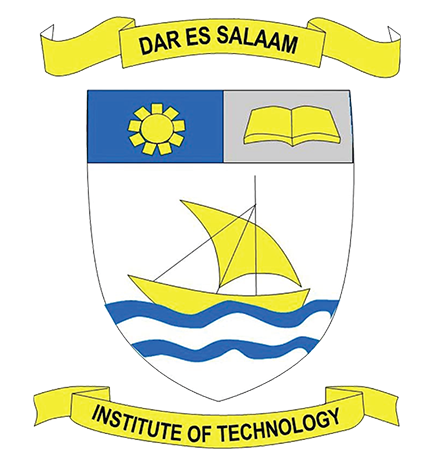
**DAR ES SALAAM INSTITUTE OF TECHNOLOGY (DIT)**

****

**DEPARTMENT OF COMPUTER STUDIES**

**ORDINARY DIPLOMA IN COMPUTER ENGINEERING**

**DATA STRUCTURE FOR TECHNICIANS**

**GROUP ASSIGNMENT**

|  |  |  |
| --- | --- | --- |
| **S/N** | **GROUP MEMBERS** | **REGISTRATION NO** |
| **1** | **FEISAL KHERI AHAMAD** | **220222498749** |
| **2** | **MISSIKE MLOLI KAMONJA** | **220222398442** |

**Introduction**

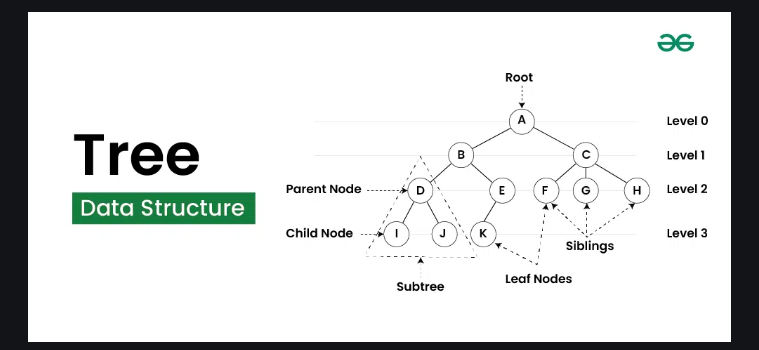
In computer science, trees and graphs are two fundamental data structures used to represent relationships between objects. While they share some similarities, they also have distinct differences that make them suitable for different applications.

Tree Data Structure

A tree is a widely used abstract data type that represents a hierarchical tree structure with a set of connected nodes

. Each node in the tree can be connected to many children (depending on the type of tree), but must be connected to exactly one parent, except for the root node, which has no parent

. These constraints mean there are no cycles or "loops" (no node can be its own ancestor), and also that each child can be treated like the root node of its own subtree, making recursion a useful technique for tree traversal



Key Characteristics of Trees

Root Node: The topmost node in the tree, which has no parent

Parent Node: The node that is the predecessor of any node

Child Node: A node that is connected below the current one

Leaf Node: A node that has no children

Depth/Level: The length of the path (edges) from the root to a node

Tree Height: The maximum depth from of any node in the tree

Types of Trees

Binary Trees: A tree where each node has at most two children

Binary Search Trees (BSTs): A binary tree with no duplicate nodes that imposes an ordering on its nodes

Graph Data Structure

A graph is a data structure that contains a set of vertices and a set of edges which connect pairs of the vertices

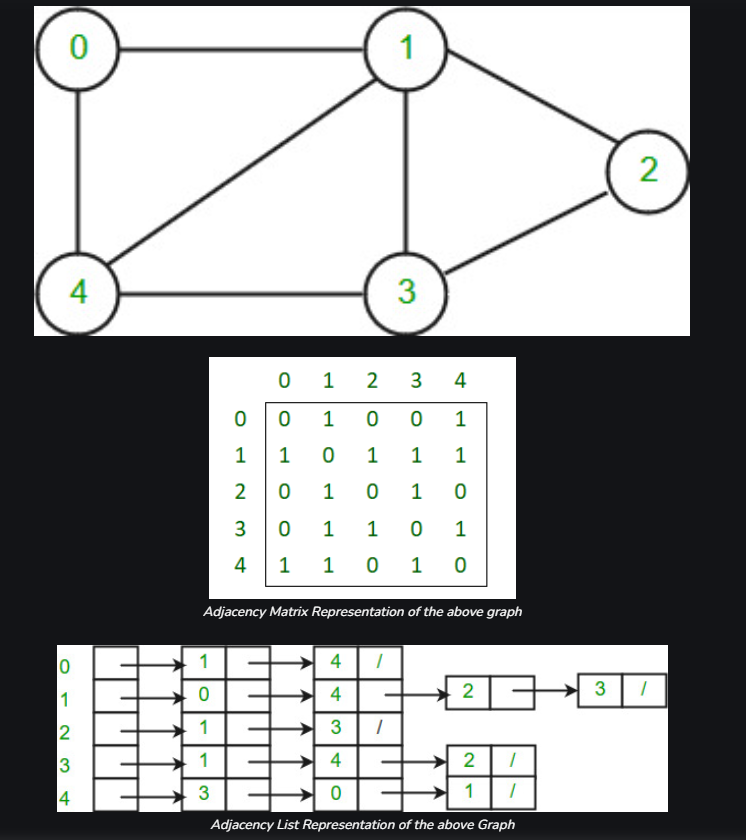
. A vertex (or node) can be connected to any number of other vertices using edges

. An edge may be bidirectional or directed (one-way)

. An edge may have a weight on it that indicates a cost for traveling over that edge in the graph

. Unlike trees, graphs can contain cycles

. In fact, a tree is an acyclic graph



Key Characteristics of Graphs

Vertices (Nodes): The points in a graph that are connected by edges

Edges: The connections between vertices

Cycles: A path that starts and ends at the same vertex, passing through at least one edge more than once

Directed vs. Undirected Edges: Directed edges have a direction and represent a one-way relationship, while undirected edges do not have a direction and represent a two-way relationship

Weighted Edges: Edges with a weight or cost associated with them, often used to represent the cost or distance of traversing the edge

Differences Between Trees and Graphs

Cycles: Graphs can contain cycles, while trees cannot

Connectivity: Graphs can be disconnected (i.e., have multiple components), while trees are always connected

Hierarchy: Trees have a hierarchical structure, with one vertex designated as the root. Graphs do not have this hierarchical structure

Applications: Graphs are used in a wide variety of applications, such as social networks, transportation networks, and computer science. Trees are often used in hierarchical data structures, such as file systems and XML documents

Conclusion

In conclusion, trees and graphs are two fundamental data structures in computer science that are used to represent relationships between objects. Trees are hierarchical structures with a single root node and no cycles, while graphs are more general structures that can have cycles and multiple components. Understanding the differences between these data structures is crucial for choosing the right data structure for a particular problem or application.