Array

By Mani Abedii

View Full DSA Repository

A clear, compact guide covering essential data structures and algorithms, designed for easy understanding and quick learning.

An **array** is a collection of elements stored at contiguous memory locations. All elements are of the same data type and can be accessed using an index (starting at 0 in most languages).

Characteristics of an Array:

- **Fixed Size:** You must define the size when the array is created which means the system allocates a contiguous block of memory upfront.
- **Indexed access:** Random access allows O(1) time to access any element by index.
- **Efficient:** Arrays are space and time efficient for many use cases.

Arrays are the foundation of many other structures like stacks, queues, matrices, hash tables, etc.

Array Operations:

All the following operations have been implemented in Java in this project.

1. **Access:** Retrieves any element instantly (in O(1) time) using its index, thanks to the contiguous memory layout.

2. **Insert:** Inserts an element at a specific index which requires shifting all elements after that index one position to the right to make room. So, this operation takes O(n) time.

3. **Delete:** Removes an element at a given index and shifts all subsequent elements left to fill the gap which takes O(n) time.

4. **Update:** Overwrites a value at a particular index in O(1) time.

5. **Search:** locates a specific element and returns its index if found, or -1 if not.
   - **Linear Search:** Check each element one by one to find a match.
     - Best for unsorted arrays or small data bases.
     - O(n) time complexity.

   - **Binary Search:** Repeatedly divide the sorted array in half to find the target value.
     - Only works on sorted arrays. But still, much faster for large datasets.
     - O(log n) time complexity.

© 2025 Mani Abedi