ADT :

QUEUE

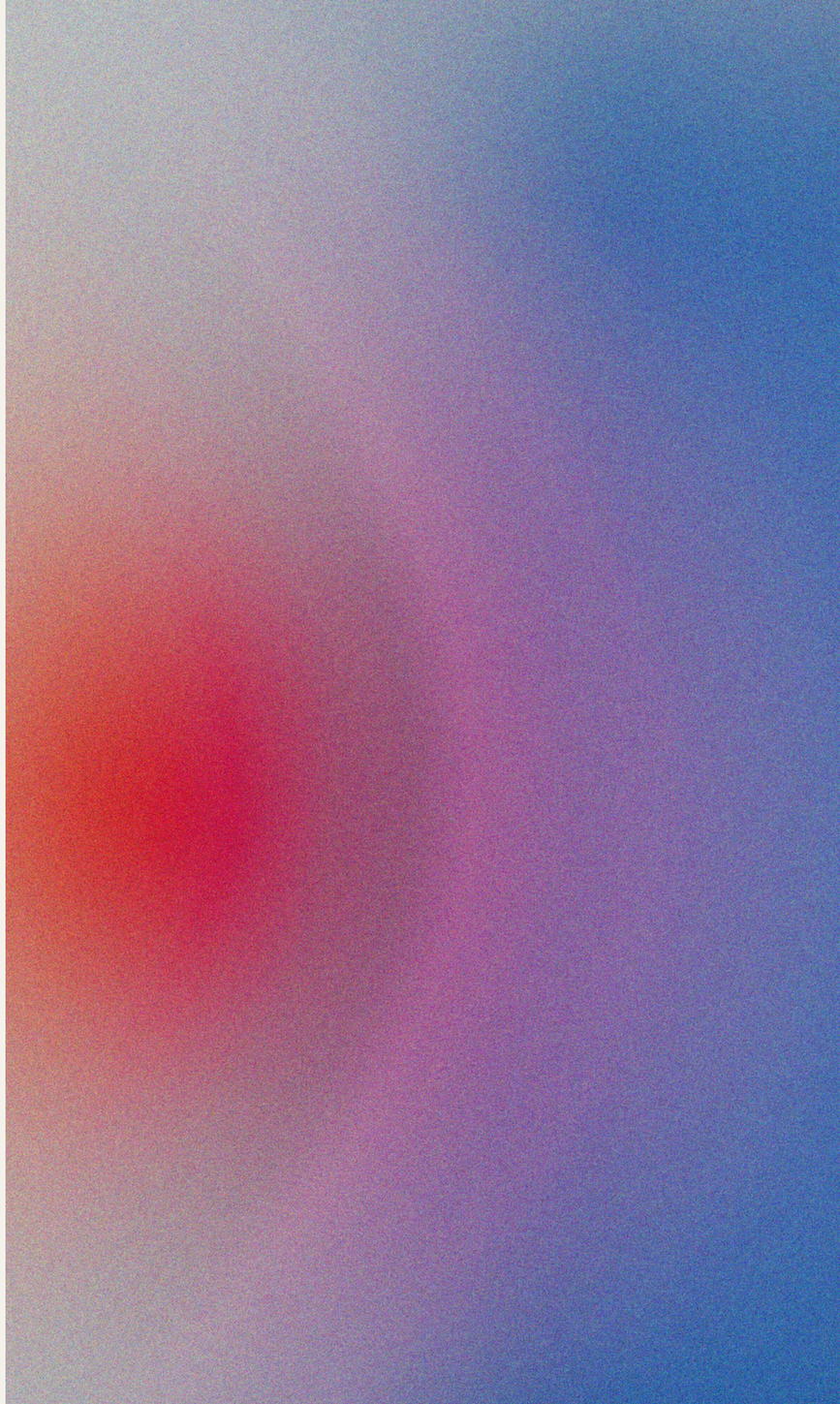
By Asma Hammedi

Table of contents

WHAT IS AN ADT ?

An Abstract Data Type is a conceptual model for how data is organized and what operations are allowed on that data. It defines what you can do with the data, not how it’s implemented.

