**快速排序**

**概念：**

通过一趟排序将要排序的数据分割成独立的两部分，其中一部分的所有数据都比另外一部分所有数据都要小，然后按此方法对这两部分数据分别进行快速排序，整个排序过程可以递归进行，以此达到整个数据变成有序序列。

**原理：**

1、先从数列中取出一个数作为基准数。

2、分区过程中，将比这个数大的数全放到他的右边，小于或等于它的数全放到它的左边。

3、再对左右区间重复第二步，直到各区间只有一个数。

**代码一：**

#include <iostream>

using namespace std;

template<typename T>

void quick\_sort(T value[],int low,int high)

{

if (low>high)

return;

int i =low,j=high;

T temp = value[low];

while(i<j)

{

while(i<j&&value[j]<temp)

j--;

value[i]=value[j];

while(i<j&&value[i]>=temp)

i++;

value[j]=value[i];

}

value[i] = temp;

quick\_sort(value,low,i-1);

quick\_sort(value,i+1,high);

}

int \_tmain(int argc, \_TCHAR\* argv[])

{

int a[]={3,4,5,19,20,51,33,88,1,0,8};

int n=(int)sizeof(a)/sizeof(\*a);

quick\_sort(a,0,n-1);

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

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

cout<<endl;

return 0;

}

**代码二：**

template<typename T>

void quick\_sort(T a[],int low,int high)

{

if (low>high)

return;

T temp = a[low];

int i=low,j=high;

while(i<j)

{

while(i<j&&a[j]>temp)

j--;

while(i<j&&a[i]<=temp)

i++;

if (i<j)

{

T value = a[i];

a[i] = a[j];

a[j] = value;

}

}

a[low] = a[i];

a[i] = temp;

quick\_sort(a,low,i-1);

quick\_sort(a,i+1,high);

}

**算法分析：**

快速排序的优点：对包含n个数的输入数组，最坏情况运行时间为O（n^2），虽然这个最坏情况运行时间比较差，但是快速排序通常用于排序的最佳的实用选择，这是因为其平均性能相当好。