# **[STL中nth\_element的用法](https://www.cnblogs.com/xenny/p/9424894.html)**

nth\_element函数原型有四个，详细我就不一一累赘了，我们就用最普通的用法寻找第k位置的元素。

函数用法为：nth\_element(first，kth，end)。

first，last 第一个和最后一个迭代器，也可以直接用数组的位置。  
kth，要定位的第k个元素，能对它进行随机访问.

将第k\_th元素放到它该放的位置上，左边元素都小于等于它，右边元素都大于等于它.

例如：

#include<iostream>

#include<algorithm>

#include<vector>

using namespace std;

int main(){

vector<int> a(9);

for(int i = 0; i < 9; i++)

a[i] = i+1;

random\_shuffle(a.begin(),a.end());

for(int i = 0; i < 9; i++)

cout << a[i] << " ";

cout << endl;

nth\_element(a.begin(),a.begin()+4,a.end());

cout << \*(a.begin()+4) << endl;

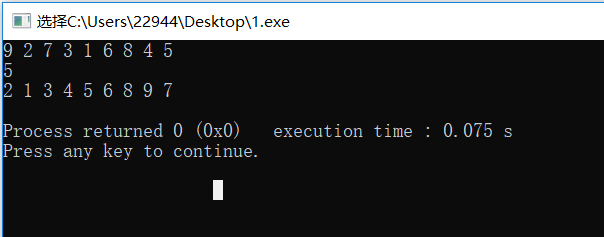
for(int i = 0; i < 9; i++)

cout << a[i] << " ";

cout << endl;

}

结果为：



可以发现函数只是把kth的元素放在了正确位置，对其他元素并没有排序，所以可以利用这个函数快速定位第k个元素，当然，这个函数也支持你直接写比较函数，此处不再累赘因为ACM中基本不会用到。

那么为什么要选择这个函数呢，这就和复杂度有关了，nth\_element的空间复杂度为O(1)，在数据量大的时候时间复杂度为O(n)，数据少的情况最坏为O(n^2)，因为函数原理是随机访问，但实际运用过程中，基本都会是O(n)的时间复杂度。所以说是非常迅速的。