

LintCode领扣题解 (/problem) / 第k大元素 · Kth Largest Element

第k大元素 · Kth Largest Element

中文

- 脸书 (/problem/?tags=facebook)
- 谷歌 (/problem/?tags=google)
- 快速排序 (/problem/?tags=quick-sort)
- 排序 (/problem/?tags=sort)

描述

在数组中找到第 k 大的元素。

❗ 你可以交换数组中的元素的位置

样例

样例 1:

输入:
n = 1, nums = [1,3,4,2]
输出:
4

样例 2:

输入:
n = 3, nums = [9,3,2,4,8]
输出:
4

挑战


要求时间复杂度为O(n)，空间复杂度为O(1)。

在线评测地址: <https://www.lintcode.com/problem/kth-largest-element/> (<https://www.lintcode.com/problem/kth-largest-element/>)

收起题目描述 ^

- 语言类型
- ALL (35)
- python (12)
- java (11)
- cpp (9)
- javascript (2)
- golang (1)

上传题解




令狐冲

更新于 6/23/2020, 5:29:25 PM

九章算法强化班里讲过的标准 Partition 模板。

邀请有礼

invitation/show



```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm\_source=code
 */
class Solution {
    /**
     * @param k : description of k
     * @param nums : array of nums
     * @return: description of return
     */
    public int kthLargestElement(int k, int[] nums) {
        if (nums == null || nums.length == 0 || k < 1 || k > nums.length){
            return -1;
        }
        return partition(nums, 0, nums.length - 1, nums.length - k);
    }

    private int partition(int[] nums, int start, int end, int k) {
        if (start >= end) {
            return nums[k];
        }

        int left = start, right = end;
        int pivot = nums[(start + end) / 2];

        while (left <= right) {
            while (left <= right && nums[left] < pivot) {
                left++;
            }
            while (left <= right && nums[right] > pivot) {
                right--;
            }
            if (left <= right) {
                swap(nums, left, right);
                left++;
                right--;
            }
        }

        if (k <= right) {
            return partition(nums, start, right, k);
        }
        if (k >= left) {
            return partition(nums, left, end, k);
        }
        return nums[k];
    }

    private void swap(int[] nums, int i, int j) {
        int tmp = nums[i];
        nums[i] = nums[j];
        nums[j] = tmp;
    }
};
```

👍 获赞 20

💬 14 条评论

你的口袋题库

2000+ 算法真题、国内外名企题库免费开放



九章算法APP

令狐冲

更新于 7/19/2020, 9:13:02 PM

使用九章算法强化班中讲过的 partition 的标准模板

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
class Solution:
    # @param k & A a integer and an array
    # @return ans a integer
    def kthLargestElement(self, k, A):
        if not A or k < 1 or k > len(A):
            return None
        return self.partition(A, 0, len(A) - 1, len(A) - k)

    def partition(self, nums, start, end, k):
        """
        During the process, it's guaranteed start <= k <= end
        """
        if start == end:
            return nums[k]

        left, right = start, end
        pivot = nums[(start + end) // 2]
        while left <= right:
            while left <= right and nums[left] < pivot:
                left += 1
            while left <= right and nums[right] > pivot:
                right -= 1
            if left <= right:
                nums[left], nums[right] = nums[right], nums[left]
                left, right = left + 1, right - 1

        # left is not bigger than right
        if k <= right:
            return self.partition(nums, start, right, k)
        if k >= left:
            return self.partition(nums, left, end, k)

