# 6.1 Queue abstract data type (ADT)

## Queue abstract data type

A ***queue*** is an ADT in which items are inserted at the end of the queue and removed from the front of the queue. The queue ***enqueue*** operation inserts an item at the end of the queue. The queue ***dequeue*** operation removes and returns the item at the front of the queue. Ex: After the operations "Enqueue 7", "Enqueue 14", and "Enqueue 9", "Dequeue" returns 7. A second "Dequeue" returns 14. A queue is referred to as a ***first-in first-out*** ADT. A queue can be implemented using a linked list or an array.



A queue ADT is similar to waiting in line at the grocery store. A person enters at the end of the line and exits at the front. British English actually uses the word "queue" in everyday vernacular where American English uses the word "line".

**PARTICIPATION ACTIVITY** 6.1.1: Queue ADT.

### Animation content:

`undefined`

### Animation captions:

1. A new queue named "wQueue" is created. Items are enqueued to the end of the queue.
2. Items are dequeued from the front of the queue.

**PARTICIPATION ACTIVITY** 6.1.2: Queue ADT.

1) Given numQueue: 5, 9, 1 (front is 5)
What are the queue contents after the following enqueue operation? Type the queue as: 1, 2, 3

Enqueue(numQueue, 4)



Check Show answer

2) Given numQueue: 11, 22 (the front is 11)

What are the queue contents after the following enqueue operations? Type the queue as: 1, 2, 3

Enqueue(numQueue, 28)
Enqueue(numQueue, 72)

Check Show answer



3) Given numQueue: 49, 3, 8
What is returned by the following dequeue operation?

Dequeue(numQueue)

Check Show answer

4) Given numQueue: 4, 8, 7, 1, 3
What is returned by the second dequeue operation?

Dequeue(numQueue)
Dequeue(numQueue)

Check Show answer

5) Given numQueue: 15, 91, 11
What is the queue after the following dequeue operation? Type the queue as: 1, 2, 3

Dequeue(numQueue)

Check Show answer



6) Given numQueue: 87, 21, 43
What are the queue's contents after the following operations? Type the queue

as: 1, 2, 3

Dequeue(numQueue)
Enqueue(numQueue, 6)
Enqueue(numQueue, 50)
Dequeue(numQueue)

Check Show answer



## Common queue ADT operations

Table 6.1.1: Some common operations for a queue ADT.

| Operation | Description | Example starting with queue: 43, 12, 77 (front is 43) |
|---|---|---|
| Enqueue(queue, x) | Inserts x at end of the queue | Enqueue(queue, 56). Queue: 43, 12, 77, 56 |
| Dequeue(queue) | Returns and removes item at front of queue | Dequeue(queue) returns: 43. Queue: 12, 77 |
| Peek(queue) | Returns but does not remove item at the front of the queue | Peek(queue) return 43. Queue: 43, 12, 77 |
| IsEmpty(queue) | Returns true if queue has no items | IsEmpty(queue) returns false. |
| GetLength(queue) | Returns the number of items in the queue | GetLength(queue) returns 3. |

Note: Dequeue and Peek operations should not be applied to an empty queue; the resulting behavior may be undefined.



**PARTICIPATION ACTIVITY** 6.1.3: Common queue ADT operations.

1) Given rosterQueue: 400, 313, 270, 514, 119, what does GetLength(rosterQueue) return?

400

- 5

2) Which operation determines if the queue contains no items?

- IsEmpty
- Peek



3) Given parkingQueue: 1, 8, 3, what are the queue contents after Peek(parkingQueue)?

- 1, 8, 3
- 8, 3

4) Given parkingQueue: 2, 9, 4, what are the contents of the queue after Dequeue(parkingQueue)?

- 9, 4
- 2, 9, 4

5) Given that parkingQueue has no items (i.e., is empty), what does GetLength(parkingQueue) return?

- -1
- 0
- Undefined

**CHALLENGE ACTIVITY** 6.1.1: Queue ADT.

Start

Given numQueue: 84, 27, 11
What are the queue's contents after the following operations?



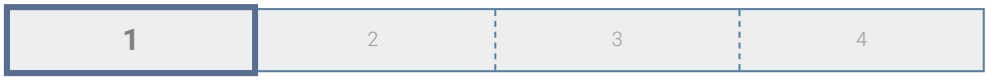
Enqueue(numQueue, 99)
Dequeue(numQueue)

Ex: 1, 2, 3

After the above operations, what does GetLength(numQueue) return?

Ex: 8






Check Next

# 6.2 Queues using linked lists

A queue is often implemented using a linked list, with the list's head node representing the queue's front, and the list's tail node representing the queue's end. Enqueueing an item is performed by creating a new list node, assigning the node's data with the item, and appending the node to the list. Dequeuing is performed by assigning a local variable with the head node's data, removing the head node from the list, and returning the local variable.

**PARTICIPATION ACTIVITY** 6.2.1: Queue implemented using a linked list.

### Animation content:

`undefined`

### Animation captions:

1. Enqueueing an item puts the item in a list node and appends the node to the list.
2. A dequeue stores the head node's data in a local variable, removes the list's head node, and returns the local variable.



**PARTICIPATION ACTIVITY** 6.2.2: Queue push and pop operations with a linked list.

Assume the queue is implemented using a linked list.

1) If the head pointer is null, the queue _____.

- is empty
- is full
- has at least one item

2) For the operation QueueDequeue(queue), what is the second parameter passed to ListRemoveAfter?

