UNIVERSITY OF CALOOCAN CITY
COMPUTER ENGINEERING DEPARTMENT

Data Structure and Algorithm

Laboratory Activity No. 10

Intro to Graphs

Submitted by:
Enverzo, Kyle Andrey D.

Instructor:
Engr. Maria Rizette H. Sayo

October 11, 2025

I. Objectives

Introduction

A graph is a visual representation of a collection of things where some object pairs are linked together. Vertices are the points used to depict the interconnected items, while edges are the connections between them. In this course, we go into great detail on the many words and functions related to graphs.

An undirected graph, or simply a graph, is a set of points with lines connecting some of the points. The points are called nodes or vertices, and the lines are called edges.

A graph can be easily presented using the python dictionary data types. We represent the vertices as the keys of the dictionary and the connection between the vertices also called edges as the values in the dictionary.

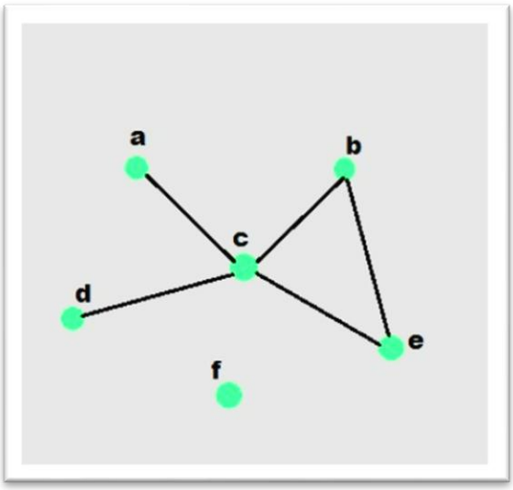

Figure 1. Sample graph with vertices and edges

This laboratory activity aims to implement the principles and techniques in:

- To introduce the Non-linear data structure – Graphs
- To discuss the importance of Graphs in programming

II. Methods

A. Discuss the following terms related to graphs:

1. Undirected graph
2. Directed graph
3. Nodes
4. Vertex
5. Degree
6. Indegree
7. Outdegree
8. Path
9. Cycle
10. Simple Cycle

III. Results

1. Undirected Graph – An undirected graph is a type of graph where the edges have no direction, meaning the connection between two vertices goes both ways. If vertex A is connected to vertex B, then B is also connected to A.

2. **Directed Graph** – A directed graph is a graph where each edge has a specific direction, showing a one-way relationship between vertices. If there is an edge from A to B, you can go from A to B but not necessarily from B to A.

3. **Nodes** – Nodes, also called vertices, are the individual points in a graph that represent data, objects, or locations. For example, in a social network graph, each node can represent a person.

4. **Vertex** – A vertex is a single node in the graph. It is one of the fundamental units that form the structure of a graph.

5. **Degree** – The degree of a vertex refers to the number of edges connected to it. In an undirected graph, it is simply the number of adjacent vertices.

6. **Indegree** – Indegree is the number of incoming edges to a vertex in a directed graph. It shows how many vertices are pointing toward that particular vertex.

7. **Outdegree** – Outdegree is the number of outgoing edges from a vertex in a directed graph. It shows how many vertices a particular vertex points to.

8. **Path** – A path is a sequence of connected vertices that shows a route between two or more points in a graph. For example, a path from A to D could be A → B → D.

9. **Cycle** – A cycle is a path that starts and ends at the same vertex, forming a loop. It shows that there is a circular connection among vertices.

10. **Simple Cycle** – A simple cycle is a cycle that does not repeat any vertex or edge except for the starting and ending vertex. It is the shortest possible circular path without duplication.

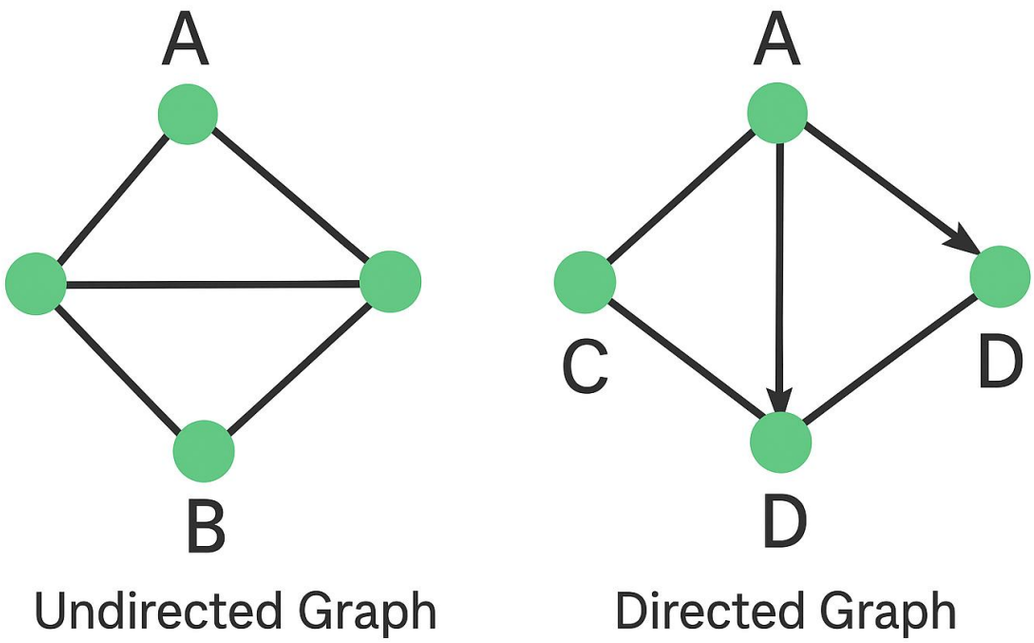

Figure 1 Screenshot of Graph

Following figure explains the distinction between Undirected and Directed Graph.

The edges in the Undirected Graph have no arrows, so the vertices are connected bidirectionally. Directed Graph : In Directed graph, edges are represented by arrows which means there is a direction on how vertices are connected with each other. Undirected graphs depict mutual relationships, directed graphs indicate one-way or directional relationship.

IV. Conclusion

In conclusion, the discussion helped me understand important graph concepts such as vertices, edges, degree, path, and cycle, which describe how elements in a graph are connected. The graphs I made — the undirected and directed graphs — clearly show the difference between two-way and one-way connections, helping me visualize how relationships work in various systems and applications.

References

[1] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd ed. Cambridge, MA: MIT Press, 2009.

[2] D. B. West, *Introduction to Graph Theory*, 2nd ed. Upper Saddle River, NJ: Prentice Hall, 2001.

[3] R. Sedgewick and K. Wayne, *Algorithms*, 4th ed. Boston, MA: Addison-Wesley, 2011.

[4] J. A. Bondy and U. S. R. Murty, *Graph Theory with Applications*. New York, NY: Elsevier, 1976.

[5] G. Chartrand and P. Zhang, *A First Course in Graph Theory*. Mineola, NY: Dover Publications, 2012.