Data Structure and Algorithm

Laboratory Activity No. 10

Intro to Graphs

|  |  |
| --- | --- |
| *Submitted by:* | *Instructor:* |
| Aquino, Jester J. | Engr. Maria Rizette H. Sayo |

October 11, 2025

# 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.

A diagram of a triangle with green dots

AI-generated content may be incorrect.

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

# Methods

* 1. 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

# Results

1. Undirected Graph - Like a family tree, it's a simple tree where the parent-child ties aren't always one way.

Examples:

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

2. Directed Graph - A directed graph (or digraph) is a group of nodes connected by edges (or arcs) in a specific direction.

Examples:

V = {A → C}

V = {C → A}

V = {B → C}

V = {C → B}

V = {B → E}

V = {E → B}

V = {C → D}

V = {D → C}

V = {C → E}

V = {E → C}

3. Node - is a fundamental component of a data structure, such as a tree or linked list, that consists of data and one or more links to other nodes, often implemented as pointers.

Examples:

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

4. Vertex - is a fundamental unit that is used to show a particular point on a graph. It is an essential component of has edges that link it to other vertices and stores data.

Examples:

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

5. Degree - The number of edges pointing in a vertex's direction is known as its in-degree, and the number of edges pointing in the opposite direction is known as its out-degree.

Example:

A = 1

B = 2

C = 4

D = 1

E = 2

F = 0

6. Indegree - The count of incoming connections from other vertices is represented by the

indegree of a vertex, which is the number of edges that point to it.

Examples:

A = 1

B = 0

C = 2

D = 1

E = 2

F = 0

7. Outdegree - is used for showing the number of edges that originate from a certain vertex. This notion, It is important for analyzing the connectivity and flow in directed networks since it counts the number of connections a given vertex has pointing away from it.

Examples:

A = 1

B = 2

C = 2

D = 0

E = 0

F = 0

8. Path - is defined as a set of edges that connect a set of vertices to allow for traversal.   
in between them. In formal terms, it can be described as a set of vertices where every pair of vertices is adjacent, or as a set of edges where each edge is incident to the next. With the possible exception of the start and last vertices of a closed path, a path is considered simple if it has no repetitions of vertices or edges.

Examples:

A → D

V = {(A, C), (C, D)

D → E

V = {(D, C), (C, E)}

A → E

V = {(A. C), (C, E),

V = {(A, C), (C, B), (B, E)}

B → D

V = {(B, C), (C, D)}

E → A

V = {(E, B), (B, C), (C, E)}

9. Cycle - is a path that creates a closed loop without repeatedly crossing any edges,   
starting at the same vertex and finishing there.  
No cycle based on graph

10.Simple Cycle - a cycle in which each path vertex is different and only the start   
as well as repeated ending vertices. So, a closed path that begins and ends at the same vertex without returning to any other vertex is called a simple cycle.

No simple cycle based on graph

# Conclusion

This activity helped me understand the basic concepts of nodes, vertices, degrees, paths, and how direction affects the structure of a graph.

**References**

[1] Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduction to Algorithms* (3rd ed.). MIT Press. [cs.mcgill.ca/~akroit/math/compsci/Cormen Introduction to Algorithms.pdf](https://www.cs.mcgill.ca/~akroit/math/compsci/Cormen%20Introduction%20to%20Algorithms.pdf)

[2] Rosen, K. H. (2012). *Discrete Mathematics and Its Applications* (7th ed.). McGraw-Hill Education.

[lib.zu.edu.pk/ebookdata/Engineering/Cyber Security/Discrete mathematics and its applications-BY Kenneth H. Rosen -McGraw-Hill (2013).pdf](https://lib.zu.edu.pk/ebookdata/Engineering/Cyber%20Security/Discrete%20mathematics%20and%20its%20applications-BY%20Kenneth%20H.%20Rosen%20-McGraw-Hill%20(2013).pdf)

[3] NetworkX Developers. (2025). *NetworkX Documentation – Graph Theory and Analysis in Python*. Retrieved from [Tutorial — NetworkX 3.5 documentation](https://networkx.org/documentation/stable/tutorial.html)