        return nums[k]
```

👍 获赞 7

💬 14 条评论

DDBear

更新于 10/11/2020, 6:10:52 PM

算法: 快速选择算法

最容易想到的就是直接排序，返回第k大的值。时间复杂度是 $O(n\log n)$ ，这里提供 $O(n)$ 的解法。

这题其实是快速排序算法的变体，通过快速排序算法的 `partition` 步骤，可以将小于 `pivot` 的值划分到 `pivot` 左边，大于 `pivot` 的值划分到 `pivot` 右边，所以可以直接得到 `pivot` 的 `rank`。从而缩小范围继续找第 k 大的值。

`partition` 步骤：

1. 令 `left = start`，`right = end`，`pivot = nums[left]`。
2. 当 `nums[left] < pivot` 时，`left` 指针向右移动。
3. 当 `nums[right] > pivot` 时，`right` 指针向左移动。
4. 交换两个位置的值，`right` 指针左移，`left` 指针右移。
5. 直到两指针相遇，否则回到第 2 步。

每次 `partition` 后根据 `pivot` 的位置，寻找下一个搜索的范围。

复杂度分析

设数组长度为 `n`

时间复杂度 $O(n)$

- 对一个数组进行 `partition` 的时间复杂度为 $O(n)$ 。
- 分治，选择一边继续进行 `partition`。
- 所以总的复杂度为 $T(n) = T(n / 2) + O(n)$ ，总时间复杂度依然为 $O(n)$ 。

空间复杂度 $O(1)$

- 只需要快速选择游标的 $O(1)$ 额外空间。

源代码

c++

java

python

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
public class Solution {
    /**
     * @param n: An integer
     * @param nums: An array
     * @return: the Kth largest element
     */
    public int kthLargestElement(int k, int[] nums) {
        int n = nums.length;
        // 为了方便编写代码, 这里将第 k 大转换成第 k 小问题。
        k = n - k;
        return partition(nums, 0, n - 1, k);
    }
    public int partition(int[] nums, int start, int end, int k) {
        int left = start, right = end;
        int pivot = nums[left];

        while (left <= right) {
            while (left <= right && nums[left] < pivot) {
                left++;
            }
            while (left <= right && nums[right] > pivot) {
                right--;
            }
            if (left <= right) {
                swap(nums, left, right);
                left++;
                right--;
            }
        }

        // 如果第 k 小在右侧, 搜索右边的范围, 否则搜索左侧。
        if (k <= right) {
            return partition(nums, start, right, k);
        }
        if (k >= left) {
            return partition(nums, left, end, k);
        }
        return nums[k];
    }
    public void swap(int[] nums, int x, int y) {
        int temp = nums[x];
        nums[x] = nums[y];
        nums[y] = temp;
    }
}
```

👍 获赞 6

💬 1 条评论



九章管理员

更新于 6/9/2020, 7:03:46 AM

标准的 Quick Select 算法

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm\_source=code
 */
class Solution {
    /**
     * @param k : description of k
     * @param nums : array of nums
     * @return: description of return
     */
    public int kthLargestElement(int k, int[] nums) {

        // write your code here
        int low = 0, high = nums.length - 1;
        while(low <= high){
            int index = low-1;
            for(int i = low; i < high; i++){
                if(nums[i] > nums[high]){
                    swap(nums, i, ++index);
                }
            }
            swap(nums, ++index, high);
            if(index == k - 1){
                return nums[index];
            }
            if(index < k - 1){
                low = index + 1;
            }else{
                high = index - 1;
            }
        }
        return -1;
    }

    private void swap(int[] nums, int a, int b){
        int temp = nums[a];
        nums[a] = nums[b];
        nums[b] = temp;
    }
};
```

👍 获赞 6

💬 6 条评论



令狐冲

更新于 6/9/2020, 7:03:58 AM

九章算法强化班里讲过的 partition 标准模板

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm\_source=code
 */
class Solution {
public:
    /**
     * param k : description of k
     * param nums : description of array and index 0 ~ n-1
     * return: description of return
     */
    int kthLargestElement(int k, vector<int> nums) {
        if (nums.size() == 0 || k < 1 || k > nums.size()){
            return -1;
        }
        return partition(nums, 0, nums.size() - 1, nums.size() - k);
    }

private:
    int partition(vector<int> &nums, int start, int end, int k) {
        if (start >= end) {
            return nums[k];
        }

        int left = start, right = end;
        int pivot = nums[(start + end) / 2];

        while (left <= right) {
            while (left <= right && nums[left] < pivot) {
                left++;
            }
            while (left <= right && nums[right] > pivot) {
                right--;
            }
            if (left <= right) {
                swap(nums, left, right);
                left++;
                right--;
            }
        }

        if (k <= right) {
            return partition(nums, start, right, k);
        }
        if (k >= left) {
            return partition(nums, left, end, k);
        }
        return nums[k];
    }

    void swap(vector<int> &nums, int i, int j) {
        int tmp = nums[i];
        nums[i] = nums[j];
        nums[j] = tmp;
    }
};
```

