Introduction to Data Structure

Name : Rupak Sarkar

Roll No.: 14271024036

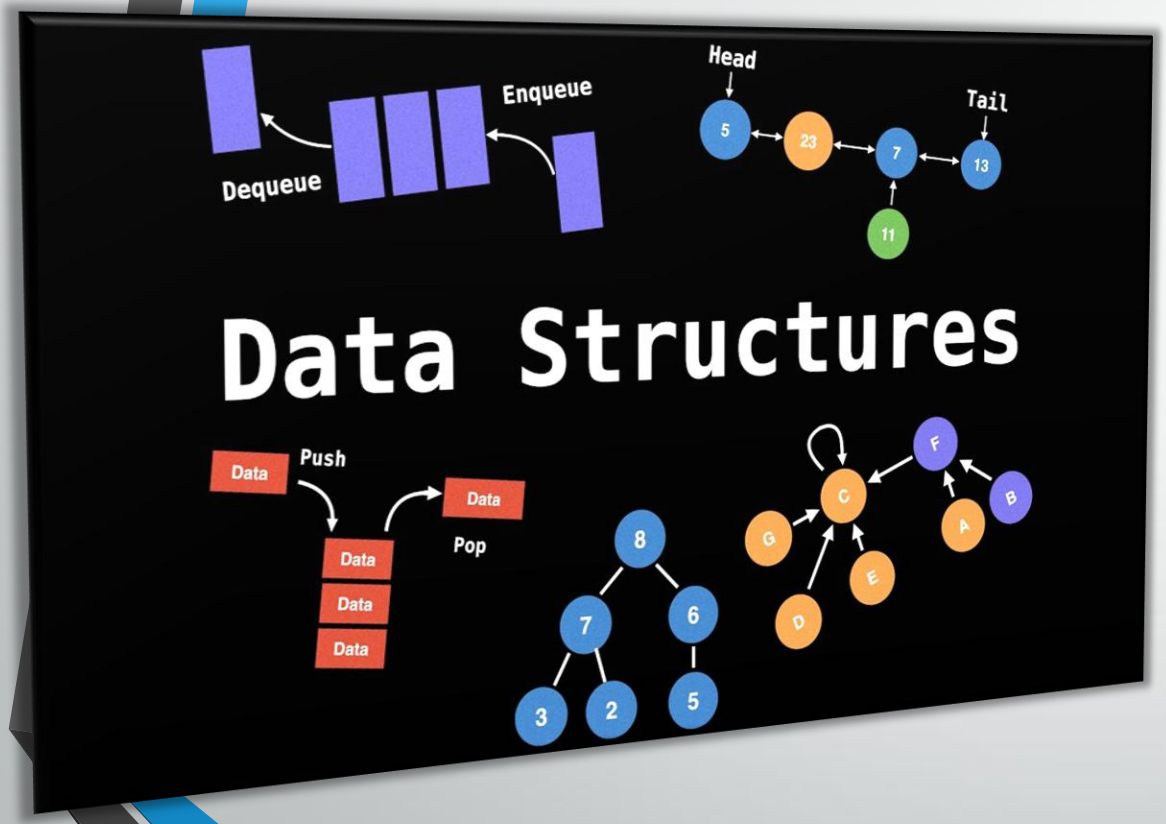
Stream : MCA

Semester : Semester 2nd

Subject : Data Structure with Python

Subject Code : MCAN-201

