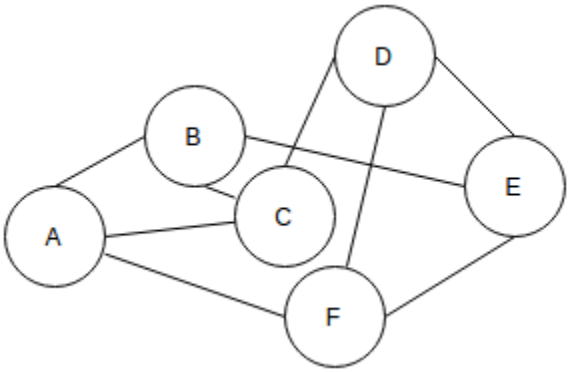
Team No.: 8

Cégep de la Gaspésie et des Îles

Montreal Campus

Final Project: Graphs

Major: Unweighted & Unidirectional Graphs



Team Members:

Dinesh Nanda – 1893551

Sagar Soni – 1893829

Jagdeep sharma - 1893739

Guided By:

Prof. Olivier Melançon

Subject: ALGORITHM, PSEUDO-CODE AND DESIGN

Class: 309

Course: Mobile Application Development (MAD)

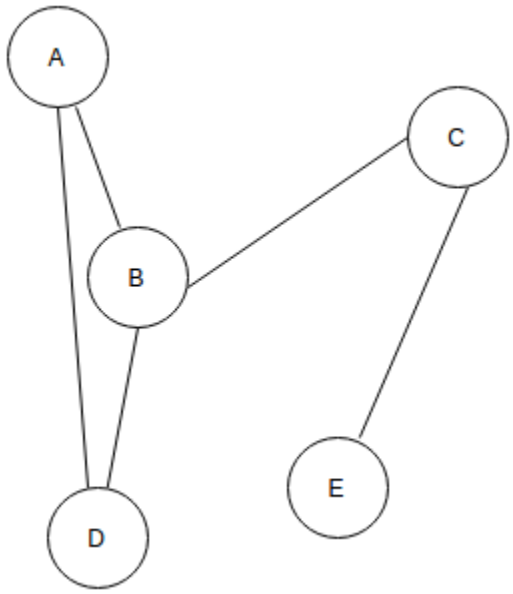
❖ What is Graph?

A Graph G is an ordered pair of a set of **V** of vertices and a set **E** of edges.

G = (V, E)

A **Graph** is pictorial representation of set of objects where some pairs of objects are connected by links. The interconnected objects are represented by points termed as vertices, and the links that connect the vertices are called edges.

Formally, a graph is a pair of sets **(V, E)**, where **V** is the set of vertices and **E** is the set of edges, connecting the pairs of vertices. Take a look at the following graph –



❖ Types of Solutions in Graphs:

There is two types of solutions are available in Graphs,

I. Adjacency List
II. Adjacency Matrix

➢ Adjacency List:

In the above graph using **List**,

V = {A, B, C, D, E}

E = {AB, AD, BD, BC, CE}

➢ Adjacency Matrix:

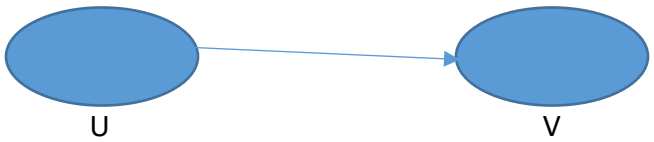
In the above graph using **Matrix**,

	A	**B**	**C**	**D**	**E**
A	0	1	0	1	0
B	1	0	1	1	0
C	0	1	0	0	1
D	1	1	0	0	0
E	0	0	1	0	0

❖ Types Of Edges:

I. Ordered Pair
II. Unordered Pair

- Ordered Pair:

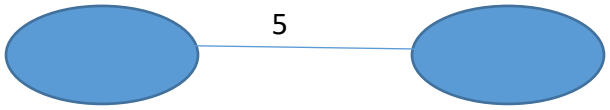


- Unordered Pair:

❖Types Of Graphs:

I. Weighted Graph
II. Unweighted Graph

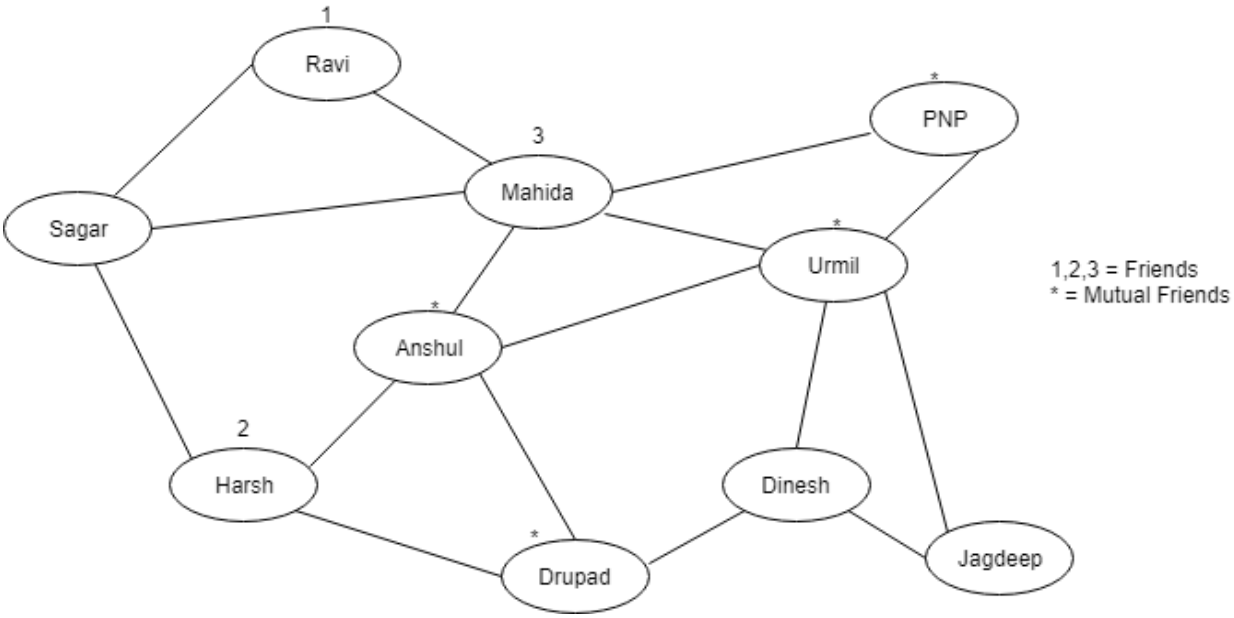
➢ Weighted Graph:



