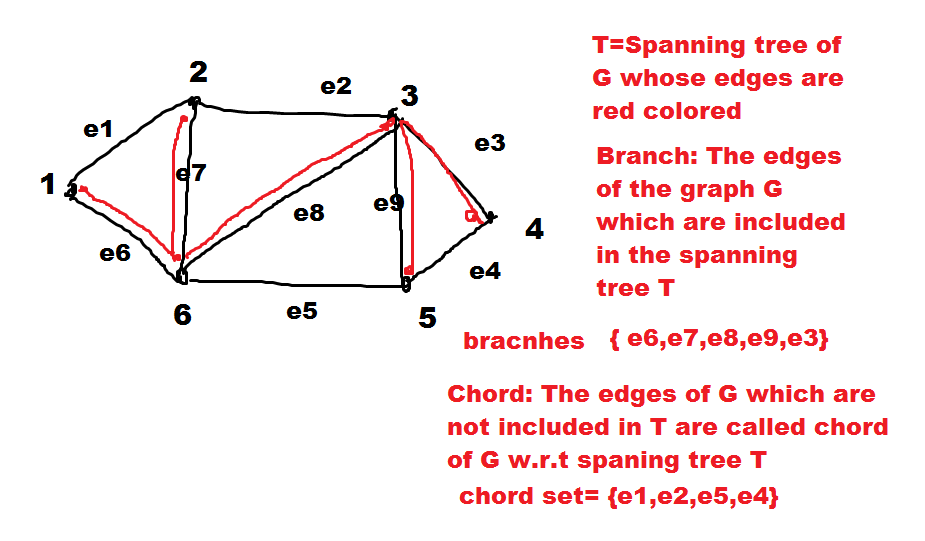
**Every connected graph has at least one spanning tree.**

**Let a graph G is connected.**

**If G has no cycle. So G is a spanning tree of itself.**

**If there is a cycle in G we can remove some edges from G remove the cycle. We don’t remove vertices. So if G contains multiple cycles this process can be repeated to remove all the cycles leaving G connected and circuitless. So this becomes a spanning tree of G as it contains all the vertices of G.**

**So Every connected graph has at least one spanning tree.**

****

**If a connected graph G has n vertices then any spanning tree T of G will have (n-1) branches and (|E|-n+1) chords.**

**The graph G-T=T’ is a collection of chords that we get after removing branches of T from G. This graph T’ is called chord set or co-tree.**

**THEOREM 3-12**

**With respect to any of its spanning trees, a connected graph of n vertices and e edges has n − 1 tree branches and e − n + 1 chords.**

**Fundamental numbers of a graph**

**n=Order= No of vertices**

**E=Size =No of edges**

**k=Number of components**

**Rank and Nullity:**

Fundamental numbers of a graph.

Order of a graph=Number vertices of a graph=n

Size of a graph=Number of edges of a graph=e

Number of components =k

A graph has n vertices and k components.

Each components are having n1,n2,….,nk number of vertices.

If each component is a legitimate graph that it must contain at least one vertex.

n1>=1

or (n1-1)>=0

similarly (n2-1)>=0,……..,(nk-1)>=0

add up for all components

(n1-1)+(n2-1)+……+(nk-1)>=0

or(n1+n2+…..+nk)-k>=0

or (n-k)>=0

(n-k)=Rank of a graph=r

Each components is connected. Each component must be at least a tree.

1st components has edges e1.

so : e1>=(n1-1)

similarly for 2nd components

e2>=(n2-1)

and so on

Adding up

e1+e2+e3+………+ek>=(n1-1)+(n2-1)+….+(nk-1)

