目录

一、实验题目.....2
二、实验目的.....2
三、实验设计与分析.....3
1、Implement exercise 2.3-7.....3
2、Implement priority queue.....3
3、Implement Quicksort and answer the following questions.....3
4、Give an O(lg m + lgn) time algorithm for computing the kth largest element in the union of the two lists.....4
四、实验环境.....4
五、项目测试.....5
1、Implement exercise 2.3-7.....5
2、Implement priority queue.....5
3、Implement Quicksort and answer the following questions.....6
4、Give an O(lg m + lgn) time algorithm for computing the kth largest element in the union of the two lists.....6

一、实验题目

1.Implement exercise 2.3-7.

2.Implement priority queue.

3.Implement Quicksort and answer the following questions. (1) How many comparisons will Quicksort do on a list of n elements that all have the same value? (2) What are the maximum and minimum number of comparisons will Quicksort do on a list of n elements, give an instance for maximum and minimum case respectively.

4.Give a divide and conquer algorithm for the following problem: you are given two sorted lists of size m and n, and are allowed unit time access to the ith element of each list. Give an $O(\lg m + \lg n)$ time algorithm for computing the kth largest element in the union of the two lists. (For simplicity, you can assume that the elements of the two lists are distinct).

二、实验目的

掌握分治法的思想和步骤，并能够用代码实现。

1、实现一个能确定整数数组中是否存在两元素之和为整数 x 的算法，要求时间复杂度为 O(nlogn)。

2、使用其他数据结构，设计和实现一个优先队列。

3、实现快排并且分析元素全部相同的数组比较的次数和最坏/好情况下的比较次数。

4、实现一个能够找到两个有序数组合并后的第 K 大元素的算法，要求时间复杂度为 O(logm+logn)。

三、实验设计与分析

1、Implement exercise 2.3-7.

因为要求算法时间复杂度为为O(nlogn)，所以不能使用暴力解法，可以选择先使用时间复杂度为O(logn)的排序算法排序后再进行二分查找。常用的排序算法中快排的最差时间复杂度为 $O(n^2)$，归并排序和堆排序的最好最坏情况时间复杂度都是 O(nlogn)，这里选择归并排序。二分查找则是对排序后的数组进行遍历，针对每个元素判断是否大于 x，若小于 x 则用二分查找查找 x 与此元素的差值，否则算法结束。

如果使用哈希表，对每个元素 s 查找哈希表中是否存在两数之和 X-s，可以使时间复杂度达到O(n)。

2、Implement priority queue.

优先队列是最高优先级的元素最先出队，由于大顶堆每个结点的值都大于或等于其左右孩子结点的值，可以十分方遍的找到优先级最高的元素，所以使用大顶堆实现优先队列。

构造大顶堆需要实现 sink（下沉）和 swim（上浮）函数，sink 将父节点和较大的子节点比较，如果子节点大则交换位置并继续比较，否则结束比较；swim 是子节点与父节点比较，如果子节点大则交换位置并继续比较，否则结束比较。二者分别自顶向下和自下向上保持堆有序。

优先队列的两个基本功能是 push 和 pop，Push 操作将元素插入末尾，这样可以保持其他部分满足大顶堆的性质，执行 swim 保持堆有序；Pop 操作去除堆顶元素后将末尾元素放到顶端，这样可以保持除了堆顶元素其他元素都满足大顶堆的性质，执行 sink 保持堆有序。

3、Implement Quicksort and answer the following questions.

快速排序可以使用递归的方式实现，将数组左侧第一个元素作为 pivot，遍历数组将比 pivot 小的元素都交换到前面，遍历结束后满足[left+1,j] <pivot，

(j,i]>=pivot，将 pivot 交换至第 j 个位置。然后对第 j 个位置前后的数组递归。

当数组元素全部相同时，时间复杂度为 $O(n^2)$，比较次数为(n*(n-1))/2；最坏情况是已经被正序或倒序排好的数组，这时每次分治都只比上一次分治少一个元素，递归树退化成为链表，前文所述的元素全部相同是以排好序的特殊情况，所以时间复杂度和比较次数也是 $O(n^2)$和(n*(n-1))/2；最好情况是每次分治都能将数组均匀分开，递归树是平衡二叉树，时间复杂度为 O(nlogn)。

由于最坏情况只会在数组已经有序时出现，所以可以随机取 pivot 来避免最坏情况。

4、Give an O(lg m + lgn) time algorithm for computing the kth largest element in the union of the two lists.

