* Assignment No-4*.

○ Title-: program for represent the graph using adjancy matrix and adjancy list.

○ Objective-:

To study and implement the graph Representation using adjancy matrix and adjancy list.

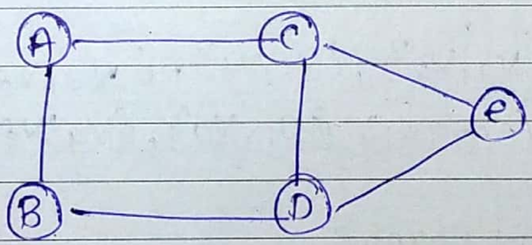
○ problem statement-:

Write function to get the number of vertices in on undirected graph add its edges. You vertices in an undirected graph and its edges. You may assume that no edges is input twice.

i) use adjancy list representation of the graph and find runtime of the function.

ii) Use adjancy matrix representation of the graph and find runtime of the function.

○ Outcome:



* Theory-:

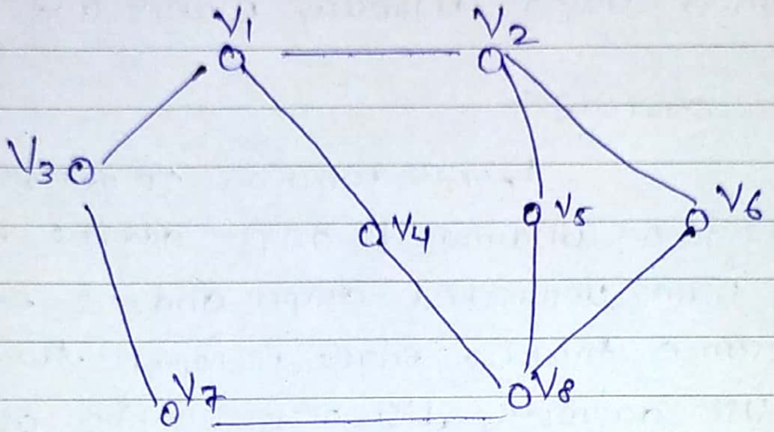
Graph -:

Definition -: A graph is set of vertices and Edges. The set V is finite non-empty set of vertices. The set E is a set of pair of vertices representation edges.

$$\boxed{G = (V, E)}$$

$V(G)$ = Vertices of a graph G,
$E(G)$ = Edges of a graph G.

An Example of graph is show below-:



The Set Representation of each of these graphs is given by

$V = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8\}$

$E = \{\{v_1, v_2\}, \{v_2, v_6\}, \{v_6, v_8,\} \{v_8, v_7\}, \{v_7, v_3\}, \{v_3, v_1\} \{v_1, v_4\}, \{v_4, v_8\} \{v_2, v_5\}, \{v_5, v_8\}\}$

+ Types of Graph:

1. Undirected graph:
2. Directed graph:
3. Complete graph:
4. weighted graph:
5. connected graph:

① Undirected graph-:

A graph conteining unordered pair of vertices is called an undirected graph.

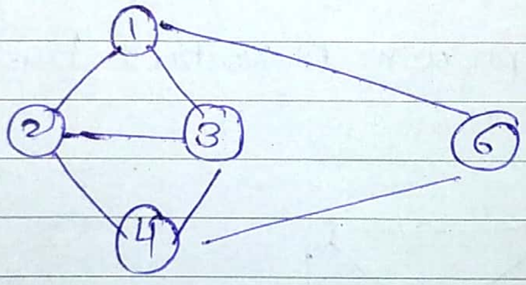
An Unorderd pair is simply a set of two elements. Order is not important here.

{a,b} = {b,a}. It does'nt matter which object is first and which object is second.

• Undirected edges can be represented Using unordered pair.

This edge is bidirectional
{u,v} = {v,u}.

e.g.:



The set of Vertices = V = {1, 2, 3, 4, 5}.

The set of edges = E = {(1,2);(1,3);(1,5),(2,3),(2,4),
(3,4),(4,5)}.

② Directed Graph-:

A graph conteining ordered pair of Vertices is called. a directed graph. If an edge is represented. Using a pair Vertices (V_1,V_2) then the edges is said tobe directed from V_1 to V_2.

Example-:

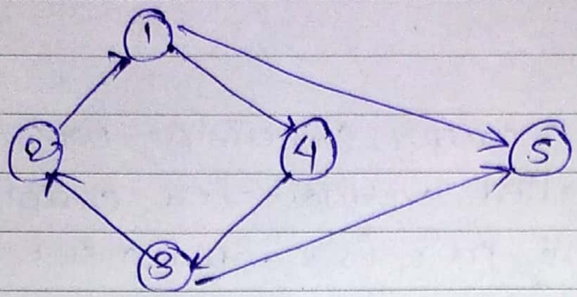


fig. directed graph.

The set of Vertices = $V = \{1,2,3,4,5\}$.

The set of edges = $E = \{(1,3), (1,5), (2,1), (2,4), (3,4), (4,5)\}$.