or e>=(n1+n2+…..+nk)-k

or e>=n-k

(e-n+k)>=0

(e-n+k)=Nullity of the graph=μ

**For a connected graph G.**

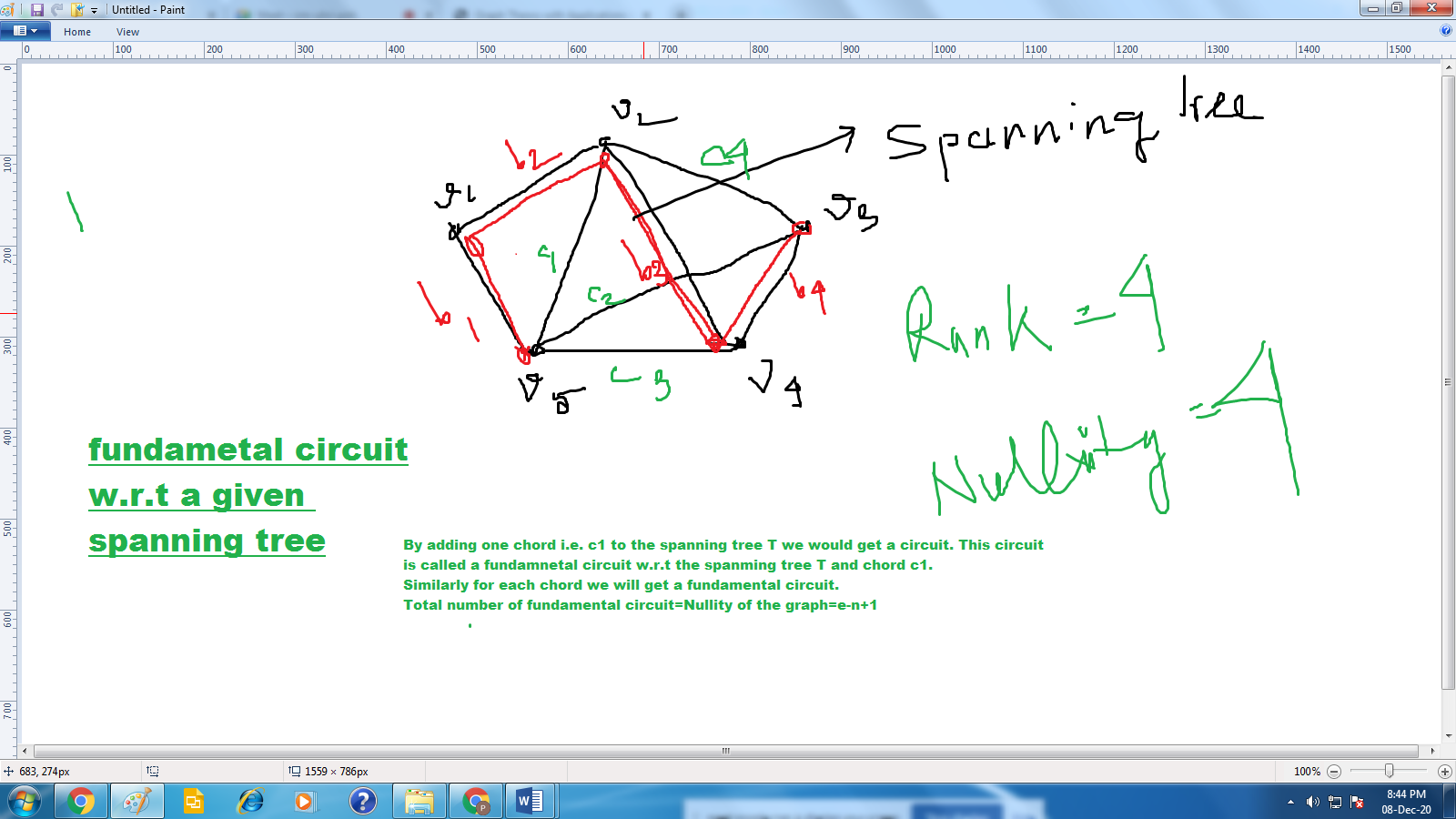
k=1

rank=n-1=Number of branches in any spanning tree of G

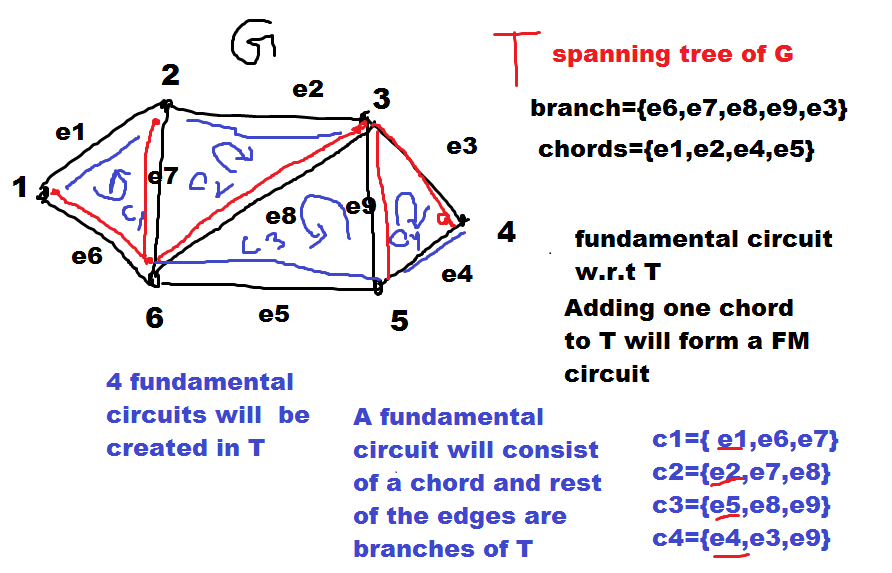
nullity=e-n+1=Number of chords w.r.t any spanning tree of G

|E|=rank+nullity

**FUNDAMENTAL CIRCUITS of a connected graph**



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

****

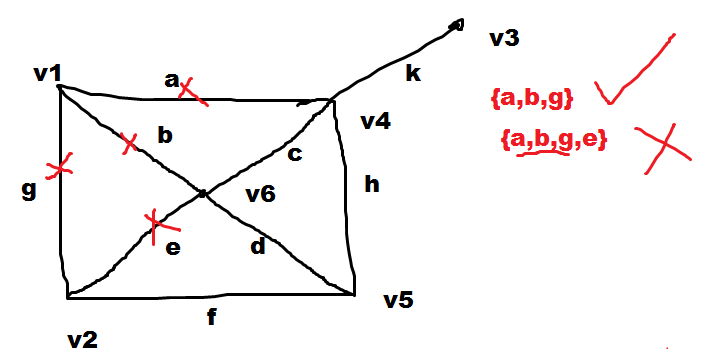
