Queue用户模拟队列这种数据结构，队列通常是指“先进先出”(FIFO，first-in-first-out)的容器。新元素插入（offer）到队列的尾部，访问元素（poll）操作会返回队列头部的元素。通常，队列不允许随机访问队列中的元素。

boolean	**add**(E e) 将指定的元素插入此队列（如果立即可行且不会违反容量限制），在成功时返回 true，如果当前没有可用的空间，则抛出 IllegalStateException。
	element() 获取，但是不移除此队列的头。
boolean	**offer**(E e) 将指定的元素插入此队列（如果立即可行且不会违反容量限制），当使用有容量限制的队列时，此方法通常要优于 add(E)，后者可能无法插入元素，而只是抛出一个异常。
	peek() 获取但不移除此队列的头；如果此队列为空，则返回 null。
	poll() 获取并移除此队列的头，如果此队列为空，则返回 null。
	remove() 获取并移除此队列的头。

实现类：

PriorityQueue

PriorityQueue保存队列元素的顺序不是按加入队列的顺序，而是按队列元素的大小进行重新排序。

排序方式

默认自然排序：数字大小排序，字符串按字典序排列。

Comparator(比较器)在队列实例化时指定排序方式。

如果多个元素都是最小值，则头是其中一个元素——选择方法是任意的。

注意：当PriorityQueue中没有指定Comparator时，加入PriorityQueue的元素必须实现了Comparable接口（即元素是可比较的），否则会导致 ClassCastException。

方法

boolean	**add**(E e) 将指定的元素插入此优先级队列。
void	**clear**() 从此优先级队列中移除所有元素。
Comparator<? super E>	**comparator**() 返回用来对此队列中的元素进行排序的比较器；如果此队列根据其元素的自然顺序进行排序，则返回 null。
boolean	**contains**(Object o) 如果此队列包含指定的元素，则返回 true。
Iterator<E>	**iterator**() 返回在此队列中的元素上进行迭代的迭代器。
boolean	**offer**(E e) 将指定的元素插入此优先级队列。
E	**peek**() 获取但不移除此队列的头；如果此队列为空，则返回 null。
E	**poll**() 获取并移除此队列的头，如果此队列为空，则返回 null。
boolean	**remove**(Object o) 从此队列中移除指定元素的单个实例（如果存在）。
int	**size**() 返回此 collection 中的元素数。
Object[]	**toArray**() 返回一个包含此队列所有元素的数组。
<T> T[]	**toArray**(T[] a) 返回一个包含此队列所有元素的数组；返回数组的运行时类型是指定数组的类型。

本质

动态数组。当添加元素到集合时，会先检查数组是否还有余量，有余量则把新元素加入集合，没余量则调用 `grow()`方法增加容量，然后调用`siftUp`将新加入的元素排序插入对应位置。PriorityQueue调用默认的构造方法时，使用默认的初始容量（`DEFAULT_INITIAL_CAPACITY=11`）创建一个 PriorityQueue，并根据其自然顺序来排序其元素。可指定初始容量，比较器 Comparator。

Deque接口

https://www.jianshu.com/p/35760d7bac0d