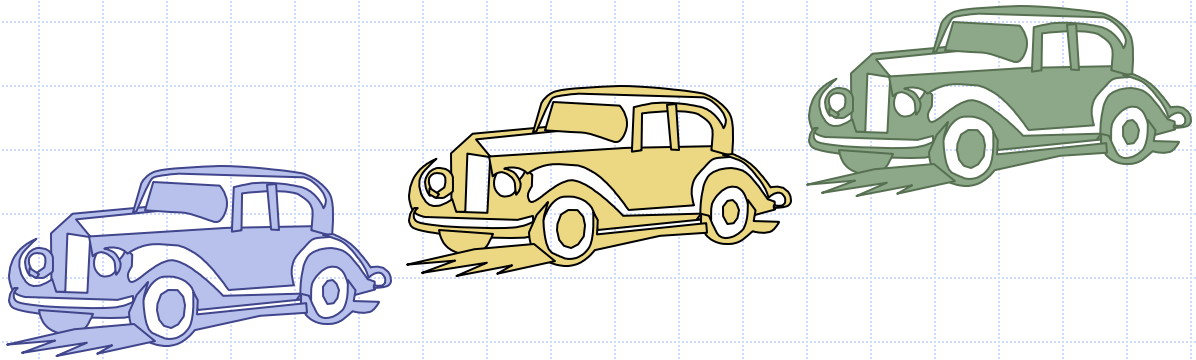
Presentation for use with the textbook Data Structures and Algorithms in Java, 6th edition, by M. T. Goodrich, R. Tamassia, and M. H. Goldwasser, Wiley, 2014

Queues



Reading

M. T. Goodrich, R. Tamassia and M. H. Goldwasser, *Data Structures and Algorithms in Java*, 6th Edition, 2014.

- **Chapter 6. Stacks and Queues**

The Queue ADT

插入与删除方式：
队列遵循先进先出（FIFO）原则，即第一个插入的元素会最先被移除。
插入操作发生在队列的尾部（rear），即新元素被加入队列的最后一个位置。
删除操作发生在队列的前部（front），即元素从队列的最前面被移除。

- The Queue ADT stores arbitrary objects
- Insertions and deletions follow the *first-in first-out* scheme
- Insertions are at the rear of the queue and removals are at the front of the queue
- Main queue operations:
  - enqueue(object): inserts an element at the end of the queue 插入操作，将一个元素插入到队列的尾部（rear）。
  - object dequeue(): removes and returns the element at the front of the queue

    删除操作，从队列的前部（front）移除并返回一个元素。
- Auxiliary queue operations: 返回队列最前面（front）元素，但不移除它。
  - object first(): returns the element at the front without removing it
  - integer size(): returns the number of elements stored 返回队列中当前存储的元素个数。
  - boolean isEmpty(): indicates whether no elements are stored 判断队列是否为空。如果队列为空，则返回 true；如果队列不为空，则返回 false。
- Boundary cases:
  - Attempting the execution of dequeue or first on an empty queue returns null

    队列为空时的情况：如果队列为空，执行 dequeue() 或 first() 操作时，都会返回 null，表示队列中没有元素可以移除或查看。

Example

Operation	*Output*	*content of the queue Q*
enqueue(5)		
enqueue(3)		
dequeue()		
enqueue(7)		
dequeue()		
first()		
dequeue()		
dequeue()		
isEmpty()		
enqueue(9)		
enqueue(7)		
size()		
enqueue(3)		
enqueue(5)		
dequeue()		

Example

Operation	*Output*	*content of the queue Q*
enqueue(5)	–	(5)
enqueue(3)	–	(5, 3)
dequeue()	*5*	(3)
enqueue(7)	–	(3, 7)
dequeue()	*3*	(7)
first()	*7*	(7)
dequeue()	*7*	()
dequeue()	*null*	()
isEmpty()	*true*	()
enqueue(9)	–	(9)
enqueue(7)	–	(9, 7)
size()	*2*	(9, 7)
enqueue(3)	–	(9, 7, 3)
enqueue(5)	–	(9, 7, 3, 5)
dequeue()	*9*	(7, 3, 5)

enqueue: 插入到尾部
dequeue: 删除队列前面的元素
first: 返回队列前面的元素
isEmpty: 是否为空，空为true，非空为false
size: 元素的个数
队列在执行 dequeue(), first() 时，如果队列为空，会返回 null，表示无法执行该操作。

Applications of Queues

- Direct applications
  - Waiting lists 等待列表
  - Access to shared resources (e.g., printer) 共享资源的访问（例如打印机）
  - Multiprogramming 多道程序设计
- Indirect applications
  - Auxiliary data structure for algorithms 算法的辅助数据结构
  - Component of other data structures 其他数据结构的组成部分