👍 获赞 1

💬 3 条评论

讲座

更新于 6/9/2020, 7:04:25 AM

quick select

```
/**
 * 本参考程序由九章算法用户提供。版权所有，转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作，授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括：九章算法班 2020升级版，算法强化班，算法基础班，北美算法面试高频题班，Java 高级工程师 P6+ 小班课，面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括：系统设计 System Design，面向对象设计 OOD
 * - 专题及项目类课程包括：动态规划专题班，Big Data - Spark 项目实战，Django 开发项目课
 * - 更多详情请见官方网站：http://www.jiuzhang.com/?utm_source=code
 */
/**
 * 本代码由九章算法编辑提供。没有版权欢迎转发。
 * - 九章算法致力于帮助更多中国人找到好的工作，教师团队均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的面试培训课程包括：九章算法班，系统设计班，BAT国内班
 * - 更多详情请见官方网站：http://www.jiuzhang.com/
 */

class Solution {
    /*
     * @param k : description of k
     * @param nums : array of nums
     * @return: description of return
     */
    public int kthLargestElement(int k, int[] nums) {
        // write your code here
        if (nums == null || nums.length == 0) {
            return 0;
        }
        if (k <= 0) {
            return 0;
        }
        return helper(nums, 0, nums.length - 1, nums.length - k + 1);
    }

    public int helper(int[] nums, int l, int r, int k) {
        if (l == r) {
            return nums[l];
        }
        int position = partition(nums, l, r);
        if (position + 1 == k) {
            return nums[position];
        } else if (position + 1 < k) {
            return helper(nums, position + 1, r, k);
        } else {
            return helper(nums, l, position - 1, k);
        }
    }

    public int partition(int[] nums, int l, int r) {
        // 初始化左右指针和pivot
        int left = l, right = r;
        int pivot = nums[left];

        // 进行partition
        while (left < right) {
            while (left < right && nums[right] >= pivot) {
                right--;
            }
            nums[left] = nums[right];
            while (left < right && nums[left] <= pivot) {
                left++;
            }
        }
    }
}
```



```
        nums[right] = nums[left];
    }

    // 返还pivot点到数组里面
    nums[left] = pivot;
    return left;
}
};
```

👍 获赞 0

💬 添加评论



Panda

更新于 6/9/2020, 7:03:46 AM

用quick sort算法，降序排列数组，然后在排序的过程中找到第k大的元素

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ /
Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
public class Solution {
    public int kthLargestElement(int n, int[] nums) {
        if (nums == null || nums.length == 0) {
            return -1;
        }
        //quick select find nth element, n should be zero based
        return quickSelect(n - 1, 0, nums.length - 1, nums);
    }
    private int quickSelect(int k, int start, int end, int[] nums) {
        if (start == end) {
            return nums[start];
        }

        int pivot = nums[start + (end - start) / 2];
        int left = start, right = end;

        while (left <= right) { //sort entire array as descending order
            while (left <= right && nums[left] > pivot) {
                left++;
            }
            while (left <= right && nums[right] < pivot) {
                right--;
            }

            if (left <= right) {
                int tmp = nums[left];
                nums[left] = nums[right];
                nums[right] = tmp;
                left++;
                right--;
            }
        }

        if (k >= start && k <= right) {
            return quickSelect(k, start, right, nums);
        } else if (k >= left && k <= end) {
            return quickSelect(k, left, end, nums);
        }
        return nums[right + 1];
    }
}
```

👍 获赞 5

💬 3 条评论

**Jet**

更新于 8/31/2020, 3:48:47 AM

Quick Select.Quick SelectQuick Select

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm\_source=code
 */
class Solution {
public:
    /**
     * @param n: An integer
     * @param nums: An array
     * @return: the Kth largest element
     */
    int kthLargestElement(int n, vector<int> &nums) {
        // write your code here
        if(nums.size()==0||nums.size()<n){
            return -1;
        }
        return quick_select(nums,n,0,nums.size()-1);
    }

    int quick_select(vector<int>& nums, int k, int start, int end){
        if(start==end){
            return nums[start];
        }
        int left=start;
        int right=end;
        int middle=nums[(start+end)/2];
        while(left<=right){
            while(left<=right && nums[left]>middle){
                left++;
            }
            while(left<=right && nums[right]<middle){
                right--;
            }
            if(left<=right){
                int temp=nums[left];
                nums[left]=nums[right];
                nums[right]=temp;
                left++;
                right--;
            }
        }
        if(k<=right-start+1){
            return quick_select(nums,k ,start,right);
        }
        if(k>left-start){
            return quick_select(nums,k-(left-start) ,left,end);
        }
        return nums[right+1];
    }
};
```