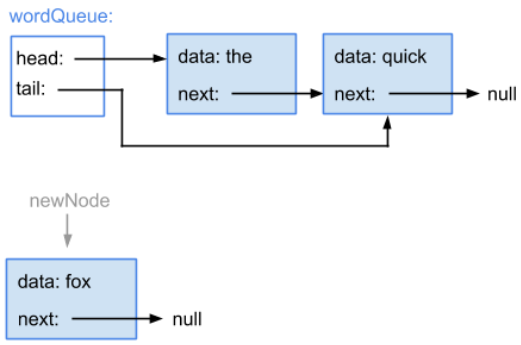
- The list's head node
- The list's tail node
- null



3) For the operation QueueDequeue(queue), headData is assigned with the list _____ node's data.

- head
- tail

4) For QueueEnqueue(wordQueue, "fox"), which pointer is updated to point to the node?




- wordQueue's head pointer
- The head node's next pointer
- The tail node's next pointer



**CHALLENGE ACTIVITY** 6.2.1: Queues using linked lists.

Start

Given an empty queue numQueue, what does the list head pointer point to? If the pointer is null, enter null.

Ex: 5 or null

What does the list tail pointer point to?



After the following operations:

QueueEnqueue(numQueue, 10)
QueueEnqueue(numQueue, 46)
QueueDequeue(numQueue)

What does the list head pointer point to?

What does the list tail pointer point to?

1 2

Check Next

# 6.3 Deque abstract data type (ADT)

## Deque abstract data type

A ***deque*** (pronounced "deck" and short for double-ended queue) is an ADT in which items can be inserted and removed at both the front and back. The deque push-front operation inserts an item at the front of the deque, and the push-back operation inserts at the back of the deque. The pop-front operation removes and returns the item at the front of the deque, and the pop-back operation removes and returns the item at the back of the deque. Ex: After the operations "push-back 7", "push-front 14", "push-front 9", and "push-back 5", "pop-back" returns 5. A subsequent "pop-front" returns 9. A deque can be implemented using a linked list or an array.



**PARTICIPATION ACTIVITY** 6.3.1: Deque ADT.

## Animation captions:

1. The "push-front 34" operation followed by "push-front 51" produces a deque with contents 51, 34.
2. The "push-back 19" operation pushes 19 to the back of the deque, yielding 51, 34, 19. "Pop-front" then removes and returns 51.
3. Items can also be removed from the back of the deque. The "pop-back" operation removes and returns 19.



**PARTICIPATION ACTIVITY** 6.3.2: Deque ADT.

Determine the deque contents after the following operations.

**push-front 71, push-front 68, push-front 97, pop-back, push-front 45**

**push-front 97, push-back 71, pop-front, push-front 45, push-back 68**

**push-back 45, push-back 71, push-front 97, push-front 68, pop-back**

| | |
|---|---|
| | 45, 97, 68 |
| | 45, 71, 68 |
| | 68, 97, 45 |

Reset

## Common deque ADT operations

In addition to pushing or popping at the front or back, a deque typically supports peeking at the front and back of the deck and determining the length. A ***peek*** operation returns an item in the deque without removing the item.



Table 6.3.1: Common deque ADT operations.

| Operation | Description | Example starting with deque: 59, 63, 19 (front is 59) |
|---|---|---|
| PushFront(deque, | Inserts x at the front of the deque | PushFront(deque, 41). Deque: |

| | | |
|---|---|---|
| x) | | 41, 59, 63, 19 |
| PushBack(deque, x) | Inserts x at the back of the deque | PushBack(deque, 41). Deque: 59, 63, 19, 41 |
| PopFront(deque) | Returns and removes item at front of deque | PopFront(deque) returns 59. Deque: 63, 19 |
| PopBack(deque) | Returns and removes item at back of deque | PopBack(deque) returns 19. Deque: 59, 63 |
| PeekFront(deque) | Returns but does not remove the item at the front of deque | PeekFront(deque) returns 59. Deque is still: 59, 63, 19 |
| PeekBack(deque) | Returns but does not remove the item at the back of deque | PeekBack(deque) returns 19. Deque is still: 59, 63, 19 |
| IsEmpty(deque) | Returns true if the deque is empty | IsEmpty(deque) returns false. |
| GetLength(deque) | Returns the number of items in the deque | GetLength(deque) returns 3. |



**PARTICIPATION ACTIVITY** 6.3.3: Common queue ADT operations.

1) Given rosterDeque: 351, 814, 216, 636, 484, 102, what does GetLength(rosterDeque) return?
   - 351
   - 102
   - 6

2) Which operation determines if the deque contains no items?
   - IsEmpty
   - PeekFront



3) Given jobsDeque: 4, 7, 5, what are the deque contents after PeekBack(jobsDeque)?
   - 4, 7, 5
   - 4, 7

4) Given jobsDeque: 3, 6, 1, 7, what are the contents of the deque after PopFront(jobsDeque)?

- 6, 1, 7
- 3, 6, 1, 7

5) Given that jobsDeque is empty, what does GetLength(jobsDeque) return?

- -1
- 0
- Undefined



**CHALLENGE ACTIVITY** 6.3.1: Deque ADT.

Start

Given an empty deque numDeque, what are the deque's contents after the following operatio

PushFront(numDeque, 73)
PushBack(numDeque, 93)
PushBack(numDeque, 19)
PushFront(numDeque, 47)

Ex: 1, 2, 3

After the above operations, what does PeekFront(numDeque) return?

Ex: 5

After the above operations, what does PeekBack(numDeque) return?

Ex: 5

After the above operations, what does GetLength(numDeque) return?

Ex: 5



1 2 3

Check Next

# 6.4 LAB: Grocery shopping list (LinkedList)

## 6.4 LAB: Grocery shopping list (LinkedList)

This section's content is not available for print.



## 6.5 LAB: Student grades (HashMap)

This section's content is not available for print.

## 6.6 LAB: Ticketing service (Queue)

This section's content is not available for print.