

Queues

What is a queue?

- First-in first-out data structure (FIFO)
- New objects are placed at rear
- Removal restricted to front
- Examples?

Queue ADT Operations

- **enqueue(o)**: Insert o at rear of queue
 - Input: Object; Output: None
- **dequeue()**: Remove object at front; error if empty
 - Input: None; Output: Object removed
- **size()**: Return number of objects in queue
 - Input: None; Output: Integer
- **isEmpty()**: Return a boolean indicating queue empty
 - Input: None; Output: Boolean
- **first()**: Return object at front without removing; error if empty
 - Input: None; Output: Object

Example

- enqueue(5)
 - enqueue(3)
 - dequeue()
 - enqueue(7)
 - dequeue()
 - front()
 - dequeue()
 - dequeue()
 - isEmpty()
- enqueue(9)
 - enqueue(7)
 - size()
 - enqueue(3)
 - enqueue(5)
 - dequeue()

Queue Interface

`int size();`

`bool isEmpty();`

`Object front() throws QueueEmptyException;`

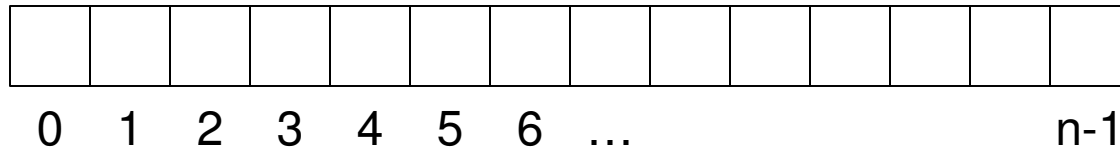
`void enqueue(Object obj);`

`Object dequeue() throws QueueEmptyException;`

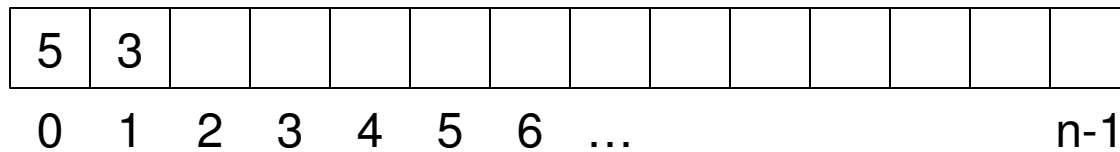
Underlying Representation

- Array versus Linked List
 - Pros and cons?
- Running time?
 - size
 - isEmpty
 - enqueue
 - dequeue
 - front

Array Implementation

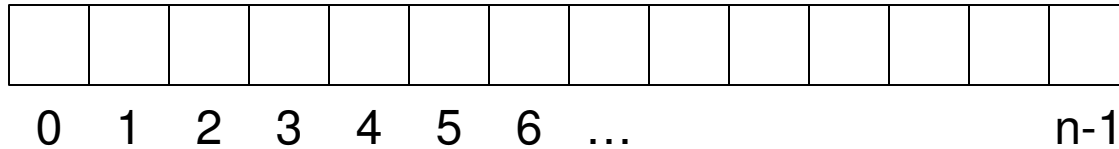


enqueue(5)
enqueue(3)

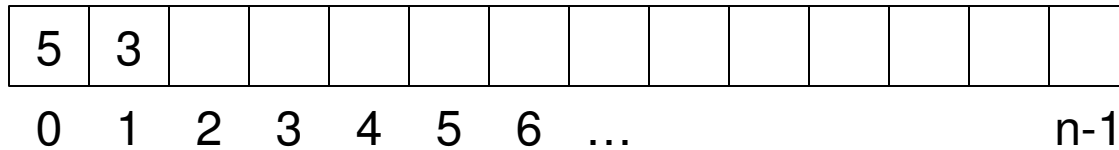


dequeue() ?