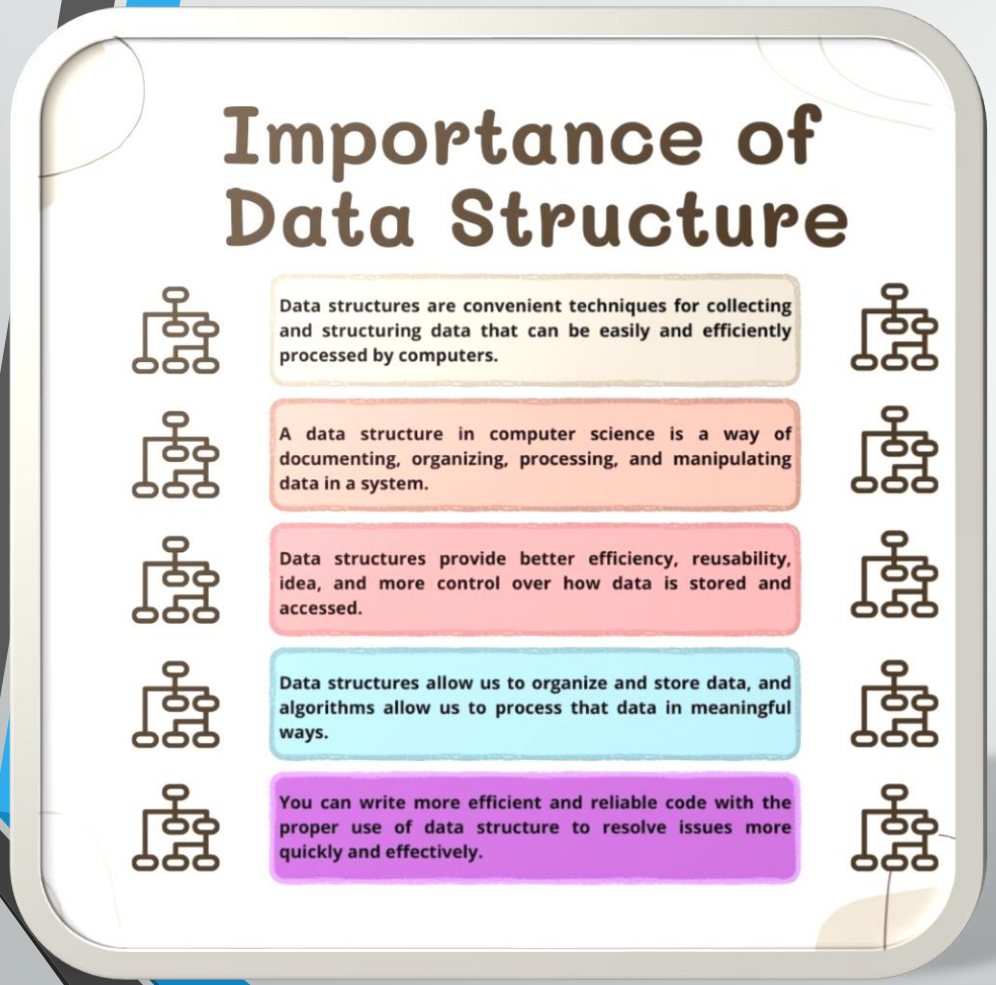
Definition of Data Structure



Data structures are the fundamental building blocks of computer programming. They define how data is organized, stored, and manipulated within a program.

