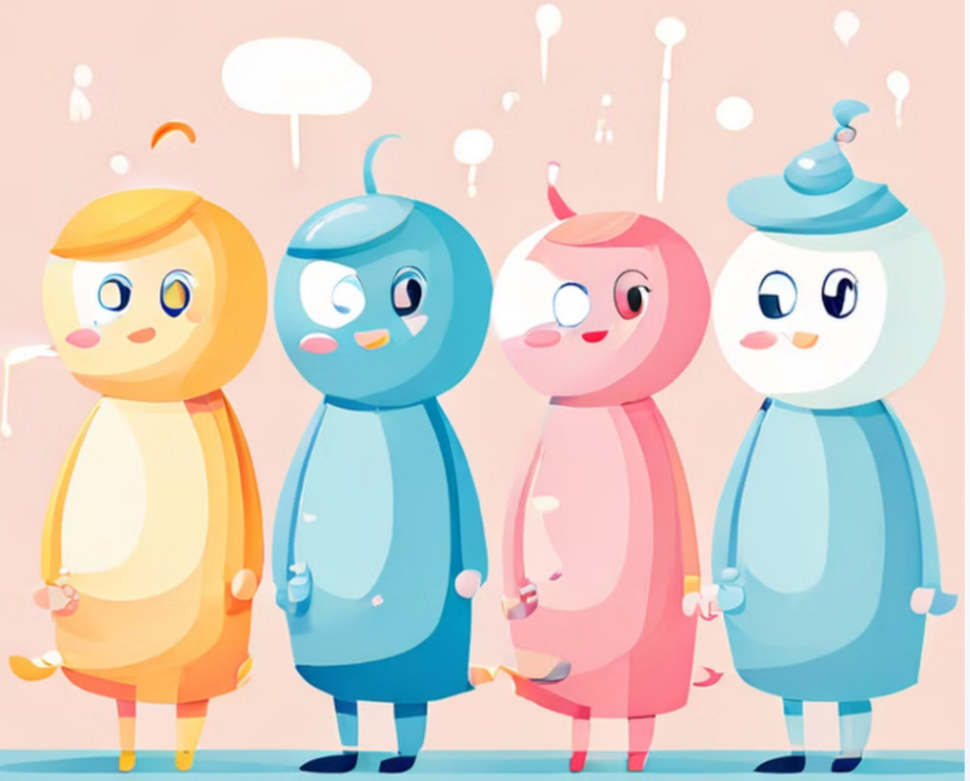

Introduction to Queue Data Structure

Queues are a fundamental data structure that follow the First-In-First-Out (FIFO) principle. They act like a real-life queue, where the first item added is the first one to be removed.

KANDLAPALLY SHIVA TEJA
227Z5A6703

FIFO (First-In-First-Out) Principle

1

Enqueue

New items are added to the end of the queue.

2

Wait in Line

Items in the queue wait their turn, just like in a real-life line.

3

Dequeue

Items are removed from the front of the queue, in the order they were added.

Enqueue and Dequeue Operations

Enqueue

Adding an item to the end of the queue.

Dequeue

Removing the item from the front of the queue.

Queue Manipulation

Queues allow you to efficiently manage the order of elements.

Applications of Queues

1 Task Scheduling

Queues are used to manage the order of tasks, ensuring they are processed in the correct sequence.

2 Event Handling

Queues are used to manage the order of events, such as user inputs or network requests.

3 Simulation and Modeling

Queues are used to simulate real-world scenarios, such as lines at a bank or traffic in a city.

4 Breadth-First Search (BFS)

Queues are used in the BFS algorithm to explore graph data structures in a systematic way.

Implementing Queues using Arrays

1

Front Pointer

Keeps track of the front of the queue.

2

Rear Pointer

Keeps track of the rear of the queue.

3

Enqueue

Adds an item to the rear of the queue.

4

Dequeue

Removes an item from the front of the queue.

Implementing Queues using Linked Lists

Node

Each node contains the data and a pointer to the next node.

Front Pointer

Keeps track of the front of the queue.

Rear Pointer

Keeps track of the rear of the queue.

Enqueue

Adds a new node to the rear of the queue.

Dequeue

Removes the node from the front of the queue.

Time Complexity of Queue Operations

Operation	Time Complexity
Enqueue	O(1)
Dequeue	O(1)
Peek (get front element)	O(1)
Empty (check if empty)	O(1)

Conclusion and Summary

Fundamental Data Structure

Queues are a fundamental data structure that follow the FIFO principle.

Efficient Operations

Queues provide efficient enqueue and dequeue operations with O(1) time complexity.

Wide Applications

Queues have a wide range of applications, from task scheduling to breadth-first search.

Flexible Implementations

Queues can be implemented using either arrays or linked lists, depending on the requirements.

THANK YOU !