由于时间复杂度要求为 O(lgm+lgn),所以不能遍历数组。采取分治的思想比较两数组第 K/2 个元素的值，每次将范围缩小一半。由于排序数组一般是升序，所以可以将问题转化为求第(m+n-k+1)小的元素。将两数组都二分，共有四种情况：第一种是两数组中点前元素个数之和大于 K 个且数组 A 的中间元素小于数组 B，此时第 K 个元素不会在 B 的后半部分，只需递归取数组 A 和数组 B 的前半部分；第二种是两数组中点前元素个数之和大于 K 个且数组 A 的中间元素大于等于数组 B，此时第 K 个元素不会在 A 的后半部分，只需递归取数组 B 和数组 A 的前半部分；第三种是两数组中点前元素个数之和小于 K 个且数组 A 的中间元素小于数组 B，此时第 K 个元素不会在 A 的前半部分，只需递归取数组 B 和数组 A 的后半部分；第四种是两数组中点前元素个数之和小于 K 个且数组 A 的中间元素大于等于数组 B，此时第 K 个元素不会在 B 的前半部分，只需递归取数组 A 和数组 B 的后半部分。

四、实验环境

Windows 10、IntelliJ IDEA 2020.3.2 x64、jdk1.8.0_281

五、项目测试

1、Implement exercise 2.3-7.

如下图所示，对于输入的数组首先进行了排序，之后会给出数组中是否存在两数之和为某个值的判断。对于第一个数组存在两元素之和为 8，打印出 exist，对于第二个数组不存在两元素之和为 97，打印出 no exist。

```
数组排序前：[9, 8, 7, 6, 5, 4, 3, 2, 1]
数组排序后：[1, 2, 3, 4, 5, 6, 7, 8, 9]
数组S中是否存在两元素之和为8: exist
数组排序前：[99, 11, 22, 88, 33, 77, 55, 66, 100]
数组排序后：[11, 22, 33, 55, 66, 77, 88, 99, 100]
数组S中是否存在两元素之和为97: no exist
```

算法时间复杂度为 O(nlogn)

2、Implement priority queue.

如下图所示，对于初始容量为 5 的优先队列先 push 后 pop，对数组打印可以验证优先队列能够维持大顶堆，并且实现了优先级高的先出。例如执行 Insert(10)时，10 取代 9 成为了堆顶，其他元素位置发生相应改变但依然满足大顶堆的规则；执行 Extract_MAX 操作时，按照优先级大小依次弹出；执行 Increase_key(100,5)时，将 5 增加 100；执行 Maximum 操作时，返回优先队列

```
优先队列插入9后为[null, 9, null, null, null, null]
此时优先队列最大值为：9
优先队列插入5后为[null, 9, 5, null, null, null]
此时优先队列最大值为：9
优先队列插入1后为[null, 9, 5, 1, null, null]
此时优先队列最大值为：9
优先队列插入8后为[null, 9, 8, 1, 5, null]
此时优先队列最大值为：9
优先队列插入10后为[null, 10, 9, 1, 5, 8]
此时优先队列最大值为：10
将5增加100
此时优先队列最大值为：105
已取出优先队列最大值105
已取出优先队列最大值10
已取出优先队列最大值9
已取出优先队列最大值8
已取出优先队列最大值1
```

中最大值。

3、Implement Quicksort and answer the following questions.

如下图，无论排序前数组是否有序，排序后数组均有序，排序算法正确；通过记录比较次数，数组中元素全部相等和数组中元素升序/降序排列时比较次数为 36 次，与计算得到的最差情况(9*(9-1)/2 相等。随机取 pivot 后再进行排序，比较次数明显降低，算法性能变好

```
排序前数组为：[1, 1, 1, 1, 1, 1, 1, 1, 1]
排序后数组为：[1, 1, 1, 1, 1, 1, 1, 1, 1]
比较次数为： 36
排序前数组为：[9, 8, 7, 6, 5, 4, 3, 2, 1]
排序后数组为：[1, 2, 3, 4, 5, 6, 7, 8, 9]
比较次数为： 36
排序前数组为：[1, 2, 3, 4, 5, 6, 7, 8, 9]
排序后数组为：[1, 2, 3, 4, 5, 6, 7, 8, 9]
比较次数为： 36
排序前数组为：[5, 1, 9, 3, 7, 4, 8, 6, 2]
排序后数组为：[1, 2, 3, 4, 5, 6, 7, 8, 9]
比较次数为： 17
排序前数组为：[1, 1, 1, 1, 1, 1, 1, 1, 1]
排序后数组为：[1, 1, 1, 1, 1, 1, 1, 1, 1]
比较次数为： 36
排序前数组为：[9, 8, 7, 6, 5, 4, 3, 2, 1]
排序后数组为：[1, 2, 3, 4, 5, 6, 7, 8, 9]
比较次数为： 17
排序前数组为：[1, 2, 3, 4, 5, 6, 7, 8, 9]
排序后数组为：[1, 2, 3, 4, 5, 6, 7, 8, 9]
比较次数为： 18
排序前数组为：[5, 1, 9, 3, 7, 4, 8, 6, 2]
排序后数组为：[1, 2, 3, 4, 5, 6, 7, 8, 9]
比较次数为： 18
```

4、Give an O(lg m + lgn) time algorithm for computing the kth largest element in the union of the two lists.

如下图，可以准确找到第 K 大的元素，即使有重复元素也不影响答案。

```
有序数组A：[1, 50, 300, 500, 700, 900]
有序数组B：[2, 100, 200, 500, 600, 800]
数组A和数组B的第7大元素为：300
```