  

void QuickSort2(int a[2000005],int low,int high)//交换a b c可以看出Quicksort2比Quicksort更好

{

int i = low;

int j = high;

int key = a[low];

if (low > high) //如果low > high说明排序结束了

{

return ;

}

while(i<j)

{

while (i<j&&a[j]>key)j--;//从右向左扫描，找到<=key的元素

a[i]=a[j];

while (i<j&&a[i]<=key)i++;//从左向右扫描，找到>key的元素

a[j]=a[i];

}

a[i]=key;

QuickSort2(a, i, low-1); //用同样的方式对分出来的左边的部分进行同上的做法

QuickSort2(a, low+1, j); //用同样的方式对分出来的右边的部分进行同上的做法

}

void merge(int a[],int b[],int low,int high) //传入数组的两端

{

if(low>=hith) return ; //设置结束的标志很重要

int len=high-low,mid=low+(len>>1); //计算长度，有优先级，要先打括号

int start1=low,end1=mid; //保留数组的两端

int start2=mid+1,end2=high;

int k=low; //保留后数组现在存储到了哪里

merge(a,b,start1,end1); //先对左边进行归并

merge(a,b,start2,end2); //再对右边进行归并

while(start1<=end1&&start2<=end2)

b[k++]=a[start1]>a[start2]?a[start1++]:a[start2++]; //把左右两个已经有序的数组进行归并

while(start1<=end1)

b[k++]=a[start1++];

while(start2<=end2)

b[k++]=a[start2++];

for(k=low;k<=high;k++)

a[k]=b[k]; //最后把数组拷贝过来

}

void mergesort(int a[],int n)

{

int b[1000];

merge(a,b,0,n-1); //有数字的进行运算

}