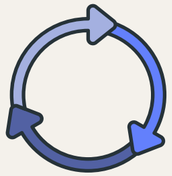
FUNDAMENTALS



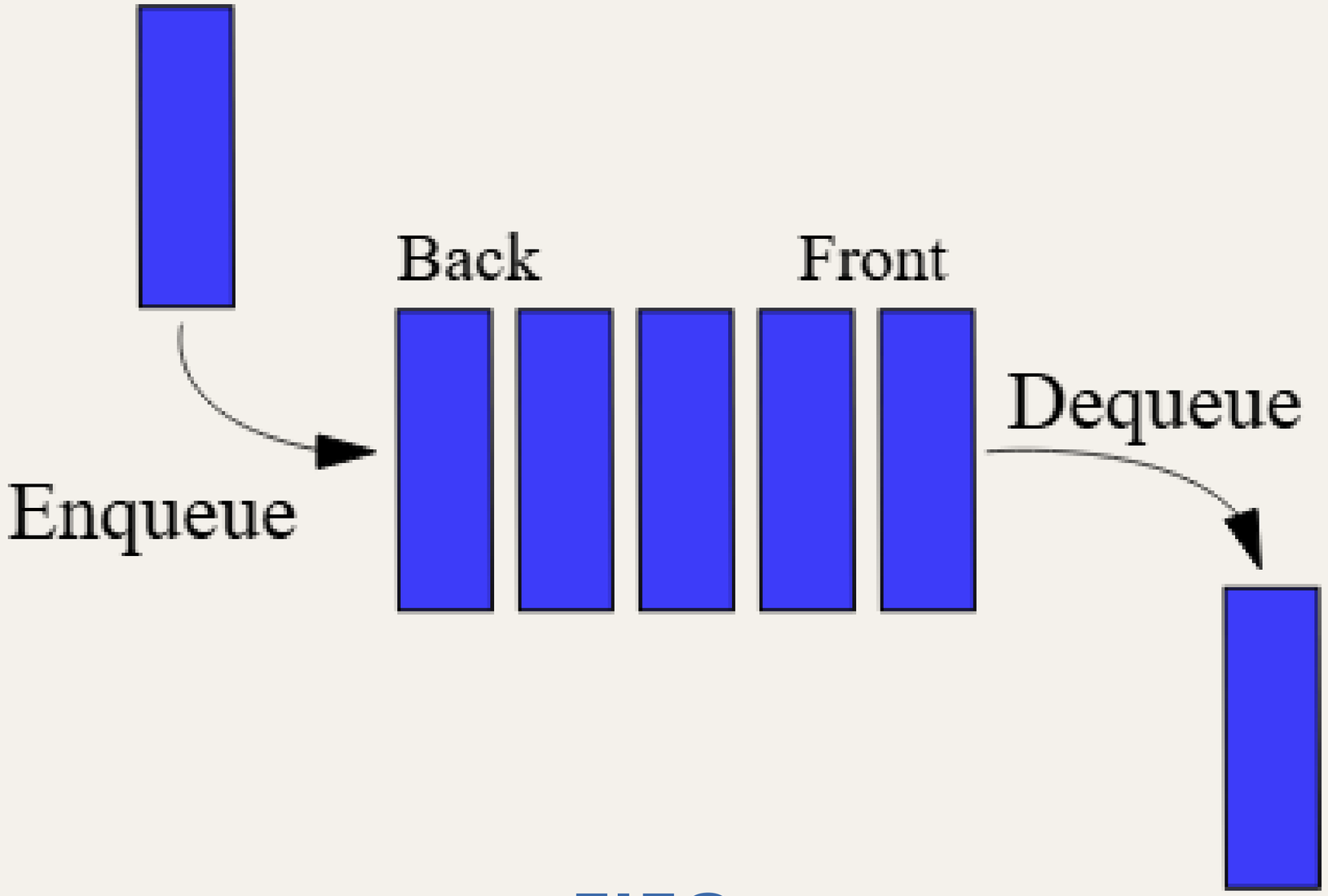
What is a Queue?

- Linear structure using FIFO (First-In, First-Out).

- Insert at rear, remove from front.



- Some variations (e.g., priority queues, deques) do not strictly follow FIFO.



