**实验报告**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **课程名称** | 《算法分析与设计》 | **实验日期** | 2021 年4月 19日 至2021 年 4月 19日 | | | | |
| **学生姓名** | 王海轩 | **所在班级** | 计算机195 | | | **学号** | 2019212212194 |
| **实验名称** | 在数组a 中找到第k 小的元素 | | | | | | |
| **实验地点** | 勤园13-208 | | | **同组人员** |  | | |

# 问题

在数组a 中找到第k 小的元素。

# 解析

抽取一个元素m，对于小于m 的元素划分为一个子集A，其余元素划分为另一个子集B。若A中的元素个数+1k，则m为解。

若A中的元素个数>k，则递归在A中求第k大元素。否则在B中递归求第k-[AI大的元素。

对于有相同元素的情况需要特别考虑，因为可能会出现A集合为空的情况，此时需要判断当前取的x的崴量是否超过k，若超过k则x为答案，否则在另一个子集C(子集C为大于x的元素的集合)中求解第k-count(x)大的元素。

对于抽取的元素m，应当抽取中位数，可采用对所有元素划分为多组，每组5个元素，对每组排序后取每组中位数，递归至只剩1个元素。

# 设计

#include <stdio.h>

#include <stdlib.h>

#include<math.h>

#define NUM 50 //最大容量

int s1[NUM], s2[NUM],mid[15];

int cmp(const void \* a, const void \* b) {

return \*(int \*)a - \*(int \*)b;

}

int select(int num[], int p, int r, int k) {

//当前数组长度

int len = r - p + 1;

//判断长度是否小于5，小于5直接排序求第k小的数

if (len <= 5) {

qsort(num, len, sizeof(int), cmp);

return num[k - 1];

}

else {

//将数组分为group组，每组5个数

int group = len / 5;

int i;

//将数组剩余的数赋值给m

int m = len - group \* 5;

//每个group组进行快速排序

for (i = 0; i < group; i++) {

qsort(num + i \* 5, 5, sizeof(int), cmp);

//排序后，把每个group组的中间值放在mid中

mid[i] = num[i \* 5 + 2];

}

//若分完group组有剩余，将剩余的数进行排序

if (m > 0) {

qsort(num + group \* 5, m, sizeof(int), cmp);

//将中间值赋值给mid

mid[group] = num[group \* 5 + m / 2];

//group+1

group++;

}

//对mid数组排序，找出中间值赋值给mm

qsort(mid, group, sizeof(int), cmp);

int mm = mid[group / 2];

//s1的长度为len\_s1,s2的长度为len\_s2

int len\_s1 = 0, len\_s2 = 0;

//用mm(整个数组最中间的值)把数组分为s1和s2

for (int i = p-1; i < r; ++i) {

//比mm小的数放在s1，比mm大的数放在s2

if (num[i] < mm) {

s1[len\_s1++] = num[i];

}

else {

s2[len\_s2++] = num[i];

}

}

int f = 0;

//如果k的值==len\_s1+1,说明mm就是第k小的数

if (k == len\_s1 + 1) {

f = mm;

}

else if (k <= len\_s1) {

//如果k<=len\_s1,说明第k小的数在s1,递归调用select函数

f = select(s1, 1, len\_s1, k);

}

else {

//如果k>len\_s1,说明第k小的数在s2,递归调用select函数

f = select(s2, 1, len\_s2, k - len\_s1);

}

return f;

}

}

int main() {

//输入示例

int n[NUM] = { 13,12,11,14,15,7,9,14,16,18,2,4,6,8,10,1,3,5,7,9,2,3};

printf("%d\n", select(n, 1, 22, 10));

return 0;

}

# 分析

时间复杂度 O(n)

# **源码**

[github源码地址]

https://github.com/whx116/lab6.git