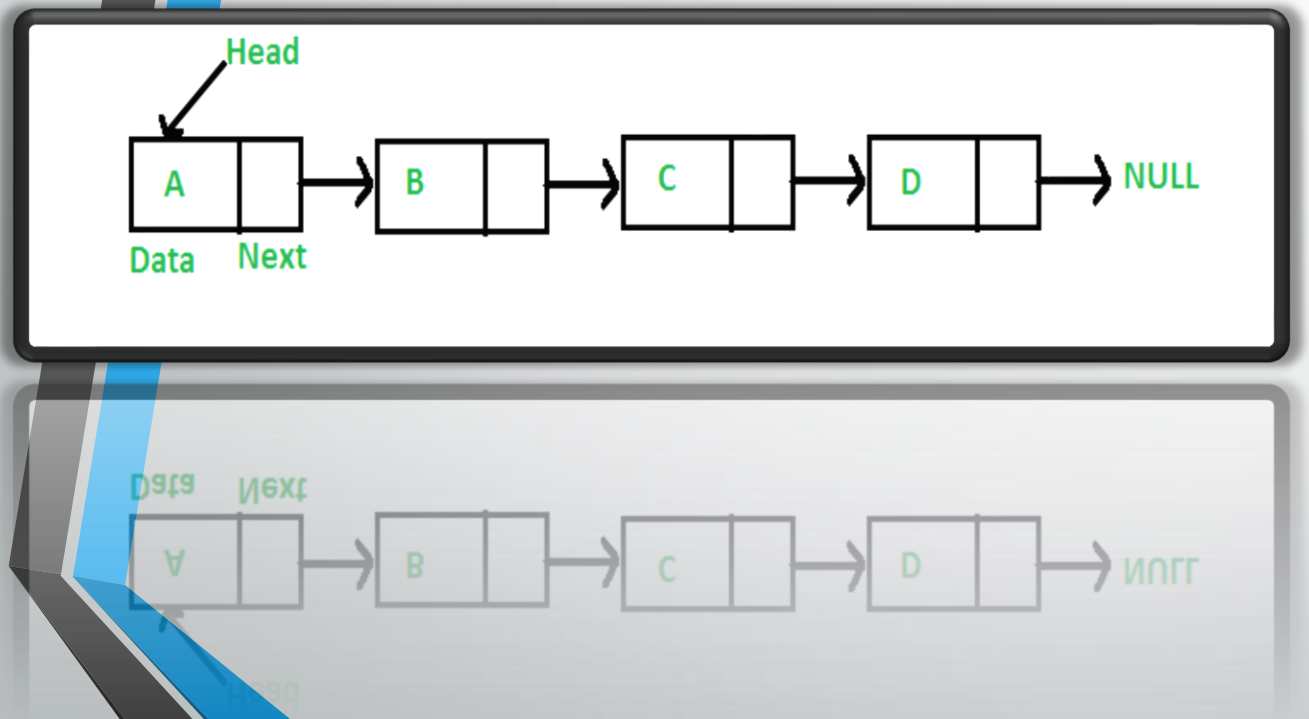
- A data structure is a storage that is used to store and organize data.
- It is a way of arranging data on a computer so that it can be accessed and updated efficiently.
- A data structure is not only used for organizing the data.
- It is also used for processing, retrieving, and storing data.

Why should we use Data Structure?



- Data Structures are structures programmed to store ordered data, so that various operations can be performed on it easily.
- A data structure in computer science is a way of documenting, organizing, processing, and manipulating data in a system.
- Data structures provide an efficient way to organize, manage, and store data.
- Data structures ensure efficiency, reusability, and abstraction.
- Data structures allow us to organize and store data, and algorithms allow us to process that data in meaningful ways.

Different types of Data Structures



Linked List

A **Linked list** is a linear data structure in which elements are not stored at contiguous memory locations.

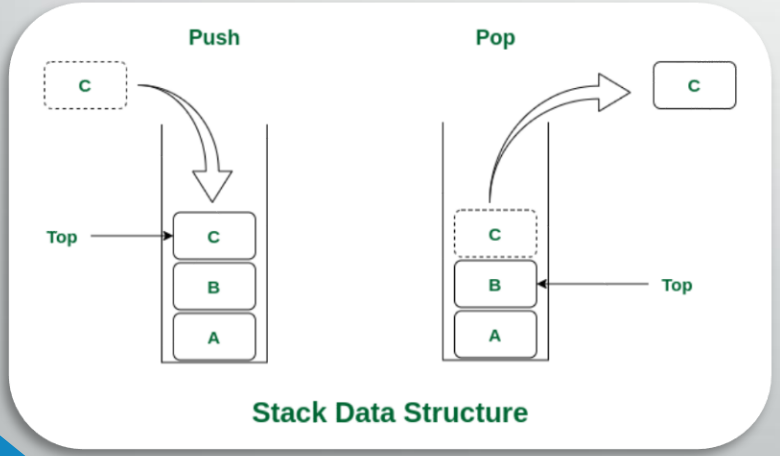
- Linked lists are used to implement other data structures like stacks, queues, etc.
- It is used for the representation of sparse matrices.
- It is used in the linked allocation of files.
- Linked lists are used to display image containers.
- Users can visit past, current, and next images.
- They are used to perform undo operations.