**A connected graph G is a tree if and only if adding an edge between any two vertices in G creates exactly one circuit.**

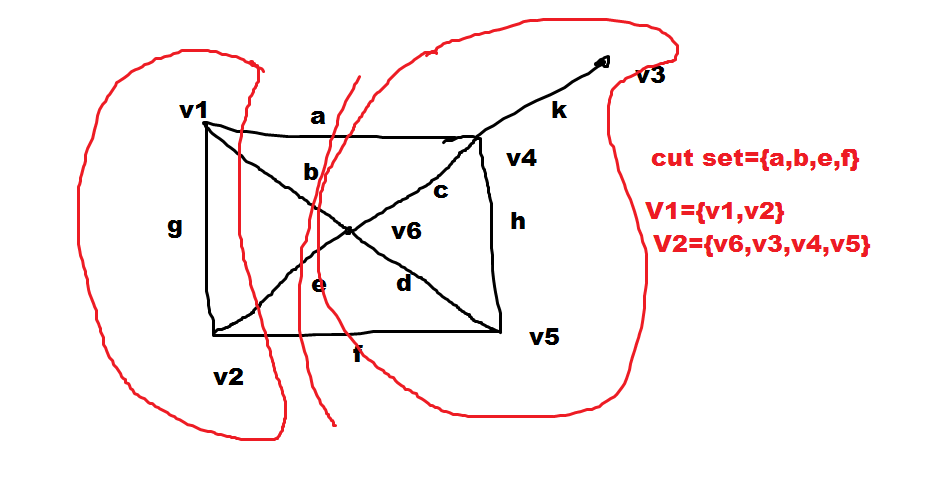
Let us consider a connected graph G. From the statement, if we add an edge e between any two vertices v1 and v2 of G then it will create exactly one circuit. So this must be a fundamental circuit. Hence there must be a spanning tree of G associated with it. If we remove the chord e from G then we must get the spanning tree of G which is the graph itself. Hence G is a tree.

