Queues

Queue

- a queue represents a sequence of elements where elements can be added at the back of the sequence and removed from the front of the sequence

Queue

Queue Operations

- classically, queues only support two operations
  1. enqueue
     - add to the back of the queue
  2. dequeue
     - remove from the front of the queue

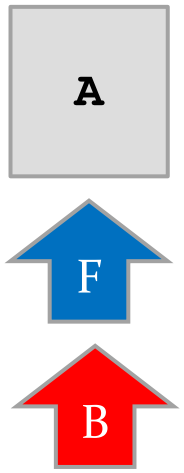
Queue Optional Operations

- optional operations
  1. size
     - number of elements in the queue
  2. isEmpty
     - is the queue empty?
  3. peek
     - get the front element (without removing it)
  4. search
     - find the position of the element in the queue
  5. isFull
     - is the queue full? (for queues with finite capacity)
  6. capacity
     - total number of elements the queue can hold (for queues with finite capacity)

Enqueue

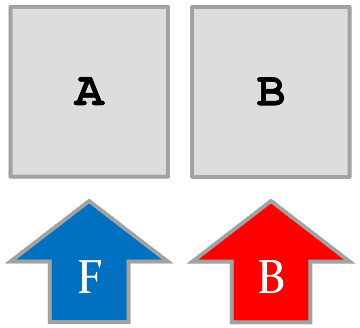
1. `q.enqueue("A")`

The enqueue operation adds elements to the back of the queue. If you enqueue an element into an empty queue then the front (F) and back (B) of the queue both refer to the same element.