Different types of Data Structures

Stack

Stack is a linear data structure that follows LIFO(Last in first out) principle.

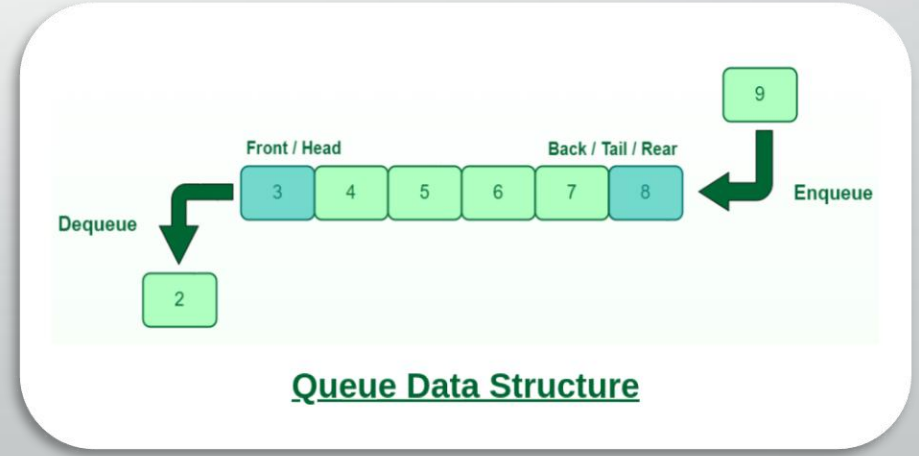
- The stack data structure is used in the evaluation and conversion of arithmetic expressions.
- It is used for parenthesis checking and string reversal.



Queue

Queue is a linear data structure that follows First In First Out(FIFO) principle.

- Queue is used for handling website traffic.
- Queues are used for job scheduling in the operating system.

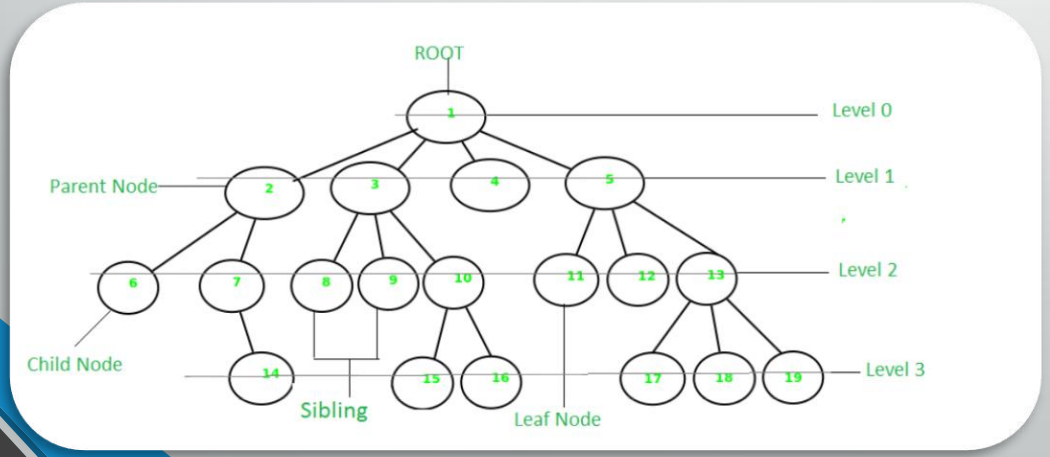


Different types of Data Structures

Tree

A **Tree** is a non-linear and hierarchical data structure where the elements are arranged in a tree-like structure.

