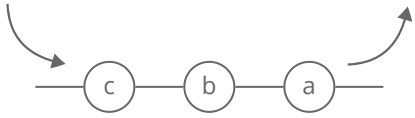


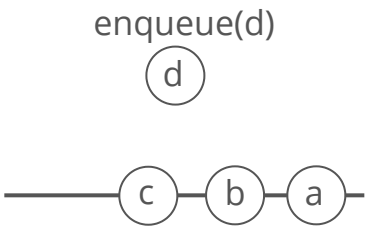
Queue

Data Structure (/data-structures-reference)

Quick reference

A **queue** stores items in a first-in, first-out (FIFO) order.

Picture a queue like the line outside a busy restaurant. First come, first served.



	Worst Case
space	$O(n)$
enqueue	$O(1)$
dequeue	$O(1)$
peek	$O(1)$

Strengths:

- **Fast operations**. All queue operations take $O(1)$ time.

Uses

- **Breadth-first search (/concept/bfs)** uses a queue to keep track of the nodes to visit next.

- **Printers** use queues to manage jobs—jobs get printed in the order they're submitted.
- **Web servers** use queues to manage requests—page requests get fulfilled in the order they're received.
- **Processes** wait in the CPU scheduler's queue for their turn to run.

Implementation

Queues are easy to implement with linked lists (/concept/linked-list):

- To enqueue, insert at the tail of the linked list.
- To dequeue, remove at the head of the linked list.

> You *could* implement a queue with an array (/concept/array) or dynamic array (/concept/dynamic-array), but it would get kinda messy. Try drawing it out. You'll notice that you'd need to build out a "scoot over" or "re-center" operation that automatically fires when your queue items hit the bottom edge of the array.

← course home (/table-of-contents)

Next up: Stack → (/concept/stack?course=fc1&section=queues-stacks)