③ Complete graph -:

i) An Undirectional graph, in which every ~~anghol~~ Vertex has on edge to all other Vertices is called a Complete graph.

A Complete graph with N Vertices has $\frac{N(N-1)}{2}$ edges

Example-:

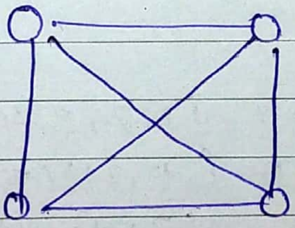


fig. Complete graph.

In a Complete graph, there is an edge between every pair of vertices

Number of edges (a pair of Vertices) in a graph with n Vertices is equal to combination of n element taking 2 at a time.

$$ {}^{n}C_2 = \frac{\lfloor n}{\lfloor n-2 \times \lfloor 2} = \frac{n \times (n-1)}{2}. $$

4) weigheted graph:-

A weighed graph is a graph inwhich edge are assigned some here. most of the physical situation are shown Using weighted graph.

An edge may represent a heighway link between two cities. The weight will denote the distance between two Conneeted cities Using highway. Wleighted of an edge is called its cost.

e.g.:

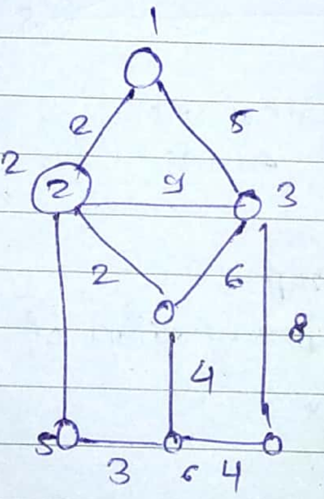


fig. weighted graph.

(5) conneeted graph:-

A graph is said to be conneated if there exists a path between every pair of Verties Vi and Vj.

e.g.

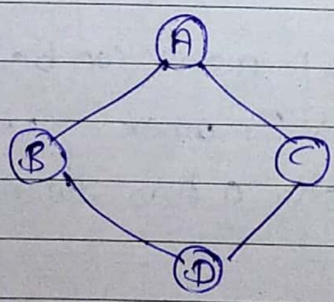
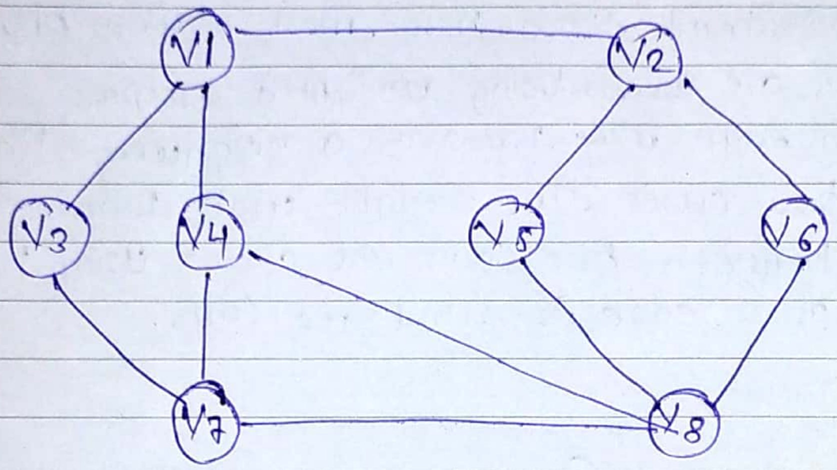


fig. Conneeted graph.

* properties of graph.:



|V| → Number of Vertices
|E| → Number of edges.

* Representation of graph-:

Methods for Representation of graph includes

1) Adjanceay matrix.
2) Adjunceey list.
3) Multilist Representation of graph.
4) Inverse Adjancy list.

1) Adjuneey Matrix-:

• Use adjucency Matrix Representation of graph and find runtime of function

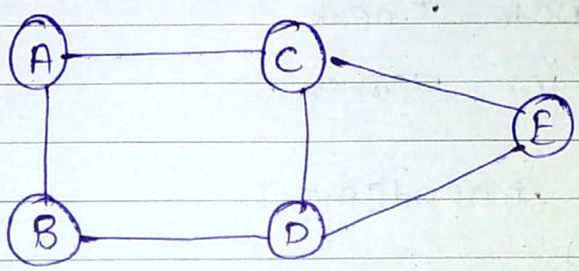
A two dimentional matrix can be used to store a graph. A graph G = (V,E) where V = {0,1,2....n-1} can be represented Using a two dimential integer array of size.

int adj[20][20] can be used to store a graph when 20 vertices.

adj[i][j] = 1 indicates presence of edges betn. two vertices i and j.

adj[i][j] = 0 , indicates absence of edges betn two vertices. i and j.

for E.g.