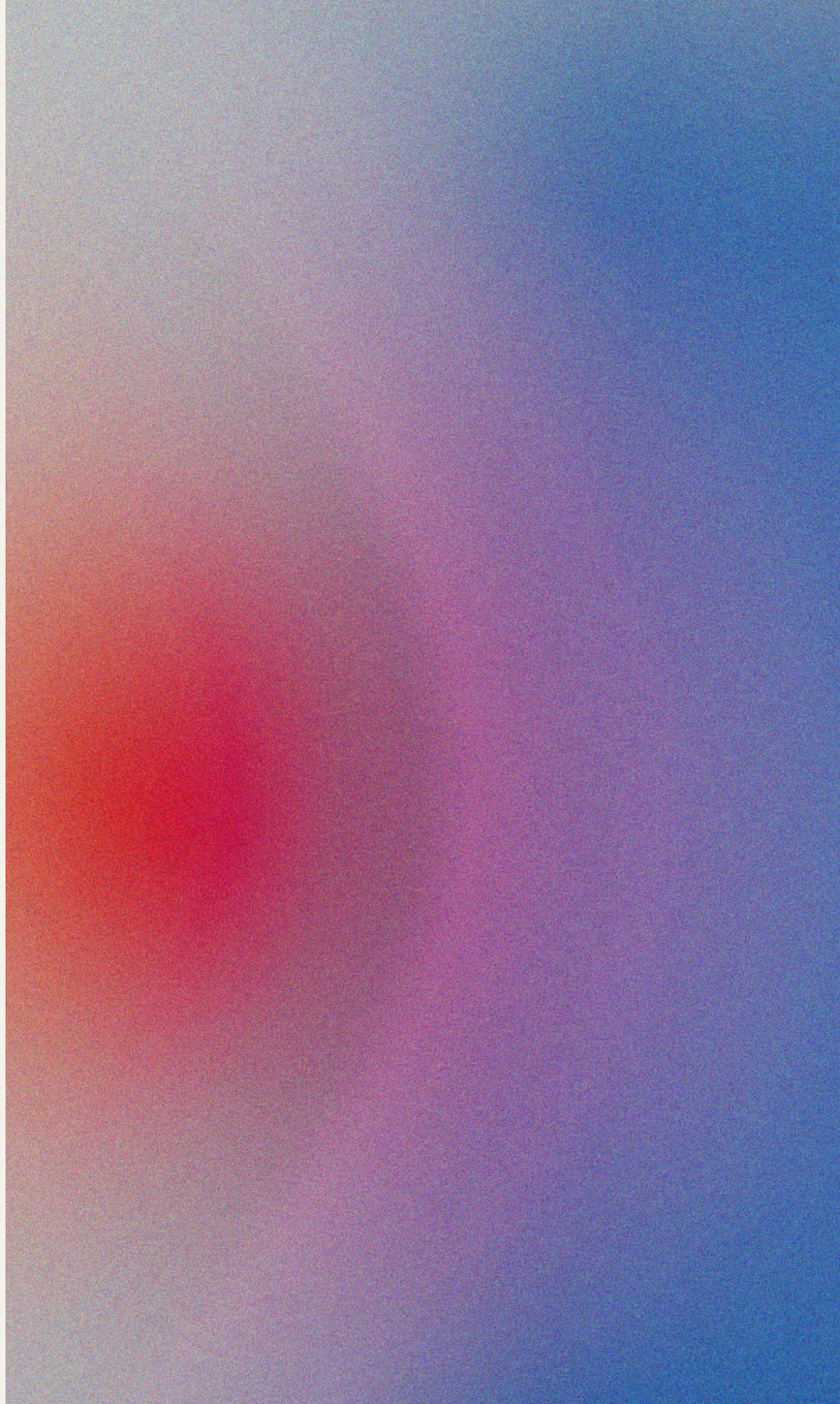
FIFO

Representation of Queues

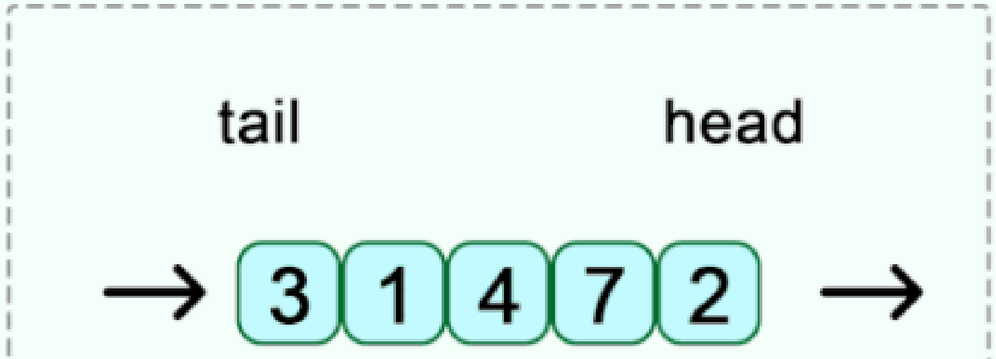
Queues can be implemented using :

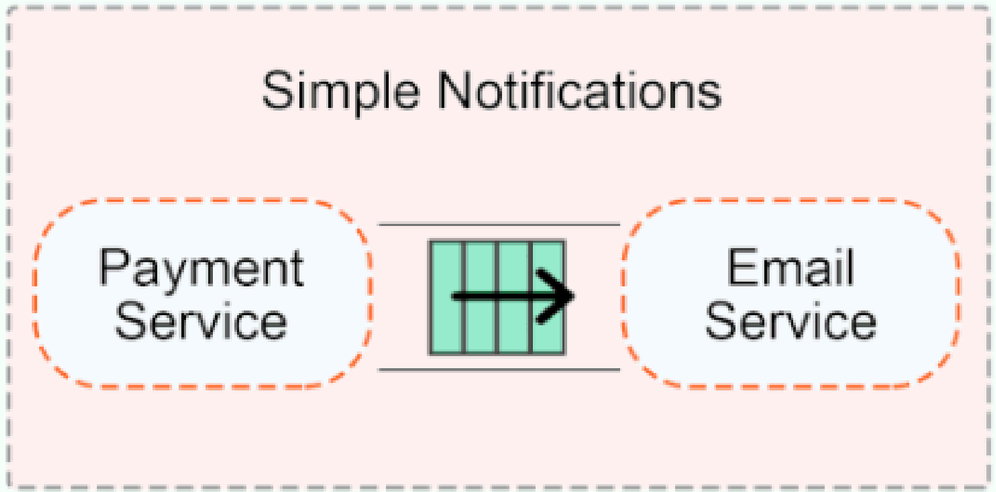
- 🧮 Arrays: Elements stored in a fixed block of memory, fast but limited in size.
- 🔗 Linked Lists: Each item links to the next; size can grow or shrink easily.
- 🏹 Pointers: Used to reference memory addresses.