👍 获赞 3

💬 添加评论

**Tony**

更新于 6/9/2020, 7:03:48 AM

使用Quick Select查找第k大元素。平均时间复杂度 $O(n)$, 最差是 $O(n^2)$, 当输入数组是按从最大值到最小值排序时发生, 此时每次最外层的循环仅可排除一个元素。空间复杂度 $O(1)$ 。使用循环替代尾递归, 降低了对栈空间的消耗。

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm\_source=code
 */
class Solution {
    /**
     * @param k : description of k
     * @param nums : array of nums
     * @return: description of return
     */
    public int kthLargestElement(int k, int[] nums) {
        // write your code here
        return quickSelect(nums, 0, nums.length - 1, k);
    }

    private int quickSelect(int[] nums, int start, int end, int k){
        while(true){
            if (start >= end){
                return nums[end];
            }

            int pivot = nums[start];
            int i = start, j = end + 1;
            int temp;
            while (true){
                while (nums[++i] > pivot && i < end){};
                while (nums[--j] < pivot && j > start){};
                if (i >= j){
                    break;
                }
                temp = nums[i];
                nums[i] = nums[j];
                nums[j] = temp;
            }
            nums[start] = nums[j];
            nums[j] = pivot;

            if (j == k - 1){
                return nums[j];
            }
            if (j > k - 1){
                end = j - 1;
            } else {
                start = j + 1;
            }
        }
    }
};
```

👍 获赞 3

💬 添加评论



九章用户43XEJ8

更新于 6/9/2020, 7:03:47 AM

九章的高票模板已经足够干净整洁了, 有两点小瑕疵。我的代码第 20 行 和 第 44 行对比九章的。这个版本应该是最干净的了。

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
public class Solution {
    public int findKthLargest(int[] nums, int k) {
        if (nums == null || nums.length == 0 || k <= 0 || k > nums.length) {
            return -1;
        }
        return partition(nums, 0, nums.length - 1, nums.length - k);
    }

    public int partition(int[] nums, int start, int end, int k) {
        if (start == end) {
            return nums[k];
        }
        int pivot = nums[(start + end) / 2];
        int left = start;
        int right = end;
        while (left <= right) {
            while (left <= right && nums[left] < pivot) {
                left++;
            }
            while (left <= right && nums[right] > pivot) {
                right--;
            }
            if (left <= right) {
                swap(nums, left, right);
                left++;
                right--;
            }
        }
        if (k <= right) {
            return partition(nums, start, right, k);
        }
        return partition(nums, left, end, k);
    }

    private void swap(int[] nums, int a, int b) {
        int tmp = nums[a];
        nums[a] = nums[b];
        nums[b] = tmp;
    }
}
```

