In computer science, a data structure is a way of organizing and storing data in a computer's memory or storage system. It provides a specific format and set of operations for efficiently accessing and manipulating the data.

Data structures serve as the building blocks for creating algorithms and solving various computational problems. They are designed to optimize the efficiency, speed, and space requirements of data operations such as searching, sorting, inserting, and deleting data.

Here are some commonly used data structures:

Arrays: Arrays are a collection of elements stored in contiguous memory locations. They provide constant-time access to individual elements but have fixed sizes and may require shifting elements when inserting or deleting.

Linked Lists: Linked lists consist of nodes, where each node holds data and a reference to the next node in the list. They allow dynamic memory allocation, efficient insertion and deletion operations, but require traversal for accessing specific elements.

Stacks: Stacks follow the Last-In-First-Out (LIFO) principle. Elements can only be inserted or removed from the top of the stack, which makes it useful for tracking function calls, managing recursion, or implementing undo operations.

Queues: Queues follow the First-In-First-Out (FIFO) principle. Elements can only be inserted at the back and removed from the front of the queue. They are commonly used for scheduling tasks, managing waiting lists, or implementing breadth-first search algorithms.

Trees: Trees are hierarchical data structures composed of nodes, with a root node at the top and child nodes branching out from it. Common types include binary trees, binary search trees, and balanced trees like AVL or red-black trees. Trees enable efficient searching, sorting, and hierarchical representation of data.

Graphs: Graphs consist of nodes connected by edges. They are used to represent relationships and connections between elements. Graphs can be directed or undirected, weighted or unweighted, and have various applications in network analysis, social networks, and routing algorithms.

Hash Tables: Hash tables (or hash maps) use a hash function to map keys to values. They provide constant-time average-case access and insertion but can have collisions that require handling. Hash tables are efficient for lookups and indexing large amounts of data.

These are just a few examples of data structures, and there are many more variations and advanced data structures available based on specific requirements and problem domains. Choosing the right data structure depends on the nature of the data, the operations to be performed, and the desired efficiency and performance characteristics.