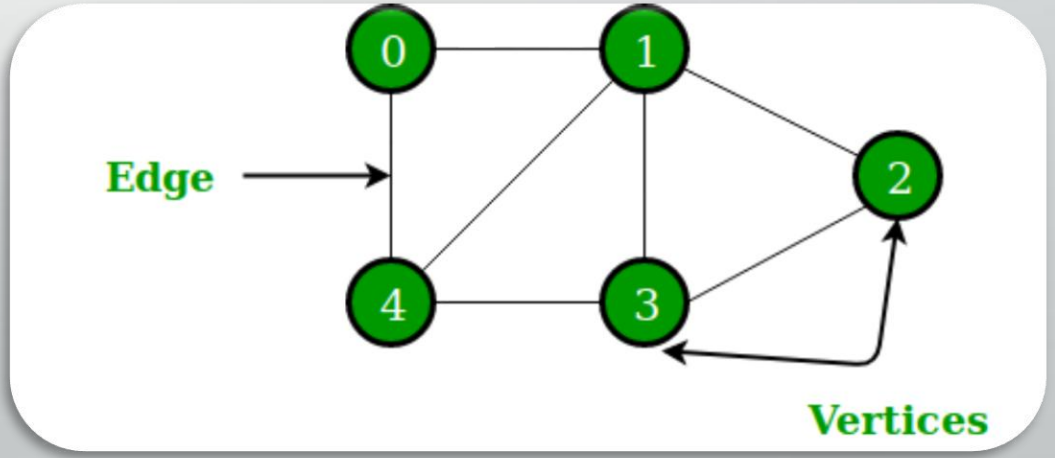
- Spanning trees are used in routers in computer networks.
- Domain Name Server also uses a tree data structure.



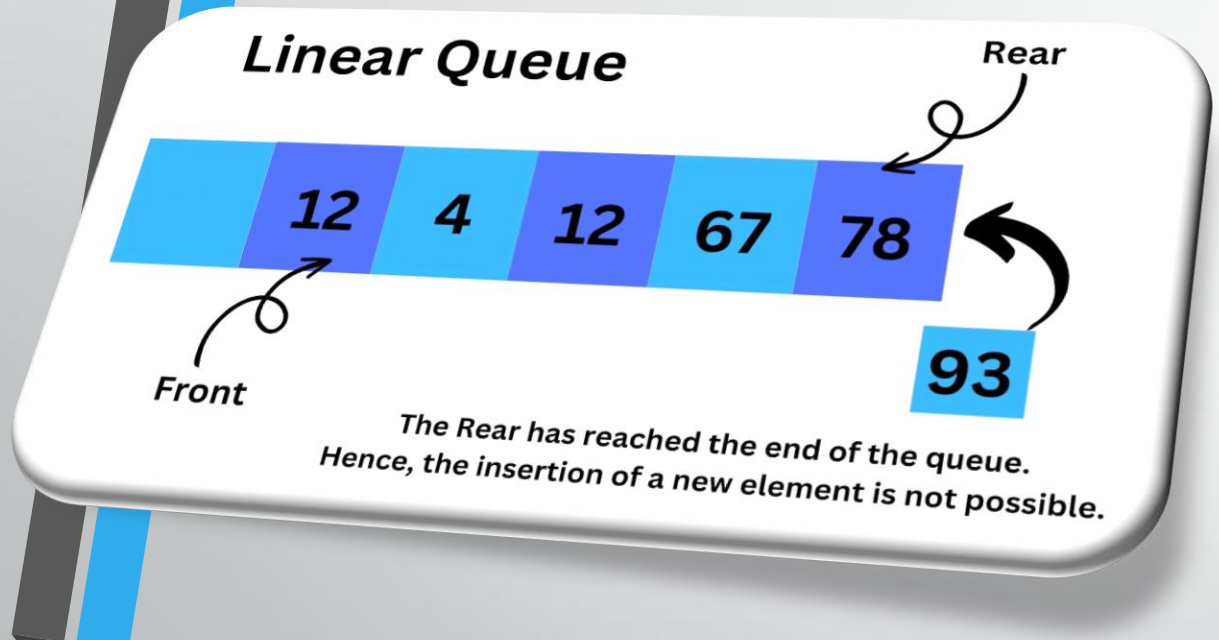
Graph

A **Graph** is a non-linear data structure that consists of vertices (or nodes) and edges.