👍 获赞 3

💬 添加评论



九章用户HLV3F5

更新于 12/1/2020, 6:31:37 PM

QuickSelect的思想, 递归寻找pivotIndex==k, 结构清楚, 简单易懂。

```
/**
 * 本参考程序由九章算法用户提供。版权所有，转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作，授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括：九章算法班 2020升级版，算法强化班，算法基础班，北美算法面试高频题班，Java 高级工程师 P6+ 小班课，面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括：系统设计 System Design，面向对象设计 OOD
 * - 专题及项目类课程包括：动态规划专题班，Big Data - Spark 项目实战，Django 开发项目课
 * - 更多详情请见官方网站：http://www.jiuzhang.com/?utm_source=code
 */
public class Solution {
    /**
     * @param n: An integer
     * @param nums: An array
     * @return: the Kth largest element
     */
    public int kthLargestElement(int n, int[] nums) {
        return partition(n, nums, 0, nums.length - 1);
    }

    private int partition(int n, int[] nums, int start, int end) {
        int pivot = nums[end];
        int pivotIndex = start;
        for (int i = start; i < end; i++) {
            if (nums[i] > pivot) {
                swap(nums, i, pivotIndex);
                pivotIndex++;
            }
        }

        swap(nums, pivotIndex, end);

        if (pivotIndex == n - 1) {
            return nums[pivotIndex];
        }

        if (pivotIndex > n - 1) {
            return partition(n, nums, start, pivotIndex - 1);
        } else {
            return partition(n, nums, pivotIndex + 1, end);
        }
    }

    private void swap(int[] nums, int a, int b){
        int temp = nums[a];
        nums[a] = nums[b];
        nums[b] = temp;
    }
}
```

👍 获赞 1 💬 添加评论



九章用户Z1607T

更新于 8/10/2020, 2:23:51 PM

QuickSelect解法，不在递归的时候返回nums[k]的版本

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm\_source=code
 */
def kthLargestElement(self, k, nums):
    if not nums or k <= 0 or k > len(nums):
        return None

    self.quickSelect(nums, 0, len(nums) - 1, len(nums) - k)

    return nums[len(nums) - k]

def quickSelect(self, nums, start, end, k):
    if start == end:
        return

    left, right = start, end
    pivot = nums[(start + end) // 2]

    while left <= right:
        while left <= right and nums[left] < pivot:
            left += 1
        while left <= right and nums[right] > pivot:
            right -= 1
        if left <= right:
            nums[left], nums[right] = nums[right], nums[left]
            left += 1
            right -= 1

    if k <= right:
        self.quickSelect(nums, start, right, k)
    if k >= left:
        self.quickSelect(nums, left, end, k)
```

👍 获赞 1 💬 添加评论



九章用户NCSGPU

更新于 6/9/2020, 7:03:57 AM

quick select, not need to re-calculate k T: O(n), template works for both quick sort and quick select

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm\_source=code
 */
public int kthLargestElement(int k, int[] nums) {
    // write your code here
    return quickSelect(nums, 0, nums.length - 1, nums.length - k);
}
private int quickSelect(int[] nums, int start, int end, int k) {
    // System.out.println(start + " " + end + " " + k);
    if(start >= end) return nums[end];
    int left = start, right = end;
    int pivot = nums[(start + end) / 2];
    while (left <= right) {
        while(left <= right && nums[left] < pivot) left++;
        while(left <= right && nums[right] > pivot) right--;
        if (left <= right) {
            // swap
            int temp = nums[left];
            nums[left] = nums[right];
            nums[right] = temp;
            // check boundry for special case, can keep in both quick sort or quick select
            if (left < end) left++;
            if (right > start) right--;
        }
    }
    // System.out.println(left + " " + right);
    // figure out which range we should keep looking at
    if (k <= right) {
        return quickSelect(nums, start, right, k);
    } // reduce branch to save time
    // } else if (k > right && k <= left) {
    //     return quickSelect(nums, right + 1, left, k);
    // } else {
    return quickSelect(nums, right + 1, end, k);
    }
}
```

👍 获赞 1

💬 2 条评论



cc189

更新于 6/9/2020, 7:03:55 AM


```

/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
class Solution:
    def kthLargestElement(self, n, nums):
        self.k = len(nums) - n
        self.quickselect(nums, 0, len(nums) - 1)
        return nums[-n]

    def quickselect(self, nums, l, r):
        if l >= r:
            return
        ll, rr = self.partition(nums, l, r)
        if self.k <= ll:
            self.quickselect(nums, l, ll)
        if self.k >= rr:
            self.quickselect(nums, rr, r)

    def partition(self, nums, l, r):
        pivot = nums[(r - l) // 2 + l]
        while r >= l:
            while r >= l and nums[l] < pivot:
                l += 1
            while r >= l and nums[r] > pivot:
                r -= 1
            if r >= l:
                nums[l], nums[r] = nums[r], nums[l]
                l += 1
                r -= 1
        return r, l