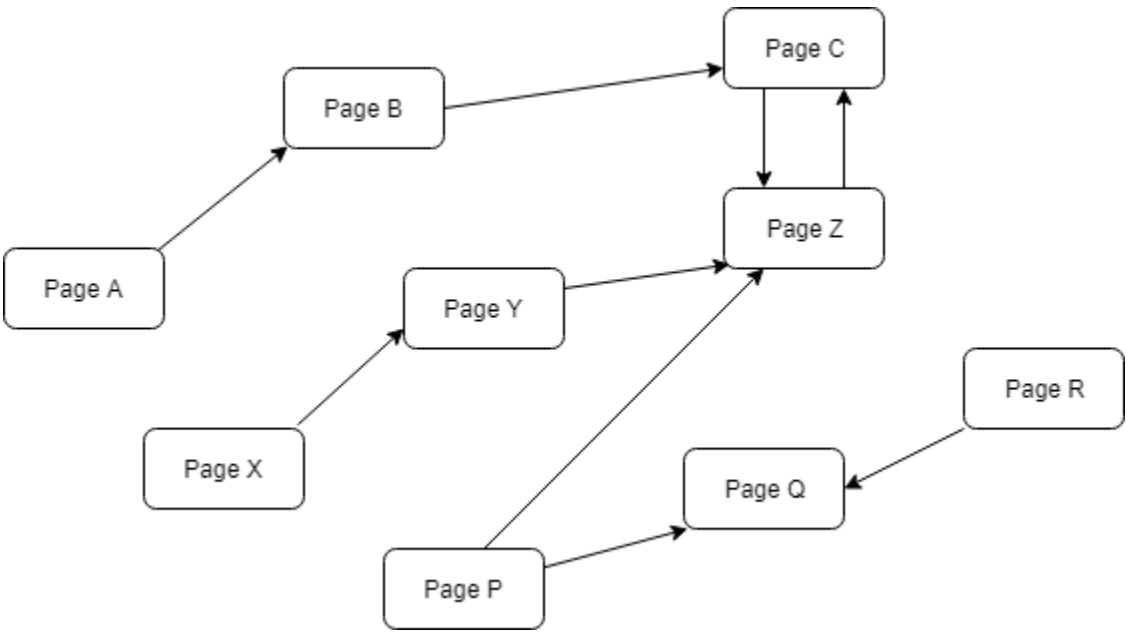
➢ Unweighted Graph:

❖ Real Life Examples of Graphs:

➢ Social Media(Face Book): Unweighted Undirected Graph



➢ World Wide Web: Unweighted Directed Graphs



❖Graph Data Structure Understanding:

Mathematical graphs can be represented in data structure. We can represent a graph using an array of vertices and a two-dimensional array of edges. Before we proceed further, let's familiarize ourselves with some important terms −

- **Vertex** – Each node of the graph is represented as a vertex. In the following example, the labeled circle represents vertices. Thus, A to G are vertices. We can represent them using an array as shown in the following image. Here A can be identified by index 0. B can be identified using index 1 and so on.
- **Edge** – Edge represents a path between two vertices or a line between two vertices. In the following example, the lines from A to B, B to C, and so on represents edges. We can use a two-dimensional array to represent an array as shown in the following image. Here AB can be represented as 1 at row 0, column 1, BC as 1 at row 1, column 2 and so on, keeping other combinations as 0.
- **Adjacency** – Two node or vertices are adjacent if they are connected to each other through an edge. In the following example, B is adjacent to A, C is adjacent to B, and so on.
- **Path** – Path represents a sequence of edges between the two vertices. In the following example, ABCD represents a path from A to D.

❖ BFS Pseudocode:

```
Function BFS (G, s):                //Where G is the graph and s is the source node
    Q = empty queue
    Q.enqueue(s)      //Inserting s in queue until all its neighbor vertices are marked. //Q = [s]
    mark s as visited
    While (Q is not empty):
        //Removing that vertex from queue, whose neighbor will be visited now
        v = Q.dequeue ( )        // v = s
        //processing all the neighbors of v
        for all neighbors w of v in Graph G:
            if w is not visited:
                Q.enqueue (w)  //Stores w in Q to further visit its neighbor // Q = [w]
                mark w as visited
```

❖ References:

1.) https://www.tutorialspoint.com/data_structures_algorithms/graph_data_structure.htm
2.) https://github.com/joeyajames/Python/blob/master/graph_adjacency-list.py