/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

快速排序

O(NlogN)

代码实例：https://www.zheshigelianjie.com/

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <algorithm>

using namespace std;

void quicksort(int l, int r, int v[])

{

if (l >= r)

return;

/\* 确定划分点 \*/

int i = l, j = r + 1;

while (1)

{

while (v[l] > v[++i]);

while (v[l] < v[--j]);

if (i >= j)

break;

swap(v[i], v[j]);

}

swap(v[l], v[j]);

/\* 分治子区间 \*/

quicksort(l, j - 1, v);

quicksort(j + 1, r, v);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

快速排序(random)

O(NlogN)

附random\_shuffle方法姿势

测试代码：https://www.zheshigelianjie.com/

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <algorithm>

#include <ctime> // time()函数需求

#include <cstdlib> // srand()函数需求

void quicksort(int l, int r, int v[])

{

if (l >= r)

return;

/\* 打乱区间元素分布 \*/

random\_shuffle(v + l, v + r + 1);

/\* 确定划分点 \*/

int i = l, j = r + 1;

while (1)

{

while (v[l] > v[++i]);

while (v[l] < v[--j]);

if (i >= j)

break;

swap(v[i], v[j]);

}

swap(v[l], v[j]);

/\* 分治子区间 \*/

quicksort(l, j - 1, v);

quicksort(j + 1, r, v);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

使[first, last)内元素排列打乱, 不提供gen时默认为rand()

建议使用前用srand(time(0))进行初始化

调用方法： random\_shuffle(first, last); //不提供随机数生成器

random\_shuffle(first, last, myrandom); //提供随机数生成器

函数源码~~(?)~~:

template <class Iter, class Generator>

void random\_shuffle (Iter first, Iter last, Generator& gen){

iterator\_traits<Iter>::difference\_type i, n;

n = (last-first);

for (i=n-1; i>0; --i){

swap(first[i], first[gen(i+1)]);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/