```

👍 获赞 1 💬 添加评论



Yummy Corgi

更新于 6/9/2020, 7:03:52 AM

quickselect based on partition (3-way)

recursive and iterative quickselect

```

/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
class Solution {
public:
    /**
     * @param n: An integer
     * @param nums: An array
     * @return: the Kth largest element
     */
    int kthLargestElement(int k, vector<int> &A) {

```

```
// write your code here
int n = A.size();
if (n == 0 || k < 0 || k > n) {
    return -1;
}
// return quickSelect(A, 0, n - 1, n - k);
return quickSelect(A, n - k);
}

private:
// iterative
int quickSelect(vector<int> &A, int k) {
    int left = 0, right = A.size() - 1;
    while (left <= right) {
        int p = partition(A, left, right);
        if (p == k) {
            return A[k];
        } else if (p > k) {
            right = p - 1;
        } else {
            left = p + 1;
        }
    }
    return -1;
}

// recursion
int quickSelect(vector<int> &A, int left, int right, int k) {
    if (left == right) {
        return A[left];
    }
    int p = partition(A, left, right);
    if (p == k) {
        return A[k];
    }
    if (p > k) {
        return quickSelect(A, left, p - 1, k);
    } else {
        return quickSelect(A, p + 1, right, k);
    }
}

int partition(vector<int> &A, int left, int right) {
    if (left == right) {
        return left;
    }
    int mid = left + (right - left) / 2;
    int pivot = A[mid];
    int l = left, i = left, r = right;
    while (i <= r) {
        if (A[i] < pivot) {
            swap(A[l++], A[i++]);
        } else if (A[i] > pivot) {
            swap(A[i], A[r--]);
        } else {
            i++;
        }
    }
    return l;
}
};
```

👍 获赞 1

💬 添加评论



九章用户QQPIZT

更新于 12/14/2020, 3:56:14 AM

思路和大家一样, 也是快排 def kthLargestElement(self, n, nums):

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ /
Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
# write your code here
def res(alist, start, end, n):
    if start >= end:
        return None
    low = start
    high = end - 1
    tmp = alist[low]

    while low < high:
        while low < high and alist[high] < tmp:
            high -= 1
        alist[low] = alist[high]
        while low < high and alist[low] >= tmp:
            low += 1
        alist[high] = alist[low]
    alist[low] = tmp
    if low == n-1:
        return alist[low]
    elif low < n-1:
        return res(alist, low+1, end, n)
    else:
        return res(alist, start, low, n)
return res(nums, 0, len(nums), n)
```

👍 获赞 0 💬 添加评论



九章用户5ZX MVC

更新于 10/21/2020, 11:00:30 PM

判断数组是否为空, k是否合理, 然后对数组进行简单的sort倒序, 取第k个值, 即第k大的值

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ /
Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
class Solution:
    """
    @param n: An integer
    @param nums: An array
    @return: the Kth largest element
    """
    def kthLargestElement(self, n, nums):
        # write your code here
        assert nums and 0 < n <= len(nums)
        nums.sort(reverse=True)
        return nums[n-1]
```

👍 获赞 0 💬 添加评论



Su

更新于 8/5/2020, 3:22:54 AM

使用快选, 其实也可以直接求第 K 大的数, 不必转换为求第 $\text{len}(\text{nums})-k$ 小的数。就是在边界上处理了一下。

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
class Solution:
    """
    @param n: An integer
    @param nums: An array
    @return: the Kth largest element
    """
    def kthLargestElement(self, n, nums):
        self.quickselect(nums, 0, len(nums) - 1, n)
        return nums[-n]

    def quickselect(self, nums, start, end, k):
        if (end - start + 1 < k):
            return
        if (k == 0):
            return
        i = start
        j = end
        mid = start + (end - start) // 2
        pivot = nums[mid]
        while (i <= j):
            while ((i <= j) and (nums[i] < pivot)):
                i += 1
            while ((i <= j) and (nums[j] > pivot)):
                j -= 1
            if (i <= j):
                nums[i], nums[j] = nums[j], nums[i]
                i += 1
                j -= 1

        delta_right = end - i + 1
        if delta_right >= k:
            self.quickselect(nums, i, end, k)
        else:
            self.quickselect(nums, start, j, k - delta_right)
```

👍 获赞 0 💬 添加评论





Jeffrey

更新于 6/9/2020, 7:04:27 AM

运用了JavaScript `sort()` 方法, 其中需要添加一个让nums数组排序的`sortNumber`方法。由于`sort()`返回的数组排序是从小到大的顺序, 所以返回一个数组长度-n。

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm\_source=code
 */
/**
 * @param n: An integer
 * @param nums: An array
 * @return: the Kth largest element
 */
function sortNumber(a,b)
{
    return a - b;
}
const kthLargestElement = function (n, nums) {
    nums.sort(sortNumber);
    return nums[nums.length-n];
}
```

