

Data Structure

Queues

Shin Hong

6 Apr 2023



DS&A. Chapter 5.2. Stacks

Queue

- A queue is a container of elements that are inserted and removed in the First-In First-Out (FIFO) manner
 - elements enter the queue at the rear, and are removed from the front
- Queue operations
 - enqueue(e): insert element e at the rear
 - dequeue(): remove element e at the front
 - front(): return a reference to the front element in the queue
 - size(): return the number of elements contained in the queue
 - empty(): return true if and only if no element is contained in the queue

Example

Operation	Output	$front \leftarrow q \leftarrow rear$
enqueue(5)	–	(5)
enqueue(3)	–	(5,3)
front()	5	(5,3)
size()	2	(5,3)
dequeue()	–	(3)
enqueue(7)	–	(3,7)
dequeue()	–	(7)
front()	7	(7)
dequeue()	–	()
dequeue()	"error"	()
empty()	true	()

Circular Queue (1/2)

- Although an array list can work as a queue, it is an inefficient solution if it moves all elements forward for each `removeFirst()` (i.e., `dequeue()`)
- To avoid shifting elements, a circular queue maintains three array indices:
 - **front**: the index of the element that has been contained longest time if the queue is not empty
 - **rear**: the index of the last inserted elements if the queue is not full
 - the index where a newly given element will be stored in next enqueue
 - **num**: the number of the elements in the queue
- front and rear are incremented by one in each dequeue/enqueue, or set to be zero if these reach the end of the array

Circular Queue (2/2)

```
Algorithm size():  
    return n  
Algorithm empty():  
    return (n == 0)  
Algorithm front():  
    if empty() then  
        throw QueueEmpty exception  
    return Q[f]  
Algorithm dequeue():  
    if empty() then  
        throw QueueEmpty exception  
    f ← (f + 1) mod N  
    n = n - 1  
Algorithm enqueue(e):  
    if size() == N then  
        throw QueueFull exception  
    Q[r] ← e  
    r ← (r + 1) mod N  
    n = n + 1
```

