CS112
Data Structures

- Stack
- Queue

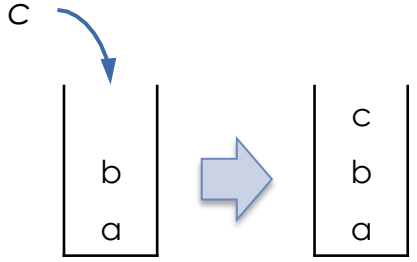
Stack

A stack is a collection with LIFO (Last In First Out) behavior

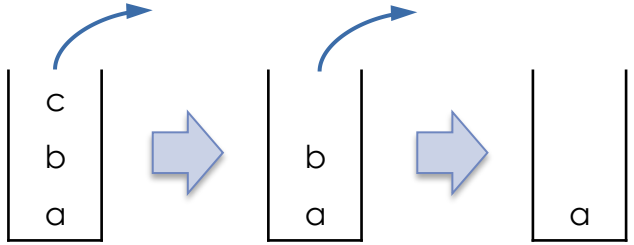
- It memorizes things and recalls in the reverse order (undo operation of the text editor)

Operations

- Push and pop are allusions to physical stacks. The order in which items are popped from the stack is the reverse of the order in which they were pushed onto the stack
- **Push**: add an entry to the top of the stack



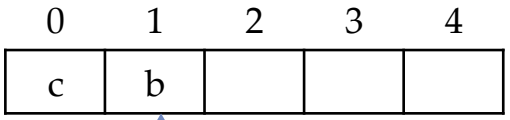
- **Pop**: removes an entry from the top of the stack



Stack Implementation

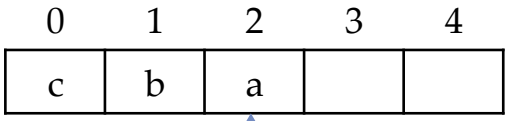
There are many ways to implement a stack, here are two:

Array



stack top

push a



stack top

Have to keep the index of the top of the stack

Linked List

push a

Use addToFront to push into the stack and RemoveFront to pop the stack

What is the running time to find how many items in the stack?

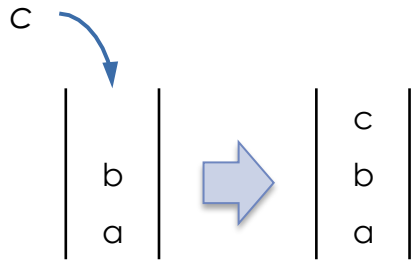
Queue

A queue is a collection with FIFO (First In First Out) behavior.

- It memorizes things and recalls them in the order they were inserted. It has a front and an end (tail).

Operations

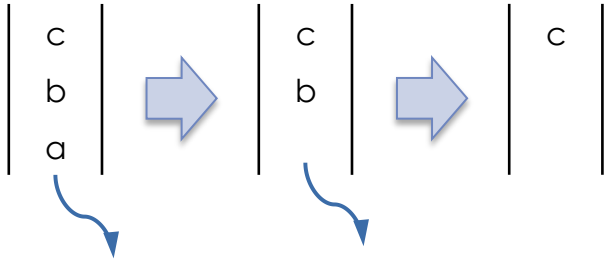
- **Enqueue**: add an entry to the end of the queue



We are going to see three ways to implement a queue:

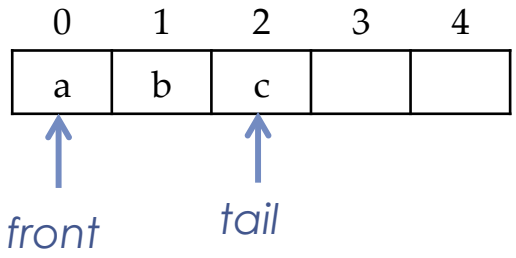
1. Array
2. Circular bounded array
3. Circular linked list

