5118006-03 Data Structures

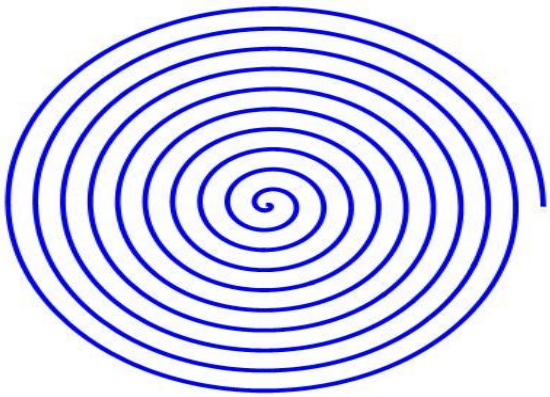
Queue

29 Mar 2024

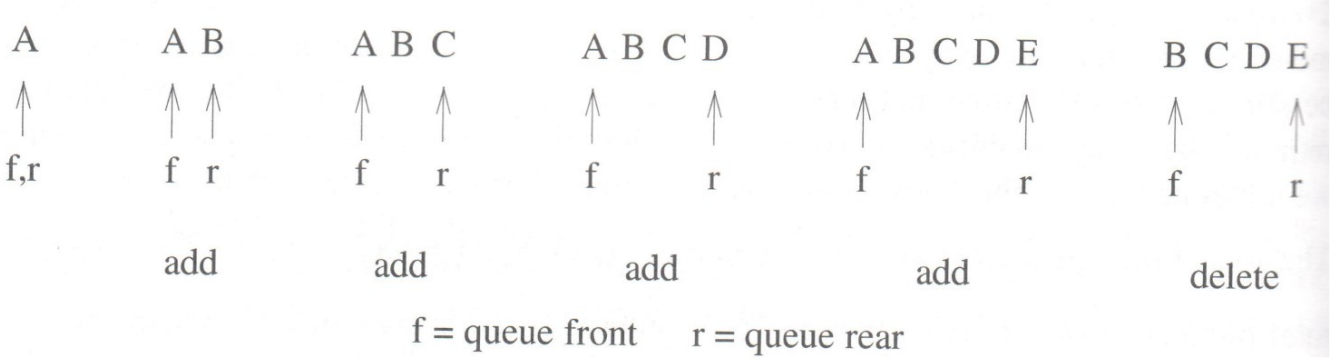
Shin Hong

Queue

- A queue is an ordered list where insertion and deletion take place at different ends
  - insertion at the rear, deletion at the front
- A queue is known as a First-In-First-Out (FIFO) list



Example


A
A B
A B C
A B C D
A B C D E
B C D E
f,r
f r
f r
f r
f r
f r
add
add
add
add
delete
f = queue front
r = queue rear

Queue ADT

- Structure
  - **elements**: an array to hold elements
  - **capacity**: the maximum number of elements that the queue can hold
    - a queue can be bounded or unbounded

- Operations
  - **add(e)**: insert a new element **e** to if the queue is not full (i.e., enqueue)
  - **delete()**: return the least recently inserted element if the queue is not empty (i.e., dequeue)
  - **isEmpty()** : return whether the queue has at least one element or not
  - **isFull()** : return whether the queue is full or not

Circular Queue

- Implement a queue as an array-based list
  - maintain the indices of the front and the rear
  - the indices rotates around the indices of the element array

- Two designs for identifying empty and full states

  1. keep at least one array element between the front and the rear
     - front indicates the next element to be removed if it is the same as rear (i.e., there exists at least one element)
     - rear indicates the empty slot for the next coming element

  2. store the number of elements that a queue currently holds
     (or whether the queue is full/empty or not)
     - front indicates the next element to be removed if there exists at least one element
     - rear indicates the empty slot for the next coming element if the queue is not full