- The operating system uses Resource Allocation Graph.
- Also used in the World Wide Web where the web pages represent the nodes.



What is Linear Queue?



A linear queue is a straightforward implementation of the queue data structure. It follows the First-In-First-Out (FIFO) principle. It is a linear data structure where elements are added at one end (the rear) and removed from the other end (the front).

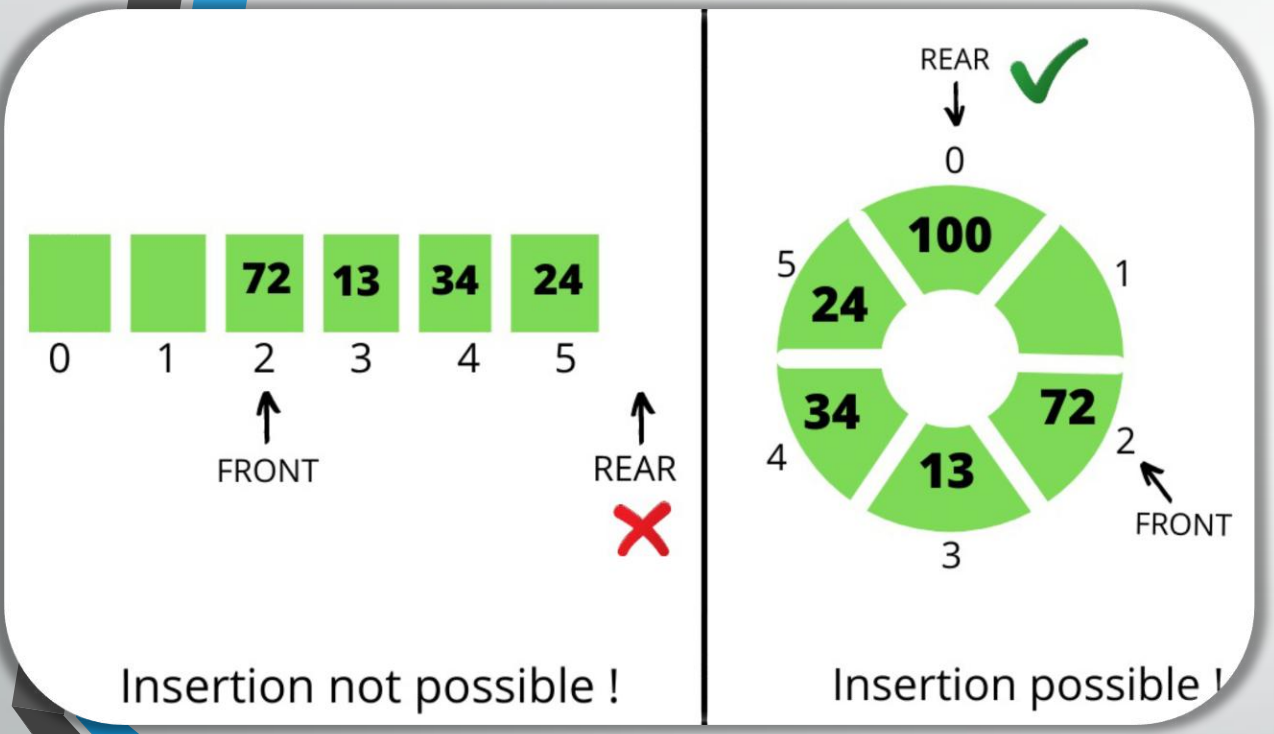
Structure: Utilizes a linear array or linked list to store elements.