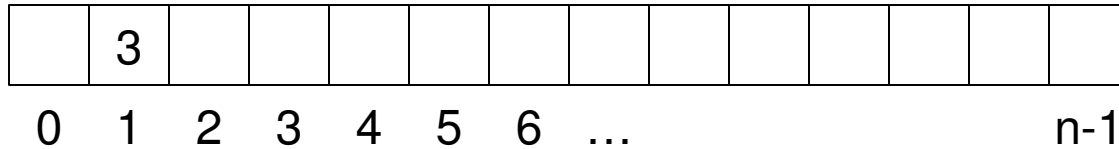
Array Implementation



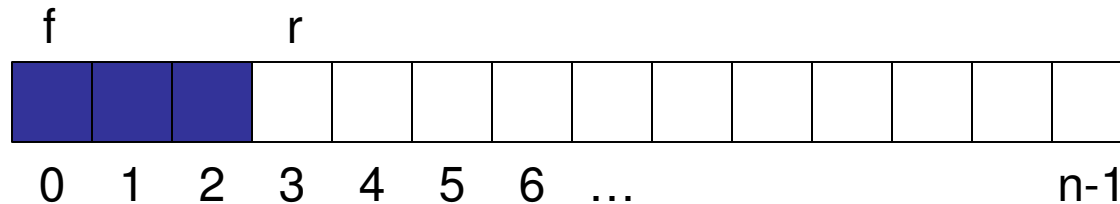
enqueue(5)
enqueue(3)



dequeue() ?

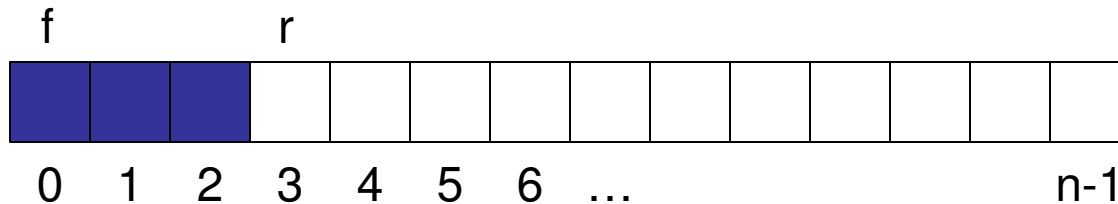


Circular Array



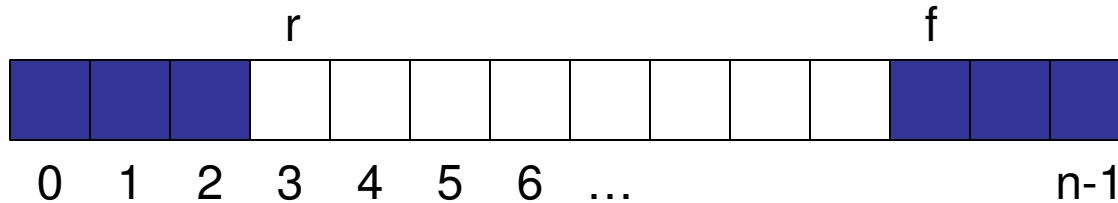
- f – stores index of cell which stores first element of queue
- r – stores index of next available cell in queue
- Initially, $f = r = 0$
- How do you add a new element?
- How do you remove an element?

