HOMEWORK REPORT

个人信息

姓名	学号
吴宇杰	19215028

题目内容

给定一个数组和k，线性时间内找出第k大的数

解题思路

通过一个粗略估算中位数的算法将原数组分割为两半
通过k与中位数做比较直接在其中一半进行搜索,实现分治

实现代码

```
#include <iostream>
#include <vector>
#include <algorithm>
#include <cassert>
#include "array_utils.h"

using namespace std;

const int GROUP_COUNT = 5;


int find_middle (vector<int> target) {
    // return the middle number in target
    if (target.size()/GROUP_COUNT <= 1) {
        sort(target.begin(), target.end());
        int middle = target.size() % 2 ? target.size() /2 + 1 : target.size() /2;
        return target[middle-1];
    }

    vector<int> child;

    for (int i = 0; i<target.size()/GROUP_COUNT;i++)
    {
        int end = (i+1)*GROUP_COUNT >= target.size()-1 ? target.size() - 1 : (i+1) *
        sort(target.begin()+i*GROUP_COUNT, target.begin() + end);
        child.push_back(*(target.begin()+i*GROUP_COUNT+GROUP_COUNT/2));
```

```
    }

    return find_middle(child);
}

int liner_search (vector<int> target, int k)
{
    assert(k > 0);
    int middle = find_middle(target);
    int pre=0, i=0, post = target.size() - 1;

    while(i <= post) {
        if (target[i] < middle) {
            swap(target[i], target[pre]);
            i++;
            pre++;
        } else if (target[i] == middle) {
            i++;
        } else if (target[i] > middle) {
            swap(target[i], target[post]);
            post--;
        }
    }

    // k begin from 1, and index begin from 0
    if (k-1 < pre) {
        vector<int> tmp(target.begin(), target.begin()+pre);
        return liner_search(tmp, k);
    } else if (k-1 > post) {
        vector<int> tmp(target.begin()+post+1, target.end());
        return liner_search(tmp, k-post-1);
    } else {
        return target[pre];
    }
}

int main () {
    const int LIST_LENGTH = 200;

    const int k = LIST_LENGTH /2;

    vector<int> test_list = ArrayUtils::genrateRandomVector(LIST_LENGTH, 0, LIST_LEN

    ArrayUtils::printVector(test_list);

    cout << liner_search(test_list, k) << endl;

    sort(test_list.begin(), test_list.end());
    ArrayUtils::printVector(test_list);
    cout << test_list[k-1];

    return 0;
}
```

测试样例

```
输入:
array: 183 86 177 115 193 135 186 92 49 21 162 27 90 59 163 126 140 26 172 136 11 16
k: 100
输出:
105
```

总结

> 通过一个分治理预估中位数的算法,将原问题分割为一个约一半长度的子问题

预估中位数算法时间复杂度为:

$$T(n) = kT(n/k)$$

> 可知时间复杂度是线性时间复杂度

寻找第K大数的时间复杂度为:

$$T(n) = T(n/2) + O(n)$$

> 可知时间复杂度也是线性时间复杂度