Operations: Insertion (enqueue) happens at the rear, and deletion (dequeue) occurs at the front.

Index Management: Uses two pointers or indices: front and rear. The front points to the first element, and the rear points to the last element.

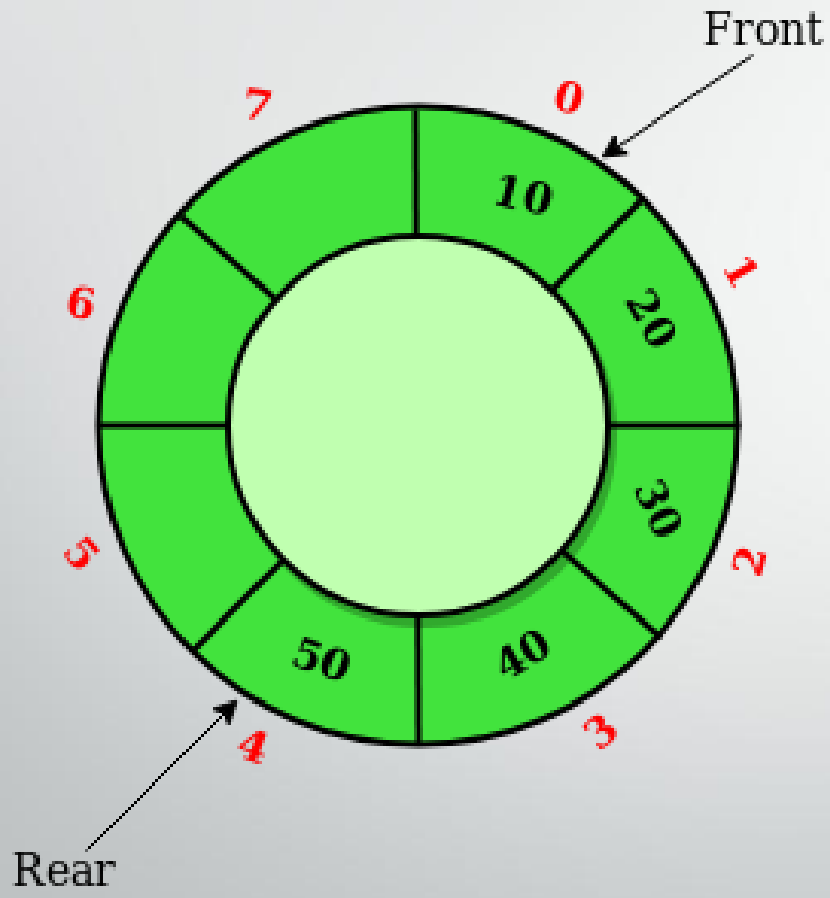
Growth: The rear index moves towards the end of the array with each addition, and the front index moves towards the end with each removal.

Drawbacks of Linear Queue



- Linear queues can be inefficient, especially when dealing with large numbers of elements.
- Unused space can accumulate at the beginning of the array, leading to wasted memory.
- The empty spaces in the front are not utilized until the queue becomes empty.
- Once the queue reaches its maximum size, it cannot accept new elements until some are removed.
- In fixed-size array implementations, the queue cannot reuse the vacated spaces.

Circular Queue and its Advantages over Linear Queue



A Circular Queue is an extended version of a normal queue where the last element of the queue is connected to the first element of the queue forming a circle.

- **Structure**: Uses a circular array or linked list to store elements.
- **Operations**: Similar to linear queues but with circular index management.
- **Index Management**: Both front and rear pointers move in a circular manner. When the rear or front reaches the end of the array, it wraps around to the beginning.
- **Efficiency**: This design ensures all available memory is utilized efficiently.

Thank You!