VARIATIONS

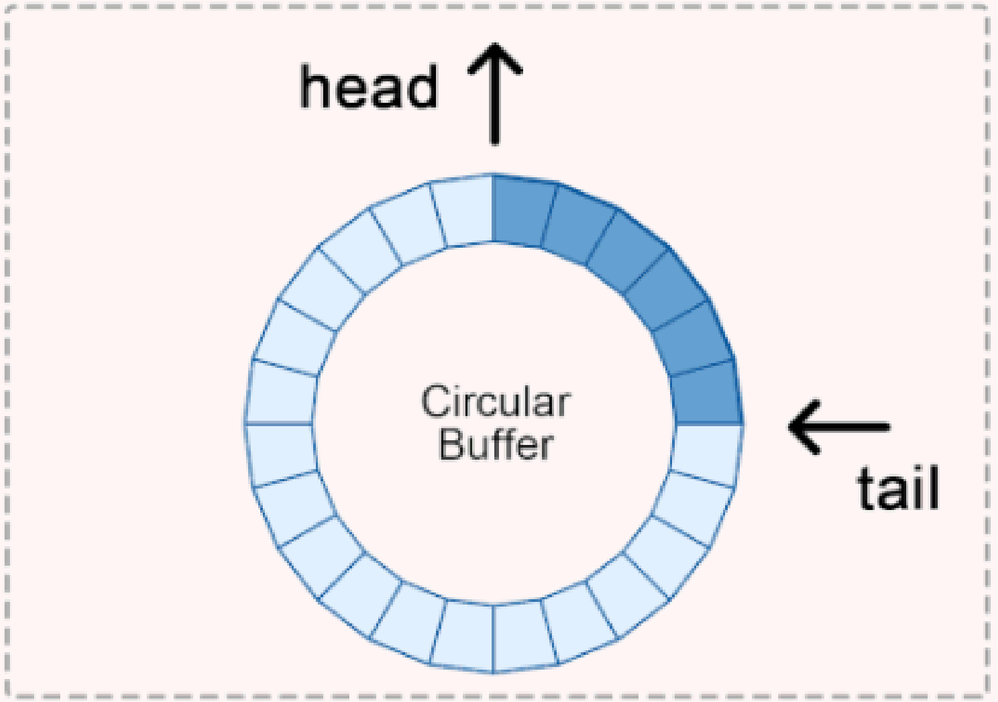


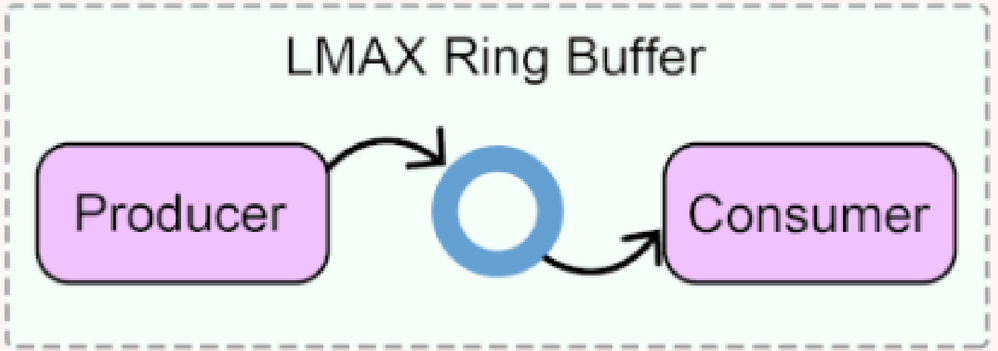
Simple FIFO Queue



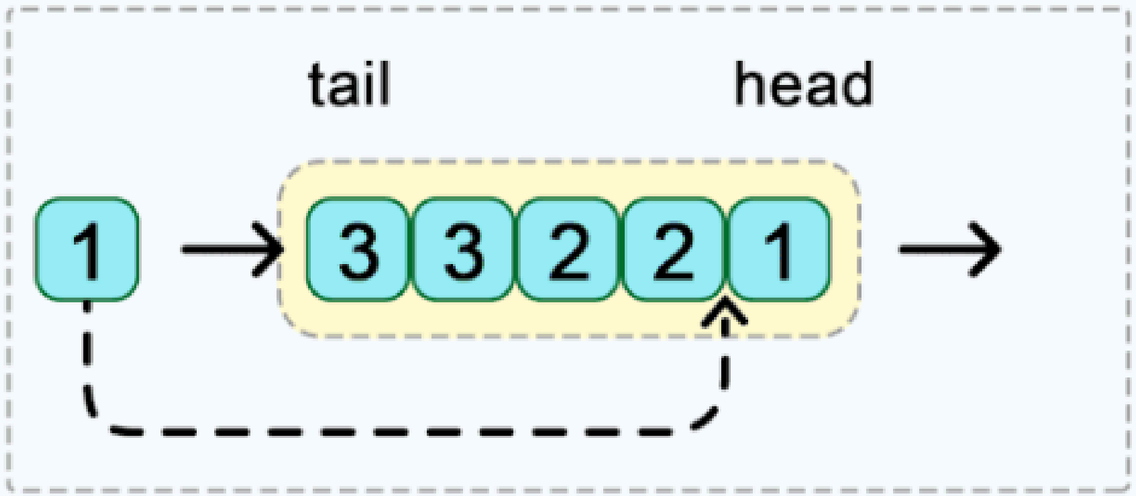


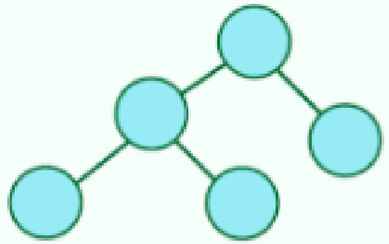
Circular Queue





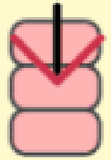
Priority Queue





