思想:

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的所有数据都要小，然后再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](http://baike.baidu.com/view/96473.htm)进行，以此达到整个数据变成有序序列。

设要排序的[数组](http://baike.baidu.com/view/209670.htm)是A[0]……A[N-1]，首先任意选取一个数据（通常选用数组的第一个数）作为关键数据，然后将所有比它小的数都放到它前面，所有比它大的数都放到它后面，这个过程称为一趟快速排序。值得注意的是，快速排序不是一种稳定的[排序算法](http://baike.baidu.com/view/297739.htm)，也就是说，多个相同的值的相对位置也许会在算法结束时产生变动。

一趟快速排序的算法是：

1）设置两个[变量](http://baike.baidu.com/view/296689.htm)i、j，[排序](http://baike.baidu.com/view/58783.htm)开始的时候：i=0，j=N-1；

2）以第一个[数组](http://baike.baidu.com/view/209670.htm)元素作为关键数据，赋值给**key**，即**key**=A[0]；

3）从j开始向前搜索，即由后开始向前搜索(j--)，找到第一个小于**key**的值A[j]，将A[j]和A[i]互换；

4）从i开始向后搜索，即由前开始向后搜索(i++)，找到第一个大于**key**的A[i]，将A[i]和A[j]互换；

5）重复第3、4步，直到i=j； (3,4步中，没找到符合条件的值，即3中A[j]不小于**key**,4中A[i]不大于**key**的时候改变j、i的值，使得j=j-1，i=i+1，直至找到为止。找到符合条件的值，进行交换的时候i， j指针位置不变。另外，i==j这一过程一定正好是i+或j-完成的时候，此时令循环结束）。

代码：

//快速排序算法

function doSort(a, s, e) {

if (s < e) {

var pos = partition(a, s, e);

doSort(a, s, pos - 1);

doSort(a, pos + 1, e);

}

}

function partition(a, st, en) {

var s = st;

var e = en + 1;

var temp = a[s];

while (1) {

while (a[++s] < temp) {

}

while (a[--e] > temp) {

}

if (s > e) {

break;

}

var tem = a[s];

a[s] = a[e];

a[e] = tem;

}

a[st] = a[e];

a[e] = temp;

return e;

}

//向数组附加快速排序算法

Array.prototype.quickSort = function () {

doSort(this, 0, this.length - 1);

};

资源链接：<http://liudaoru.iteye.com/blog/156496>