 获赞 0 1 条评论**九章用户Y1KOB5**

更新于 6/9/2020, 7:04:24 AM

QuickSelect方法

最坏的情况是 $O(N^2)$, 一般情况是 $O(N)$.

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
class Solution {
    /**
     * @param k : description of k
     * @param nums : array of nums
     * @return: description of return
     */
    public int kthLargestElement(int k, int[] nums) {
        // write your code here
        return quickSelect(nums, 0, nums.length - 1, k - 1);
    }

    int quickSelect(int[] nums, int start, int end, int k) {
        if (start >= end) {
            return nums[start];
        }
        int pivot = nums[(end - start) / 2 + start];
        int l = start;
        int m = start;
        int r = end;
        while (m <= r) {
            if (nums[m] < pivot) {
                swap(nums, m, r);
                r--;
            } else if (nums[m] > pivot) {
                swap(nums, m, l);
                l++;
                m++;
            } else {
                m++;
            }
        }

        if (k < l) {
            return quickSelect(nums, start, l - 1, k);
        } else if (k >= m) {
            return quickSelect(nums, m, end, k);
        } else {
            return nums[k];
        }
    }

    void swap(int[] nums, int l, int r) {
        int temp = nums[l];
        nums[l] = nums[r];
        nums[r] = temp;
    }
};
```

👍 获赞 0

💬 添加评论



九章用户Y1KOB5

更新于 6/9/2020, 7:04:24 AM

QuickSelect方法

最坏的情况是 $O(N^2)$, 一般情况是 $O(N)$.

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到到的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ /
Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
class Solution {
    /**
     * @param k : description of k
     * @param nums : array of nums
     * @return: description of return
     */
    public int kthLargestElement(int k, int[] nums) {
        // write your code here
        return quickSelect(nums, 0, nums.length - 1, k - 1);
    }

    int quickSelect(int[] nums, int start, int end, int k) {
        if (start >= end) {
            return nums[start];
        }
        int pivot = nums[(end - start) / 2 + start];
        int l = start;
        int m = start;
        int r = end;
        while (m <= r) {
            if (nums[m] < pivot) {
                swap(nums, m, r);
                r--;
            } else if (nums[m] > pivot) {
                swap(nums, m, l);
                l++;
                m++;
            } else {
                m++;
            }
        }

        if (k < l) {
            return quickSelect(nums, start, l - 1, k);
        } else if (k >= m) {
            return quickSelect(nums, m, end, k);
        } else {
            return nums[k];
        }
    }

    void swap(int[] nums, int l, int r) {
        int temp = nums[l];
        nums[l] = nums[r];
        nums[r] = temp;
    }
};
```

👍 获赞 0 💬 添加评论



九章用户DQHFAN

更新于 6/9/2020, 7:04:24 AM

用了同样的quick select算法, 但是没有使用递归, 本质上也很简单, 就是调节start和end知道二者想交汇

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ /
Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm_source=code
 */
class Solution {
    /**
     * @param k : description of k
     * @param nums : array of nums
     * @return: description of return
     */
    public int kthLargestElement(int k, int[] nums) {
        // write your code here
        if (nums == null || nums.length == 0 || nums.length < k) {
            throw new IllegalArgumentException();
        }

        int targetInd = nums.length - k;
        int start = 0, end = nums.length - 1;
        while (start < end) {

            int s = start, e = end;
            int pivot = nums[(s + e) / 2];
            while (s <= e) {
                while (s <= e && nums[s] < pivot) {
                    s++;
                }
                while (s <= e && nums[e] > pivot) {
                    e--;
                }
                if (s <= e) {
                    swap(nums, s, e);
                    s++;
                    e--;
                }
            }
            if (s >= targetInd) {
                end = s - 1;
            } else {
                start = s;
            }
        }
        return nums[targetInd];
    }

    private void swap(int[] nums, int n1, int n2) {
        int tmp = nums[n1];
        nums[n1] = nums[n2];
        nums[n2] = tmp;
    }
};
```

👍 获赞 0

💬 添加评论



社会我喵哥

更新于 6/9/2020, 7:04:24 AM

Quick select 模板 Python 之前用Partition函数写过, 现在会超时。


