Data structures can implement one or more operations, such as insert, delete, search, and traversal, on the data. These operations are crucial for solving problems efficiently. The choice of data structure depends on the specific requirements of the problem and the characteristics of the data. For instance, if you need fast access to any element in a collection, a hash table might be the best choice. On the other hand, if you need to perform frequent insertions and deletions, a linked list might be more suitable.

Some commonly used data structures include:

1. Arrays: A contiguous block of memory that stores elements of the same type. Arrays are simple but limited in their ability to handle dynamic data sizes.
2. Linked Lists: A linear collection of data elements, where each element (node) contains a reference (pointer) to the next element in the sequence.
3. Stacks: A Last-In-First-Out (LIFO) collection where elements are added or removed from the top.
4. Queues: A First-In-First-Out (FIFO) collection where elements are added at the rear and removed from the front.
5. Trees: Hierarchical structures that represent a collection of elements and their relationships.
6. Heaps: Ordered trees, typically implemented using an array, that maintain two properties: min heap or max heap.
7. Hash Tables: Data structures that use a hash function to map keys to indexes in an array, allowing for quick access to data.

Understanding these data structures is fundamental for any programmer, as they form the building blocks of efficient algorithms. By mastering data structures, you can write programs that are not only correct but also performant.