- **Dequeue**: remove an entry from the front of the queue

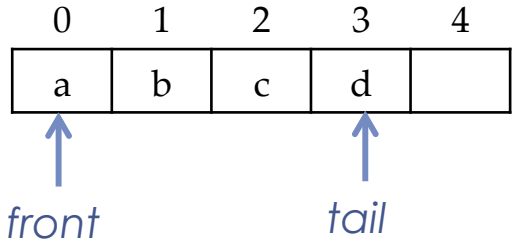


Queue Implementation: Array

Enqueue

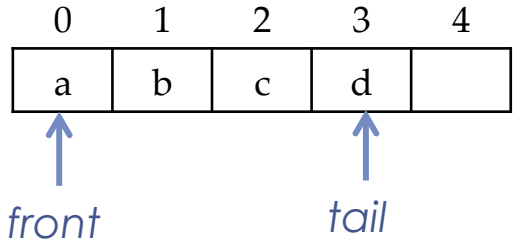


enqueue d

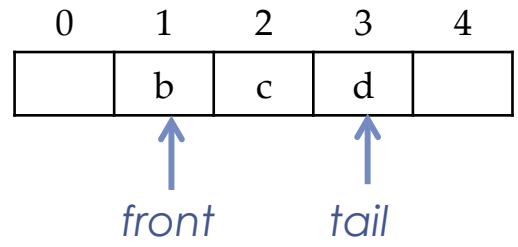


How many items in the array?
tail – front + 1

Dequeue



dequeue

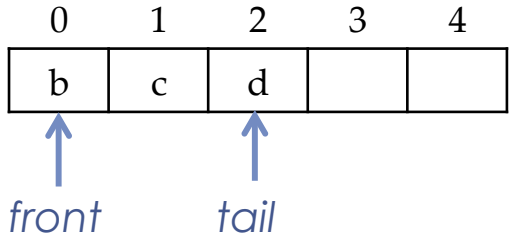


How to fill the empty spot?

Queue Implementation: Array

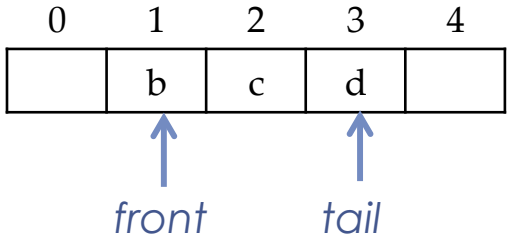
How to fill the empty spot after a dequeue?

a. Move all the entries over by 1 positions



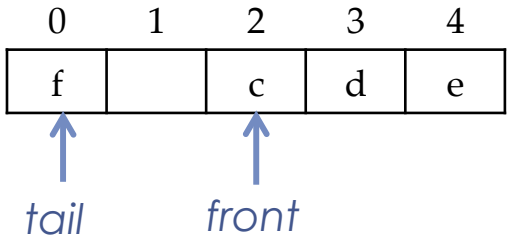
Each dequeue would take linear time: very inefficient

b. Leave the space empty



Each dequeue would take constant time but space is wasted

c. Use a circular bounded array (wraps around)



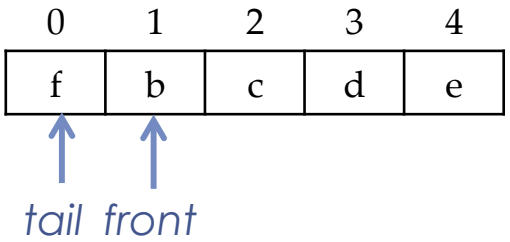
Leaves empty spaces but reclaims it

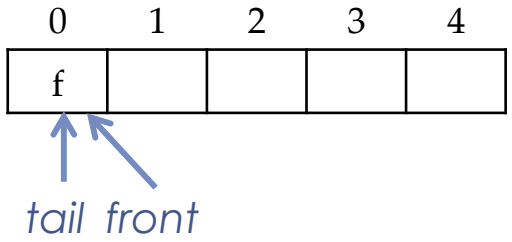
Queue Implementation: Array

Circular bounded array

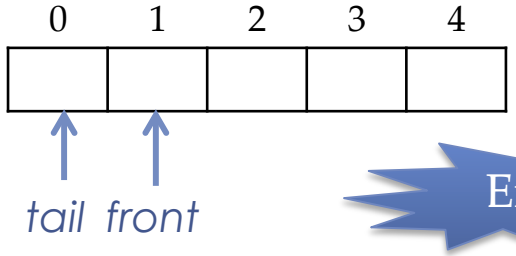
- How many items?
  - Before: tail – front + 1
  - Now we can have tail < font index
    - tail < front: tail – front + 1
    - tail > front: size – (front – tail – 1)
- Is the queue empty or full?

The only way to know if it is empty or full is to keep the size of the queue.





dequeue



Queue Implementation: CLL

Linked lists are more attractive when implementing queues

- Enqueue
  - add an element to the tail of the CLL
- Dequeue
  - remove the front item of the CLL