```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm\_source=code
 */
class Solution:
    # @param k & A a integer and an array
    # @return ans a integer
    def kthLargestElement(self, k, A):
        if not A or k < 1 or k > len(A): return None
        return self.quickSelect(A, 0, len(A)-1, len(A)-k)

    def quickSelect(self, A, start, end, k):
        if start == end: return A[k]
        left, right = start, end
        pivot = A[(start+end)//2]
        while left <= right:
            while left <= right and A[left] < pivot: left += 1
            while left <= right and A[right] > pivot: right -= 1
            if left <= right:
                A[left], A[right] = A[right], A[left]
                left += 1
                right -= 1
        if k <= right: return self.quickSelect(A, start, right, k)
        if k >= left: return self.quickSelect(A, left, end, k)
        return A[k]
```

👍 获赞 0

💬 添加评论

**Leon**

更新于 6/9/2020, 7:04:22 AM

个人感觉quick sort算法不是那么直接。如果是全部排序之后, 提取第k个数字, 程序看着不那么舒服。。如果是直接sort出第k个数字, 坑点在如果k在右侧, 需要迭代的时候考虑k-i+1 相比而言用heap sort 更直接点, 先heapify之后, 再用klog(n)的时间把第k个数字取出来就可以了。ps.不知道是不是我对heap的理解有问题, 费时好多。。

```
/**
 * 本参考程序由九章算法用户提供。版权所有, 转发请注明出处。
 * - 九章算法致力于帮助更多中国人找到好的工作, 授课老师均来自硅谷和国内的一线大公司在职工程师。
 * - 现有的求职课程包括: 九章算法班 2020升级版, 算法强化班, 算法基础班, 北美算法面试高频题班, Java 高级工程师 P6+ 小班课, 面试软技能指导 - BQ / Resume / Project 2020版
 * - Design类课程包括: 系统设计 System Design, 面向对象设计 OOD
 * - 专题及项目类课程包括: 动态规划专题班, Big Data - Spark 项目实战, Django 开发项目课
 * - 更多详情请见官方网站: http://www.jiuzhang.com/?utm\_source=code
 */
class Solution:
    # @param k & A a integer and an array
    # @return ans a integer
    def kthLargestElement(self, k, A):
        self.size = len(A)
        self.nums = A
        for i in range(self.size-1,-1,-1):
            self.heapify(i)

        for i in range(len(A)-1,len(A)-k,-1):
            self.nums[0],self.nums[i] = self.nums[i],self.nums[0]
            self.size -= 1
            self.heapify(0)

        return self.nums[0]

    def heapify(self,i):
        l = i*2 + 1
        r = i*2 + 2
        largest = i
        if l<self.size and self.nums[l] > self.nums[largest]:
            largest = l
        if r<self.size and self.nums[r] > self.nums[largest]:
            largest = r
        if largest != i:
            self.nums[i],self.nums[largest] = self.nums[largest],self.nums[i]
            self.heapify(largest)
```

👍 获赞 0

💬 1 条评论

[加载更多题解](#)

进阶课程

视频+互动	直播+互动	直播+互动	互动课
<div>九章算法班 2021 版</div> <div>8周时间精通 57 个核心高频考点，9 招击破 FLAG、BATJ 算法面试。22....</div>	<div>系统架构设计 System Design 2021 版</div> <div>成为百万架构师必上。30 课时带你快速掌握18大系统架构设计知识点与面...</div>	<div>九章算法面试高频题冲刺班</div> <div>每期更新 15% 题目，考前押题，一举拿下FLAG & BATJ Offer</div>	<div>面向对象设计 OOD</div> <div>应届生及亚马逊面试必考，IT求职必备基础</div>