Conversely, if we have a tree G then it is its own spanning tree. By adding an external edge between any two vertices of G means adding an external chord to G. Hence definitely it will create a fundamental circuit. Hence exactly one circuit can be created in this way.

**Cut set|:**

In a connected graph G, a cut-set is a set of edges† whose removal from G leaves G disconnected, provided removal of no proper subset of these edges disconnects G.





**THEOREM 4-1**

**Every cut-set in a connected graph G must contain at least one branch of every spanning tree of G.**

* Let us consider a cut set S in a connected graph G.

Let us also consider that G has a spanning tree T, which does not contain any edge of S.

If after deleting every edges of S from G, it will still be connected as the spanning tree T doesnt have any edges common with S i.e T still exists in (G-S). But is not not possible from the definition of cut set. Hence such a cut set S is not possible.

(Proved)

**THEOREM 4-2**

**In a connected graph G, any minimal set of edges containing at least one branch of every spanning tree of G is a cut-set.**

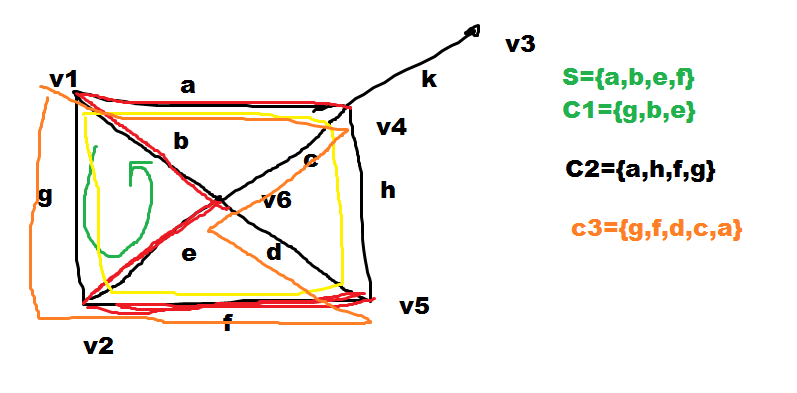
Let us consider a connected graph G. Let us also consider a minimal set Q which contains one branch of every spanning tree of G.

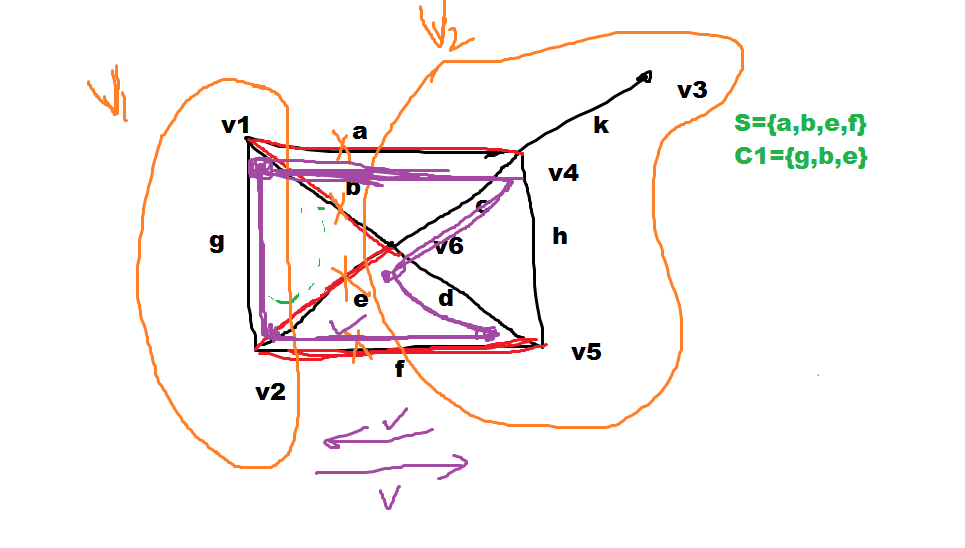
G-Q is the graph which we get after deleting every edges of Q from G.

