Data structures define how data is organized, stored, and manipulated in a program.
They are essential for developing efficient algorithms and optimizing memory usage.
Linear data structures have elements arranged sequentially, such as arrays, stacks, queues, and linked lists.
Static data structures have a fixed memory size, like arrays.
Dynamic data structures can change size during runtime, such as queues and stacks.
Non-linear data structures do not store elements sequentially, including trees and graphs.
Arrays store elements of the same type in contiguous memory.
Matrices or grids are two-dimensional arrays arranged in rows and columns.
Strings are sequences of characters used for text manipulation.
Stacks follow the Last In, First Out (LIFO) principle and are used in function calls and memory management.
Queues follow the First In, First Out (FIFO) principle and are used in scheduling and message handling.
Linked lists consist of nodes connected by pointers, allowing dynamic memory allocation.
Hashing generates fixed-size hash values for efficient searching, password storage, and cryptography.
Trees are hierarchical, non-linear structures used in file systems and databases.
Binary trees have nodes with at most two children.
Binary search trees (BST) maintain sorted order, with left subtree values smaller than the node and right subtree values larger.
Heaps are complete binary trees used to implement priority queues.
Graphs consist of vertices (nodes) and edges (links), representing relationships between entities.
Advanced data structures like segment trees, tries, binary indexed trees, and suffix arrays optimize complex operations.