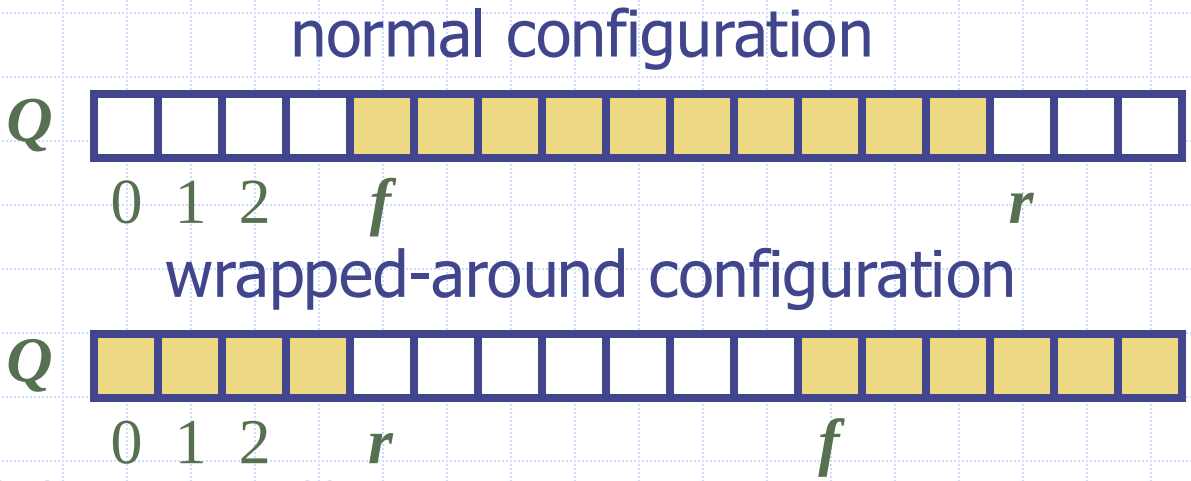
Array-based Queue

- Use an array of size N in a circular fashion
- Two variables keep track of the front and size
  - f index of the front element
  - sz number of stored elements

f：记录队列中前一个元素的索引（即下一个将被出队的元素）。
sz：记录队列中存储的元素数量，帮助确定当前队列的大小。

- *When the queue has fewer than N elements*, array location $r = (f + sz) \bmod N$ is the first empty slot past the rear of the queue

r = (f + sz) mod N：此公式用于计算在队列尾部之后的第一个空位置。当队列满时，它会“回绕”到数组的前端。

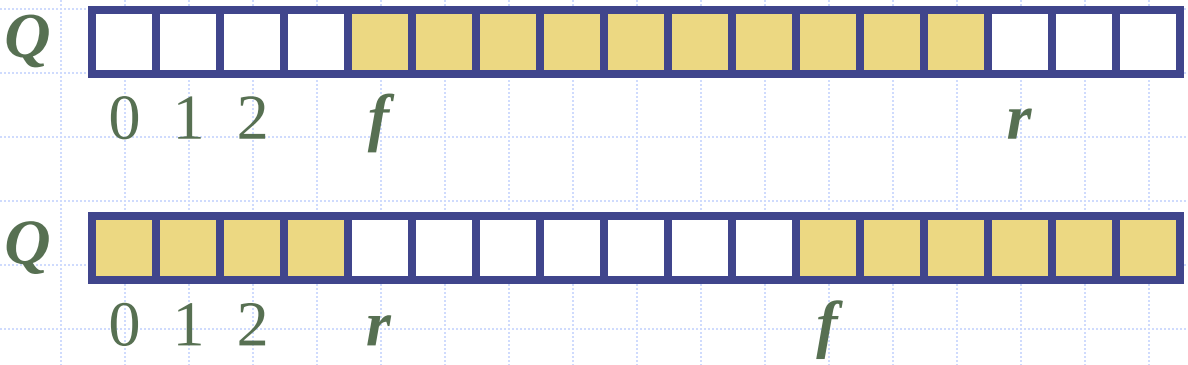


Queue Operations

- We use the modulo operator (remainder of division)