So now (G-Q) does not contain any spanning tree. So G-Q is disconnected. By definition Q is a cut set. Now if we insert an edge e from Q to G-Q then G-Q+e should contain at one spanning tree. So G-Q+e is connected. Q is minimal.

**THEOREM 4-3**

**Every circuit has an even number of edges in common with any cut-set.**

****

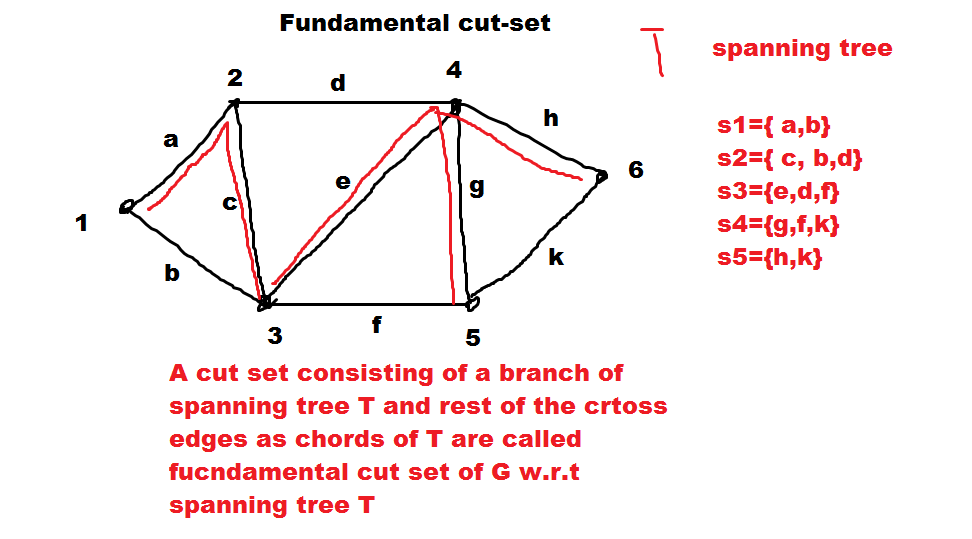
****

* Let us consider a connected graph G. Let us consider a cut set S in G. Let us consider a circuit C in G.

**Now we delete edges of S from G, so G become disconnected and the cut set S partitions the vertex set V of G into two disjoint subsets V1 and V2 such that there exist no path between any vertex of V1 to any vertex of V2.**

Let us consider the circuit C is embedded inside one of the components of G i.e the vertices of C either belongs to V1 or in V2. In that case there is no common edge between C and S. hence number of common edge is 0 i.e. even.

Let us consider the edges of C spans both the disconnected components of G. Now if we trace C from any one of the components of G when we try to cross from one component to the other we must take one edge of the cut set to cross over. Now the circuit is closed in nature so we must come back to the first component. Again we have to use another edge from the cut set. This back and forth movement can be done many times. So every time we do this back and forth movement we have take 2 edges from the cut set S. hence common edges between the circuit C and S is a multiple of 2 i.e. even.(proved)



**Assignment Graph theory**

1. Implement adjacency list from adjacency matrix.
2. Perform DFS and BFS on graph.
3. Find out all connected components of a graph.
4. Detect if there is cycle in a graph (union and find).
5. Implement Dijkstra algorithm to find shortest path from a source vertex to destination vertex.
6. Implement Floyds algorithm to find out all pair of shortest path in a graph.
7. Implement Kruskal algorithm to find minimal spanning tree of a graph.
8. Implement Prims algorithm to find minimal spanning tree of a graph.
9. Implement topological sorting of a graph.
10. Implement the algorithm to find cut set in a graph.