优先级队列

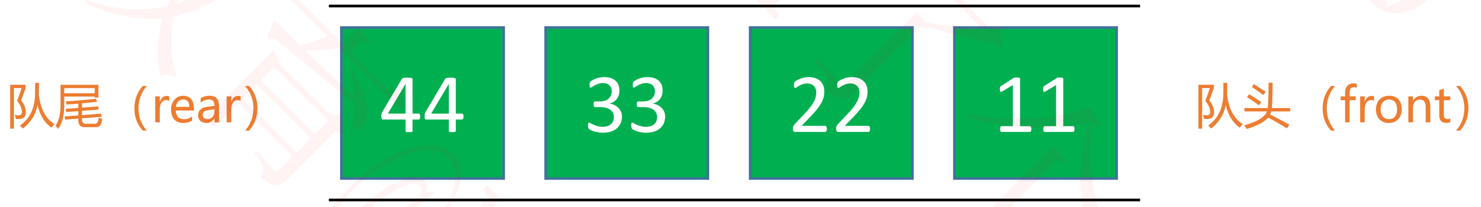
@M了个J

https://github.com/CoderMJLee

http://cnblogs.com/mjios

优先级队列（Priority Queue）

- 优先级队列也是个队列，因此也是提供以下接口
- `int size();  // 元素的数量`
- `boolean isEmpty();  // 是否为空`
- `void enQueue(E element);   // 入队`
- `E deQueue(); // 出队`
- `E front(); // 获取队列的头元素`
- `void clear();        // 清空`



- 普通的队列是 FIFO 原则，也就是先进先出

- 优先级队列则是按照优先级高低进行出队，比如将优先级最高的元素作为队头优先出队

优先级队列的应用场景举例

- ■ 医院的夜间门诊
  - ◻ 队列元素是病人
  - ◻ 优先级是病情的严重情况、挂号时间

- ■ 操作系统的多任务调度
  - ◻ 队列元素是任务
  - ◻ 优先级是任务类型

优先队列的底层实现

- 根据优先队列的特点，很容易想到：可以直接利用二叉堆作为优先队列的底层实现

- 可以通过 Comparator 或 Comparable 去自定义优先级高低

```java
public class Person implements Comparable<Person> {
    private String name;
    private int boneBreak;
    public Person(String name, int boneBreak) {
        this.name = name;
        this.boneBreak = boneBreak;
    }
    @Override
    public String toString() {
        return "Person [name=" + name + ", boneBreak=" + boneBreak + "]";
    }
    @Override
    public int compareTo(Person o) {
        return boneBreak - o.boneBreak;
    }
}
```

```java
PriorityQueue<Person> queue = new PriorityQueue<>();
queue.enQueue(new Person("jack", 1));
queue.enQueue(new Person("rose", 3));
queue.enQueue(new Person("jim", 2));
queue.enQueue(new Person("kate", 5));
queue.enQueue(new Person("larry", 10));
while (!queue.isEmpty()) {
    System.out.println(queue.deQueue());
}
```

```
Person [name=larry, boneBreak=10]
Person [name=kate, boneBreak=5]
Person [name=rose, boneBreak=3]
Person [name=jim, boneBreak=2]
Person [name=jack, boneBreak=1]
```

作业

- 数组中的第K个最大元素：https://leetcode-cn.com/problems/kth-largest-element-in-an-array/
- 根据字符出现频率排序：https://leetcode-cn.com/problems/sort-characters-by-frequency/
- 数据流中的第K大元素：https://leetcode-cn.com/problems/kth-largest-element-in-a-stream/
- 有序矩阵中第K小的元素：https://leetcode-cn.com/problems/kth-smallest-element-in-a-sorted-matrix/
- 前K个高频元素：https://leetcode-cn.com/problems/top-k-frequent-elements/
- 前K个高频单词：https://leetcode-cn.com/problems/top-k-frequent-words/
- 查找和最小的K对数字：https://leetcode-cn.com/problems/find-k-pairs-with-smallest-sums/
- 合并K个排序链表：https://leetcode-cn.com/problems/merge-k-sorted-lists/