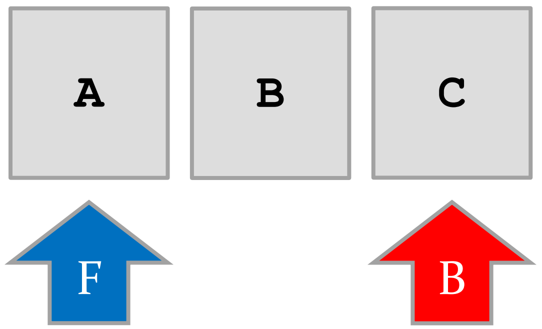
Enqueue

1. **`q.enqueue("A")`**
2. **`q.enqueue("B")`**



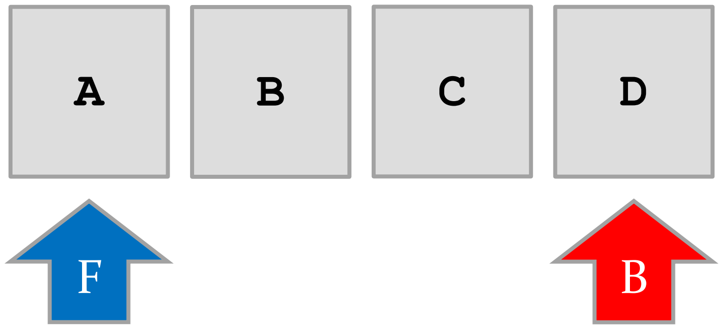
Enqueue

1. `q.enqueue("A")`
2. `q.enqueue("B")`
3. `q.enqueue("C")`



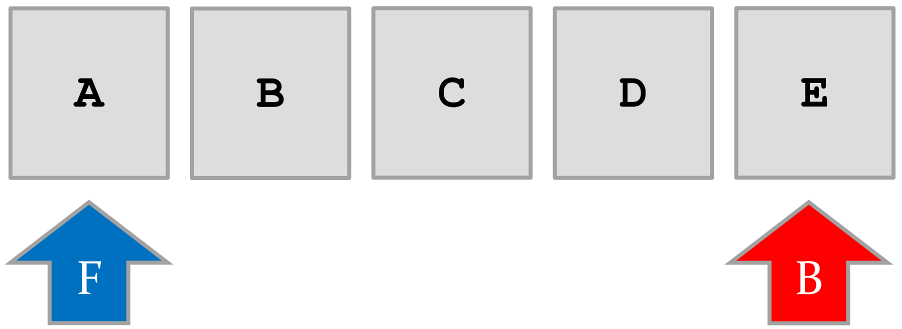
Enqueue

1. `q.enqueue("A")`
2. `q.enqueue("B")`
3. `q.enqueue("C")`
4. `q.enqueue("D")`



Enqueue

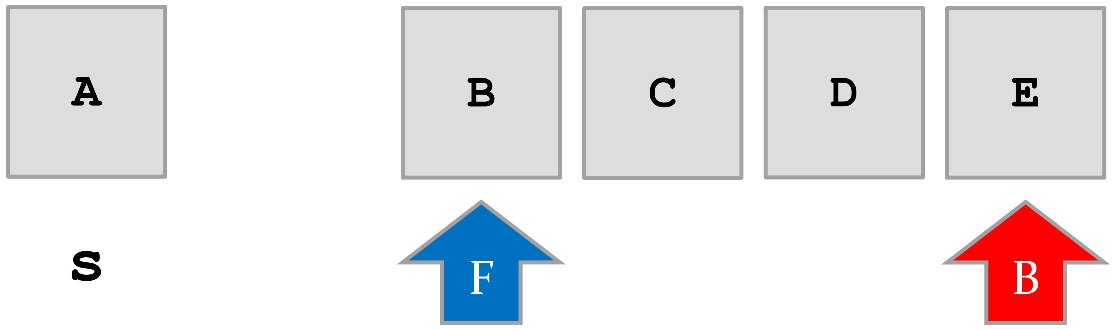
1. `q.enqueue("A")`
2. `q.enqueue("B")`
3. `q.enqueue("C")`
4. `q.enqueue("D")`
5. `q.enqueue("E")`



Dequeue

The dequeue operation removes an element from the front of the queue, and returns the element to the client.

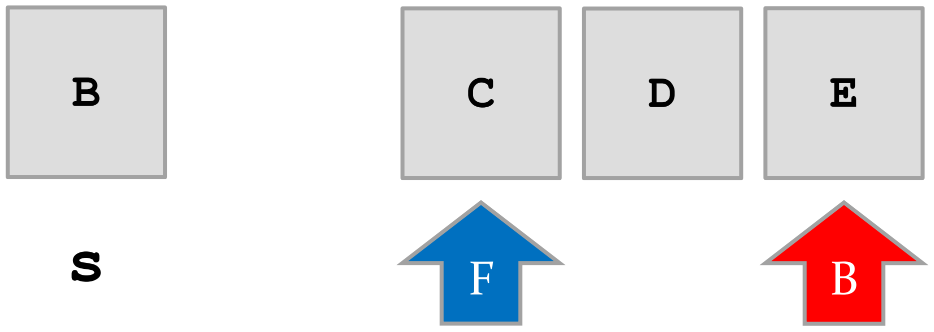
1. `String s = q.dequeue()` removes and returns "A"



Dequeue

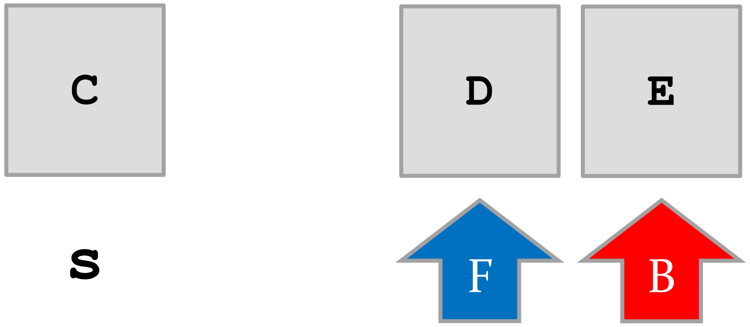
```
1.  String s = q.dequeue()
2.  s = q.dequeue()
```

removes and returns "B"



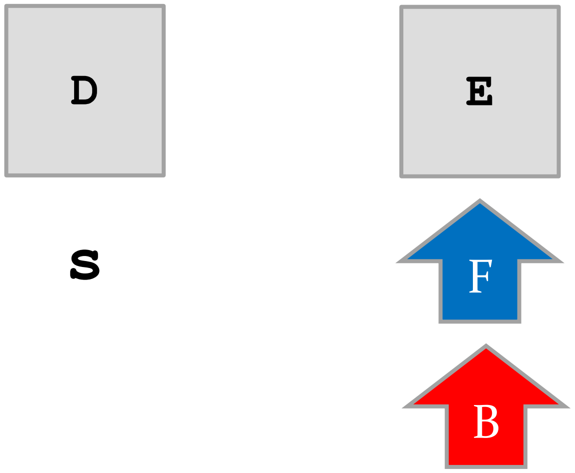
Dequeue

1. `String s = q.dequeue()`
2. `s = q.dequeue()`
3. `s = q.dequeue()` removes and returns "C"



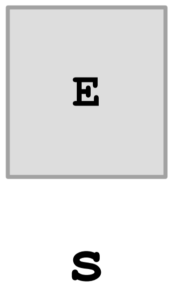
Dequeue

1. `String s = q.dequeue()`
2. `s = q.dequeue()`
3. `s = q.dequeue()`
4. `s = q.dequeue()` removes and returns "D"



Dequeue

1. **`String s = q.dequeue()`**
2. **`s = q.dequeue()`**
3. **`s = q.dequeue()`**
4. **`s = q.dequeue()`**
5. **`s = q.dequeue()`** removes and returns "E"



FIFO

- queue is a First-In-First-Out (FIFO) data structure
  - the first element enqueued in the queue is the first element that can be accessed from the queue

Queue applications

- queues are useful whenever you need to hold elements in their order of arrival
  - serving requests of a single resource
    - cashier
    - printer queue
    - disk queue
    - CPU queue
    - web server