Algorithm *size*()
  return *sz*

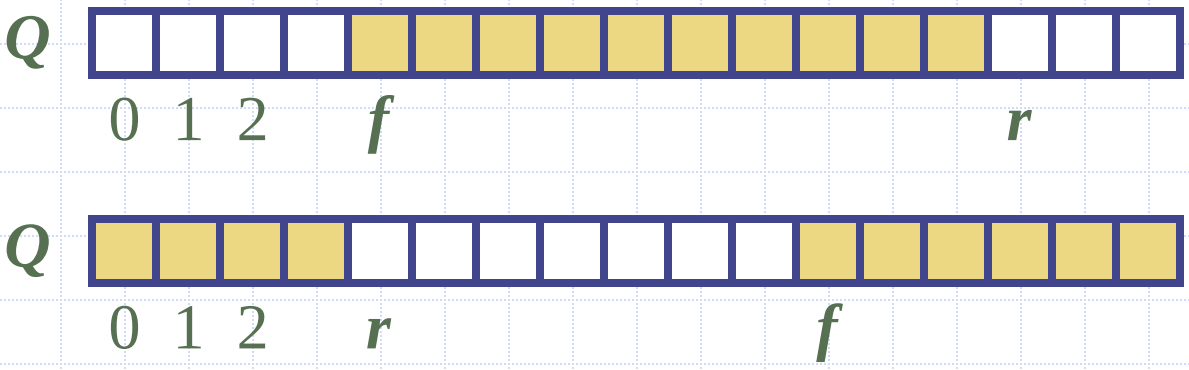
Algorithm *isEmpty*()
  return (*sz* == 0)



Queue Operations (cont.)

- Operation enqueue throws an exception if the array is full
- This exception is implementation-dependent

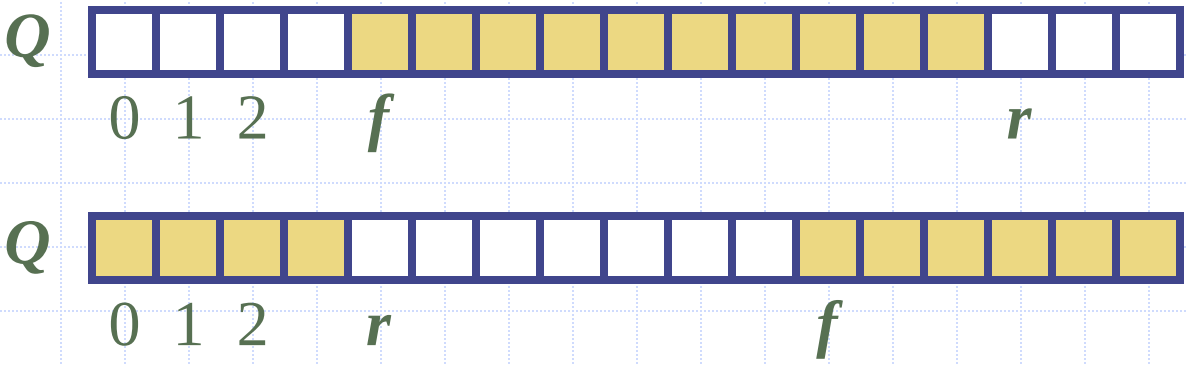
Algorithm *enqueue*(*o*)
  if *size*() = *N* **then**
    throw *IllegalStateException*
  else
    $r \leftarrow (f + sz) \bmod N$
    $Q[r] \leftarrow o$
    $sz \leftarrow (sz + 1)$



Queue Operations (cont.)

- Note that operation dequeue returns null if the queue is empty

Algorithm ***dequeue***()
  if *isEmpty*() **then**
    return *null*
  else
    $o \leftarrow Q[f]$
    $f \leftarrow (f + 1) \bmod N$
    $sz \leftarrow (sz - 1)$
    return o



Queue Interface in Java

- Java interface corresponding to our Queue ADT
- Assumes that first() and dequeue() return null if queue is empty

```
public interface Queue<E> {
   int size();
   boolean isEmpty();
   E first();
   void enqueue(E e);
   E dequeue();
}
```

Comparison to java.util.Queue

- Our Queue methods and corresponding methods of java.util.Queue:

Our Queue ADT	Interface java.util.Queue	
	throws exceptions	returns special value
enqueue(e)	add(e)	offer(e)
dequeue( )	remove( )	poll( )
first( )	element( )	peek( )
size( )	size( )	
isEmpty( )	isEmpty( )	

Application: Round Robin Schedulers

- We can implement a round robin scheduler using a queue Q by repeatedly performing the following steps:
  1. e = Q.dequeue()
  2. Service element e
  3. Q.enqueue(e)

轮询调度器的工作原理：
e = Q.dequeue()：首先从队列 Q 中取出一个元素 e，也就是将任务从队列的前端移除，准备执行。
Service element e：然后对元素 e（即任务）进行服务或执行。这一步通常是执行任务的实际工作。
Q.enqueue(e)：服务完成后，将任务 e 重新放入队列的末尾，等待下次轮到它继续执行。