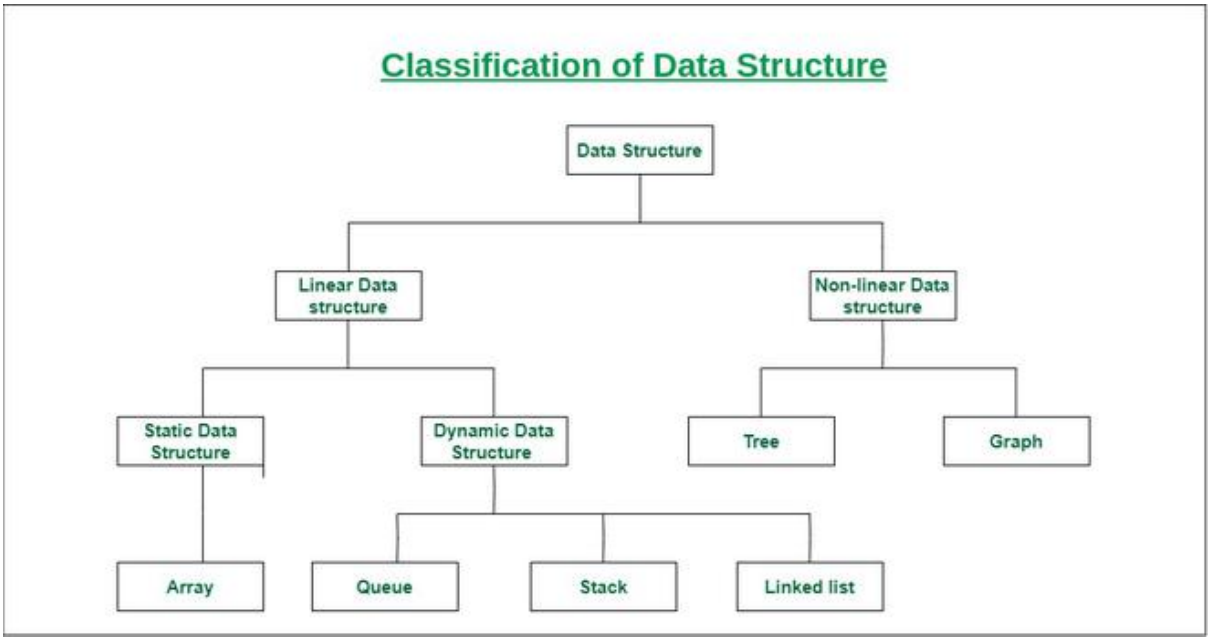
Introduction to Data Structures

Definition of Data Structures

- Data structures are fundamental concepts in computer science that involve the
  - Organization
  - Storage
  - Manipulation of data
- They provide a systematic way to store and manage information, making it accessible and efficient to work with.
- Data structures are the building blocks of algorithms and play a crucial role in software development.



Types of Data Structures

1) Linear Data Structures
2) Non-linear Data Structures

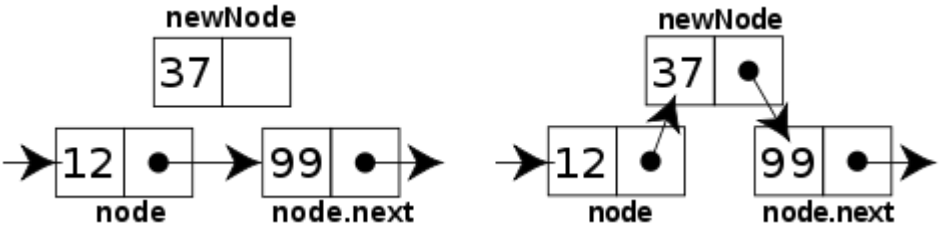
Linear Data Structures

- Linear data structures are arrangements of data elements where each element has a unique predecessor and successor, forming a sequential order.
- The following data structures are referred to as linear data structures:

a) Array: A linear collection of elements with indexed access for efficient retrieval.

