

## 4. 栈与队列

### 队列接口与实现

邓俊辉

deng@tsinghua.edu.cn

## 操作与接口

❖ 队列 ( queue ) 也是受限的序列

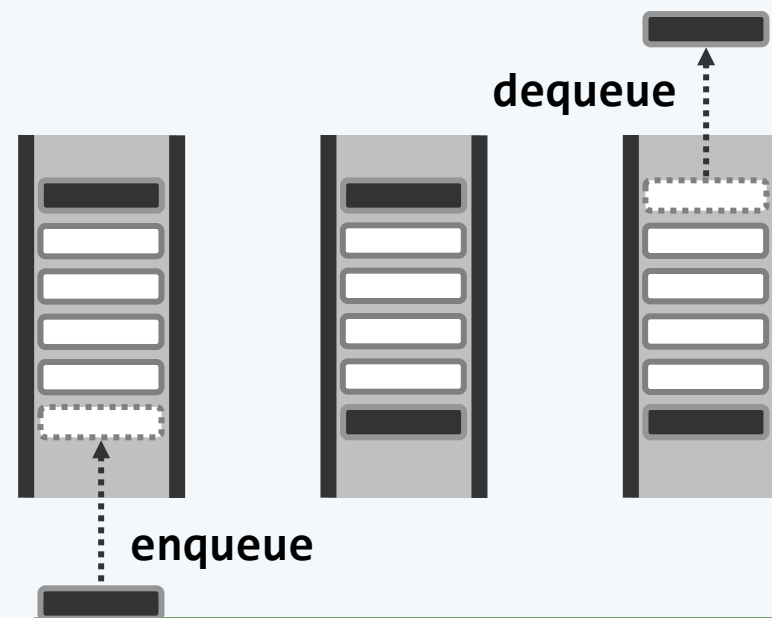
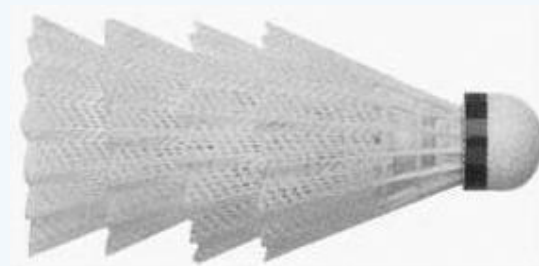
只能在队尾插入 ( 查询 ) :  
enqueue()  
rear()

只能在队头删除 ( 查询 ) :  
dequeue()  
front()

❖ 先进先出 ( FIFO )

后进后出 ( LILO )

❖ 扩展接口 : getMax()...



## 操作实例

操作	输出	队列（右侧为队头）		
Queue()				
empty()	true			
enqueue(5)		5		
enqueue(3)		3	5	
dequeue()	5	3		
enqueue(7)		7	3	
enqueue(3)		3	7	3
front()	3	3	7	3
empty()	false	3	7	3

操作	输出	队列（右侧为队头）					
enqueue(11)		11	3	7	3		
size()	4	11	3	7	3		
enqueue(6)		6	11	3	7	3	
empty()	false	6	11	3	7	3	
enqueue(7)		7	6	11	3	7	3
dequeue()	3	7	6	11	3	7	
dequeue()	7	7	6	11	3		
front()	3	7	6	11	3		
size()	4	7	6	11	3		

## 模板类

❖ 队列既然属于序列的特例，故亦可直接基于向量或列表派生

❖ `template <typename T> class Queue: public List<T> { //由列表派生的队列模板类`

`public: //size()与empty()直接沿用`

`void enqueue( T const & e ) { insertAsLast( e ); } //入队`

`T dequeue() { return remove( first() ); } //出队`

`T & front() { return first()->data; } //队首`

`}; //以列表首/末端为队列头/尾——颠倒过来呢？`

❖ 确认：如此实现的队列接口，均只需 $O(1)$ 时间

❖ 课后：基于向量，派生定义队列模板类

评测：你所实现的队列接口，效率如何？