Adjancency Matrix :

$$\begin{array}{cccccc} & A & B & C & D & E \\ A & 0 & 1 & 1 & 0 & 0 \\ B & 1 & 0 & 0 & 1 & 0 \\ C & 0 & 0 & 0 & 1 & 1 \\ D & 0 & 1 & 1 & 0 & 1 \\ E & 0 & 0 & 1 & 1 & 0 \end{array}$$

* Adjacency list :

A graph can be represented using a linked list for each vertex, a list of adjacency vertices is maintained using a linked list. It contains a seperate linked list for each vertex Vi and the graph G={V,E}

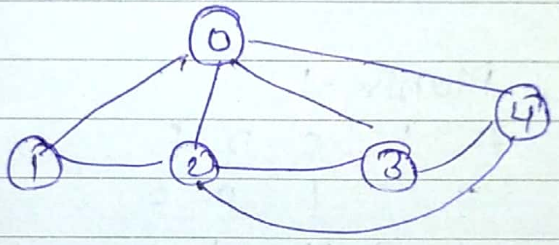
Adjacency list representation of graph is very memory efficient when the graph has a large number of vertices but very few edges.

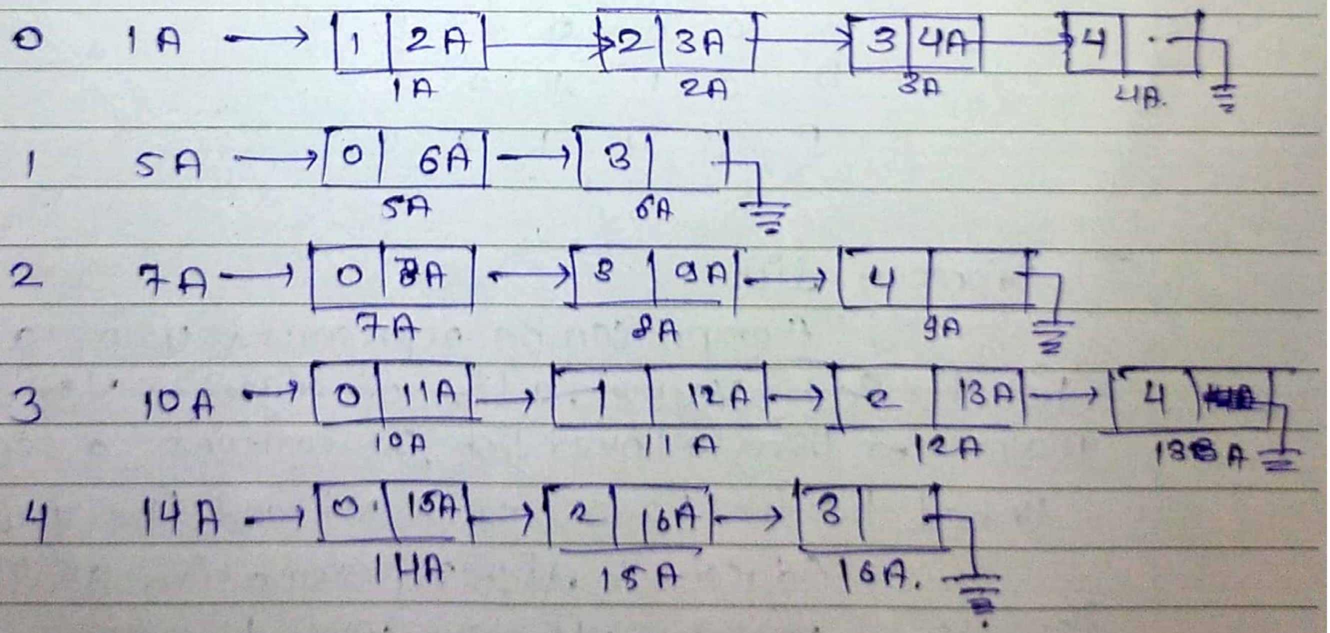
for an Undirectional graph will n vertices and edges, total number of nodes will be n size

a graph can be represented using a structure as defined below

```
# define max 30.
{
        node * next
        int vertex.
}
    node * head[max]
```

for e.g.:




0 1A → 1 | 2A → 2 | 3A → 3 | 4A → 4 | ·
1A 2A 3A 4A.
1 5A → 0 | 6A → 3 |
5A 6A
2 7A → 0 | 8A → 3 | 9A → 4 |
7A 8A 9A
3 10A → 0 | 11A → 1 | 12A → 2 | 13A → 4 |
10A 11A 12A 13A
4 14A → 0 | 15A → 2 | 16A → 3 |
14A. 15A 16A.

* Algorithm :

i) start
ii) declare member and member funcion.
iii) Take data member and member function, as per our need.
iv) Enter the vertex data value for matrix
v) show message adjancey vertex to i and f.
node is present then represented by
vi) Enter data.
vii) Repeat step 4 until user not enter n.
viii) If user enter then stop entering data.
ix) display data that user can enter.
x) stop.

* Conclusion :

Hence, we studied and implemented the concept of graph and adjancey list.