Circular Array



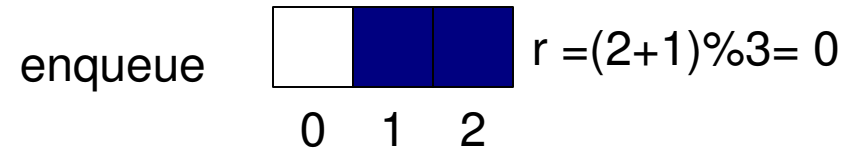
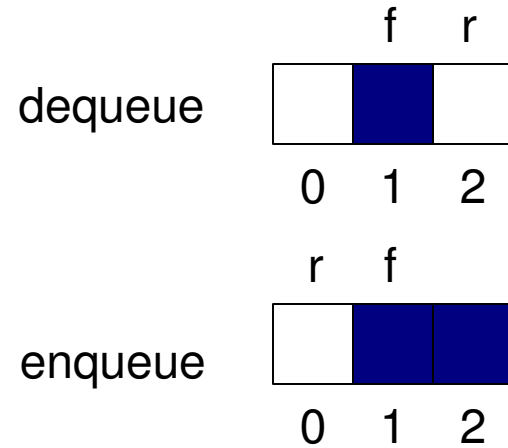
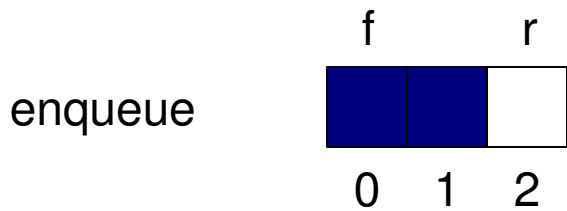
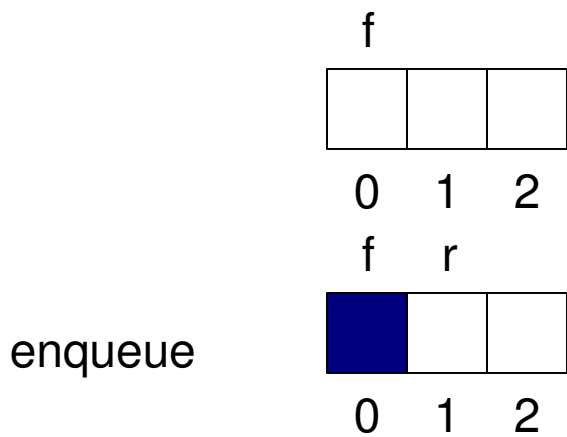
- How do you add a new element?
 - insert at `array[r]`
 - increment `r`
- How do you remove an element?
 - return `array[f]`
 - increment `f`
- What happens when $r \geq n-1$?

Circular Array



- Need to be able to wrap around
- Modulo – %
 - increment f using $(f+1)\%n$
 - increment r using $(r+1)\%n$

Circular Array



Algorithms

- size
 - return $(N-f+r) \bmod N$
- isEmpty
 - return $(f == r)$
- front
 - if isEmpty then throw QueueEmptyException
 - return array[f]
- dequeue
 - if isEmpty then throw QueueEmptyException
 - temp = array[f]
 - $f = (f+1)\%N$
 - return temp
- enqueue
 - if size == N-1 then throw QueueFullException **SIZE MUST BE < N-1**
 - array[r] = object
 - $r = (r+1)\%N$

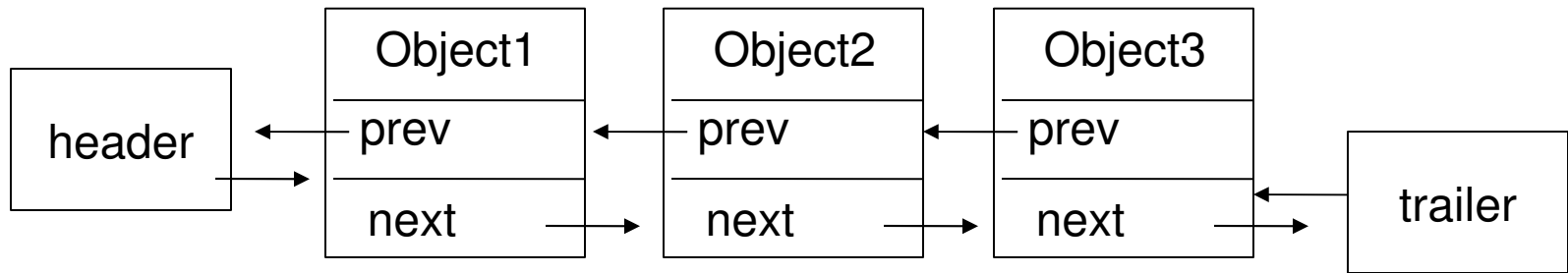
Deque

- Double-ended queue
 - insertFirst
 - insertLast
 - removeFirst
 - removeLast
 - first
 - last
 - size
 - isEmpty

Example

- insertFirst(3)
- insertFirst(5)
- first()
- removeFirst()
- insertLast(7)
- last()
- removeFirst()
- removeLast()

Doubly Linked List



- Algorithms
 - insertFirst
 - removeLast

Exercises

- Implement a queue using two stacks.
- Implement a stack using two queues.