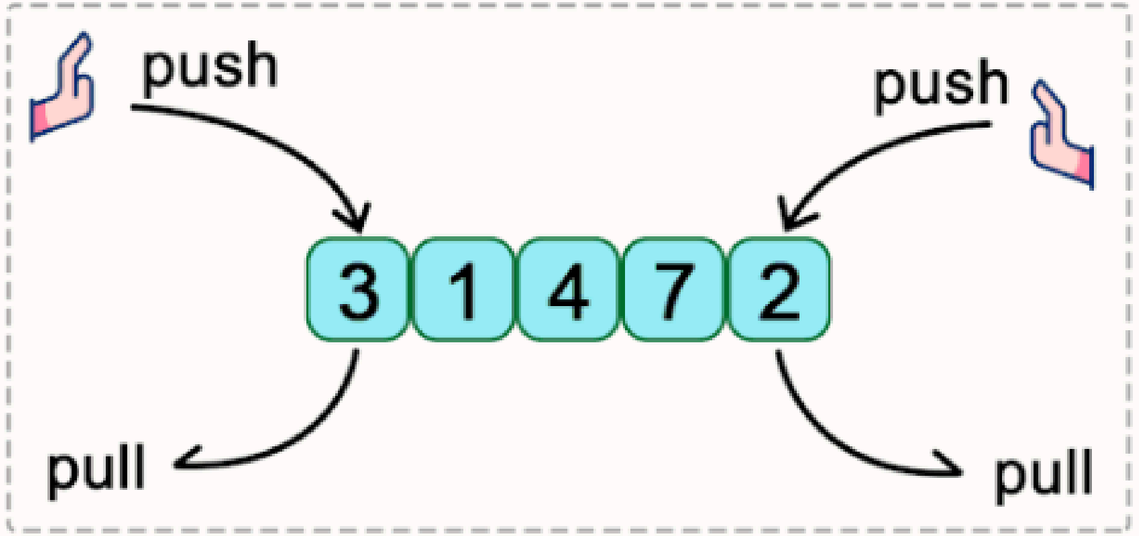
implemented using
a heap

Patient



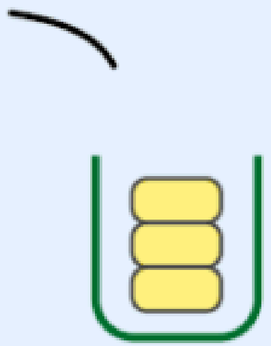
Emergency

Deque

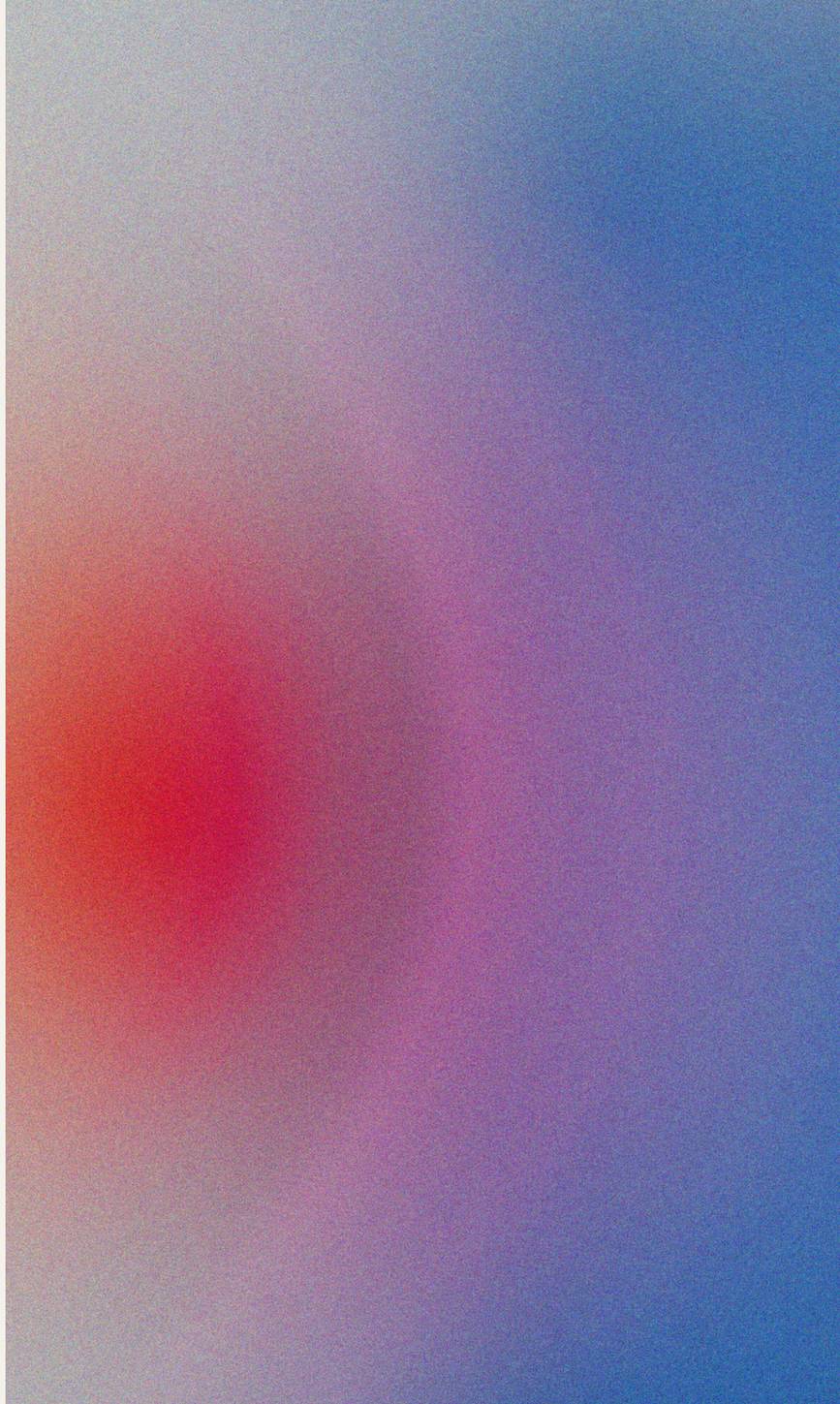


implemented using
a double-linked list

Stack



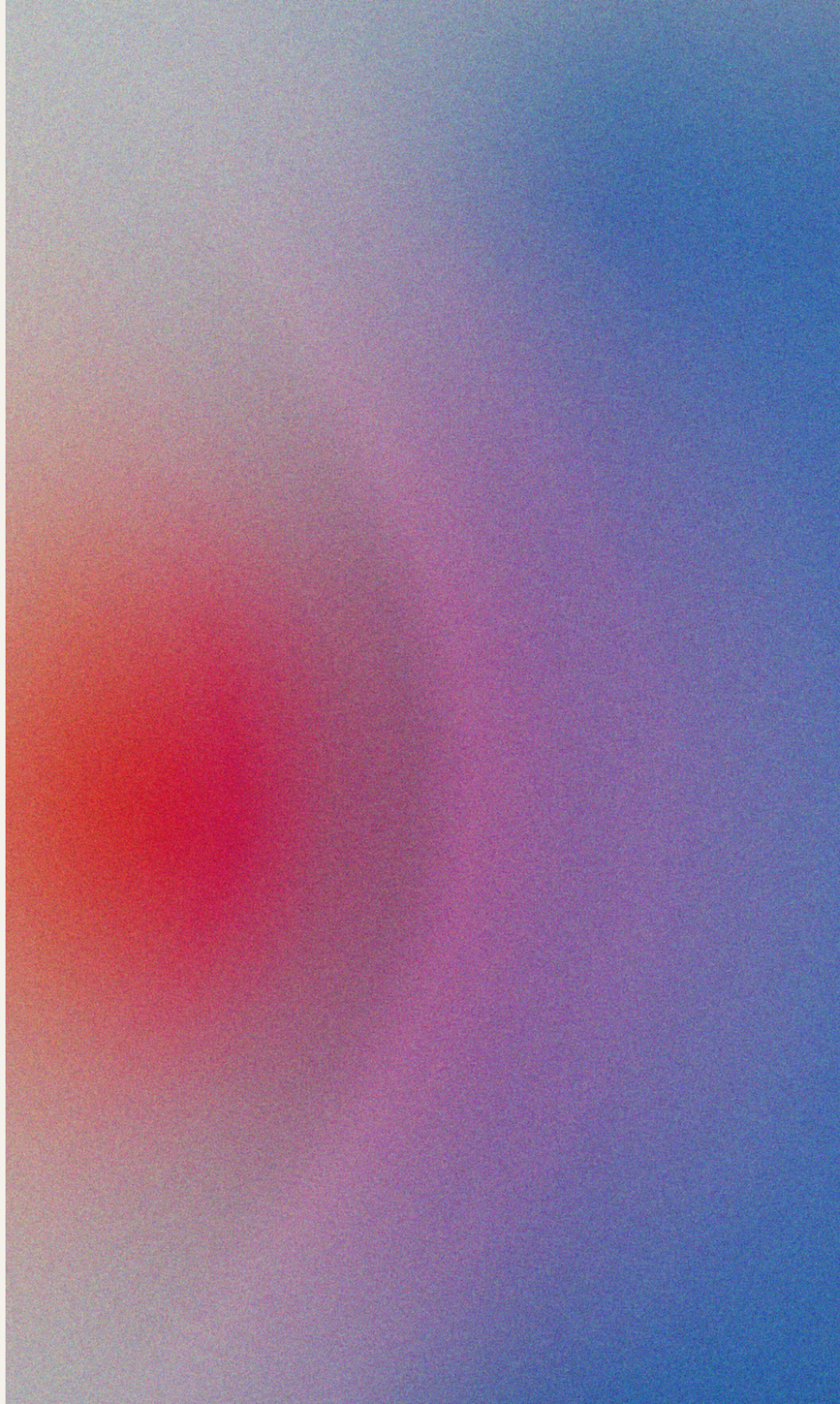
Interface



Queue ADT- Operations

- enqueue(item) : Add item at the rear.
- dequeue() : Remove and return item from the front.
- peek() or front() : View the front item without removing it.
- isEmpty() : Check if the queue is empty.
- size() : Return the number of elements in the queue.