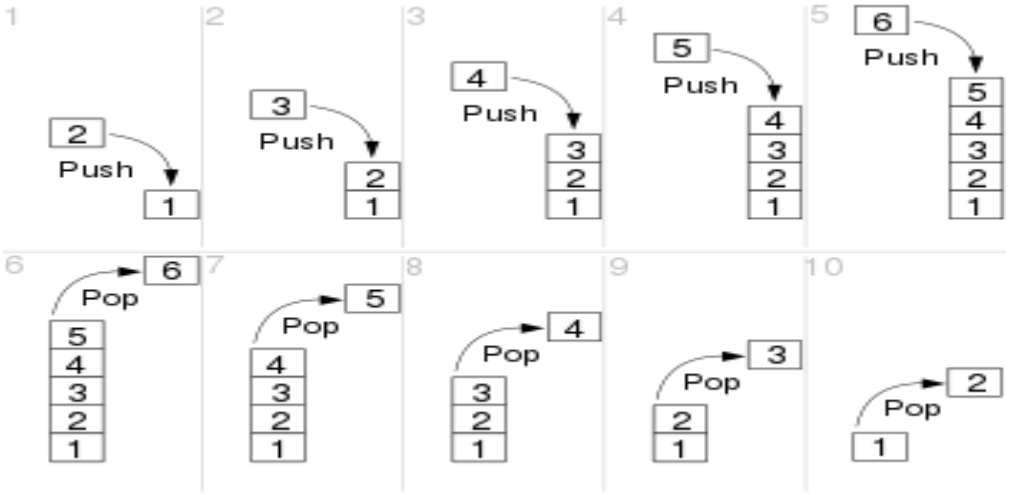
cc: Javatpoint

b) Linked List: Elements connected by pointers, allowing dynamic allocation and efficient insertions/deletions.



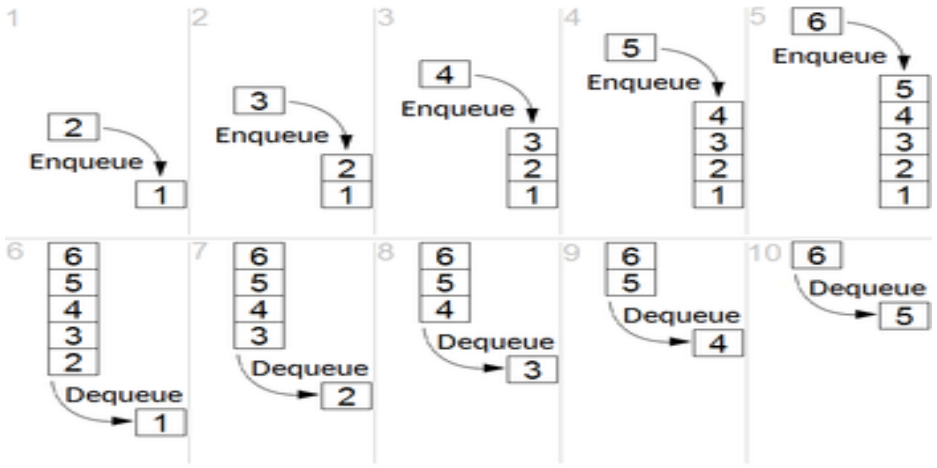
cc: Wikipedia

c) Stack: Follows the Last-In-First-Out (LIFO) principle with top-based element manipulation.



cc: Wikipedia

d) Queue: Adheres to the First-In-First-Out (FIFO) concept, used for ordered processing.



cc: Wikipedia

e) Deque: Supports insertion and removal at both ends, offering enhanced flexibility.

cc: java2novice

The Need for Linear Data Structures

- Ordered Storage: Linear data structures maintain a sequential order, which is essential for scenarios where data must be processed sequentially or accessed in a specific arrangement.
- Efficient Access: Direct indexing or traversal capabilities of linear structures allow for quick and convenient access to elements.
- Insertion and Deletion: Linear structures provide efficient methods for adding and removing elements, which is crucial for dynamic data manipulation.
- Memory Optimization: Linear structures allocate memory contiguously, optimizing memory usage and access efficiency.

Operations on Linear Data Structures

1) Access: Retrieving elements by index, position, or pointer.
2) Insertion: Adding new elements at specific positions.
3) Deletion: Removing elements from specific positions.

4) Traversal: Iterating through elements sequentially.
5) Search: Finding the position or existence of an element.
6) Update: Modifying the value of an element.
7) Sorting: Arranging elements in a specified order.
8) Merging: Combining two ordered linear structures.
9) Memory Management: Allocating and deallocating memory dynamically.

Real-world Examples of Linear Data Structure:

1) Arrays:
    a) Grocery Shopping List: Managing your shopping list with each item corresponding to an array index simplifies adding, removing, and checking off items.
    b) Image Pixels: In digital images, arrays store pixel values, allowing manipulation and editing of pictures by altering individual pixel colors.
2) Linked Lists:
    a) Music Playlist: Linked lists are suitable for creating playlists, where songs are nodes connected in a sequence, allowing easy rearrangement and modification.
    b) Train Cars: Linked lists can represent train cars linked together, enabling efficient addition and removal of cars without affecting the entire train.
3) Stacks:
    a) Undo Feature: In software applications, stacks manage to undo operations, enabling users to reverse actions in the order they were performed.
    b) Plate Stacking: Plates stacked on top of each other represent a real-world example of a stack, where the last plate placed is the first one taken.
4) Queues:
    a) Cafeteria Line: Queues model waiting in line at a cafeteria, where the first person in line is served first, maintaining order and fairness.
    b) Ticket Counter: Waiting in line to purchase tickets, like at a cinema or an event, follows the queue concept.
5) Deques (Double-Ended Queues):
    a) Sliding Glass Doors: Deques are similar to sliding glass doors at entrances, allowing people to enter or exit from both sides.
    b) Printing and Scanning: Deques mimic the process of loading and unloading papers for printing and scanning, as both ends are accessible.