Use Cases and Limitations



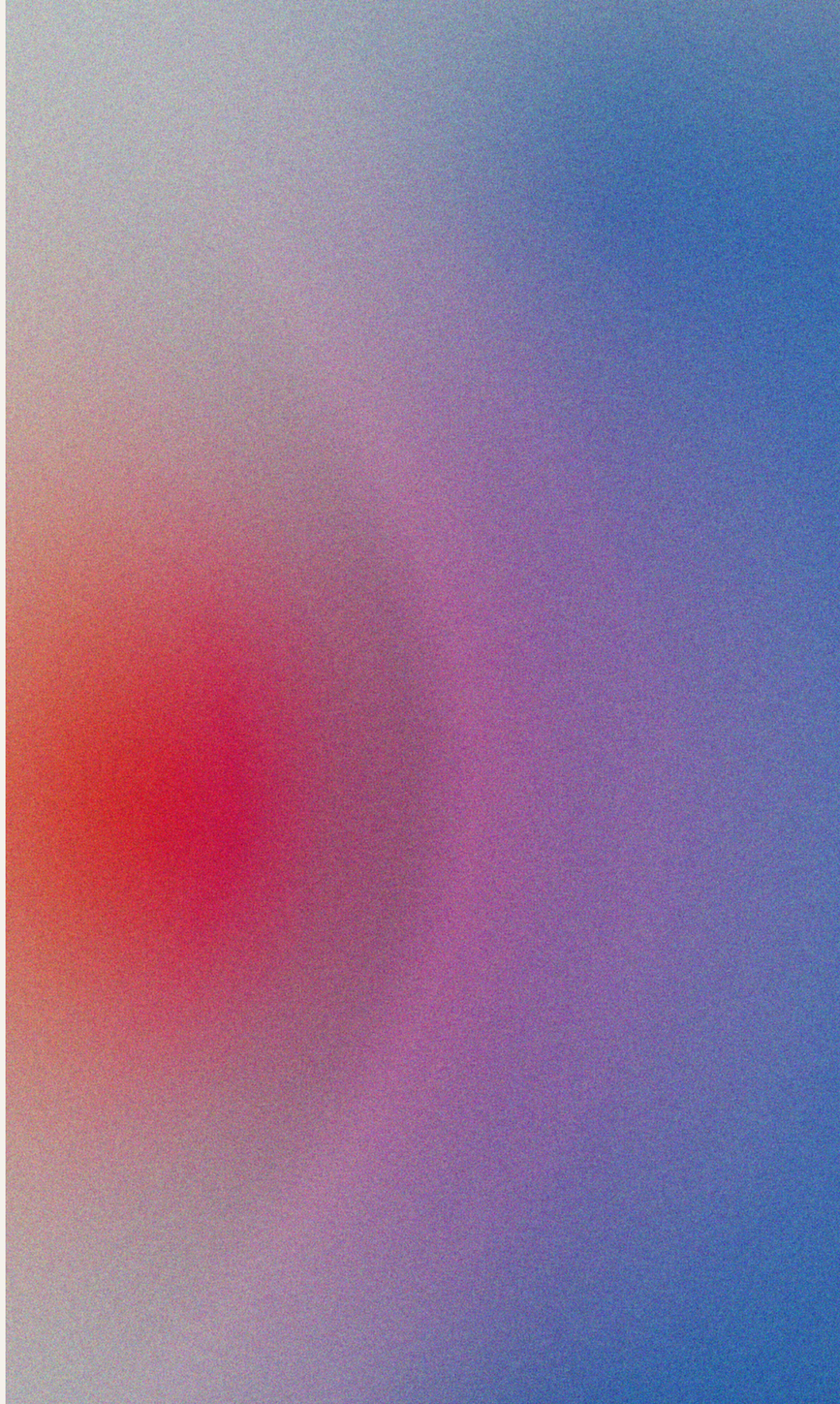
Queue ADT Applications

- ⚙️ Multithreading: Threads wait their turn to use the CPU.
- 📞 Customer Support Chat: Users are helped in the order they joined.
- 📚 Compiler Tasks: Compilation steps are processed one by one.
- 🧃 Vending Machine: Items are dispensed in the order selected.
- 🚗 Drive-Thru: Cars are served in arrival order.

Queue ADT Limitations

- No random access : You can only access the front or rear, not elements in the middle.
- Fixed size (in arrays) : Risk of overflow if the queue reaches its limit.
- Inefficient dequeue() in arrays : Removing the front may require shifting all elements.
- Not flexible for priority handling

Class



```python
class JiraTicketQueue:
    def __init__(self):
        self.tickets = []

    def enqueue(self, ticket):
        """Add a new Jira ticket to the queue."""
        self.tickets.append(ticket)
        print(f"Ticket '{ticket}' added to the queue.")

    def dequeue(self):
        """Process and remove the next ticket in the queue."""
        if self.isEmpty():
            print("No tickets to process.")
            return None
        ticket = self.tickets.pop(0)
        print(f"Ticket '{ticket}' has been processed.")
        return ticket

    def peek(self):
        """View the next ticket to be processed."""
        if self.isEmpty():
            return "No tickets in queue."
        return f"Next ticket: '{self.tickets[0]}'"

    def isEmpty(self):
        return len(self.tickets) == 0

    def size(self):
        return len(self.tickets)
```

```
PS C:\Users\user> & C:/Users/user/AppData/Local/Programs/Python/Python313
Ticket 'BUG-101: Login error' added to the queue.
Ticket 'TASK-204: Add search filter' added to the queue.
Ticket 'FEATURE-305: New dashboard design' added to the queue.
3
Next ticket: 'BUG-101: Login error'
Ticket 'BUG-101: Login error' has been processed.
2
Next ticket: 'TASK-204: Add search filter'
Ticket 'TASK-204: Add search filter' has been processed.
1
Next ticket: 'FEATURE-305: New dashboard design'
```

```python
# Example usage of the JiraTicketQueue class
queue = JiraTicketQueue()

queue.enqueue("BUG-101: Login error")    #adding a jira ticket to the queue
queue.enqueue("TASK-204: Add search filter")
queue.enqueue("FEATURE-305: New dashboard design")

print(queue.size()) #showing the queue size

print(queue.peek()) #showing the next tichet in the queue

queue.dequeue()   #process and remove the ticket in the front

print(queue.size())

print(queue.peek())

queue.dequeue()

print(queue.size())

print(queue.peek())


queue.dequeue()
print(queue.peek())

is_empty = queue.isEmpty()  # Check if the queue is empty
print(f"Is the queue empty? {is_empty}")  # Output: True
```

Thank you