排序算法：



**冒泡排序**：

for(var i=0;i<arr.length-1;i++){

for(var j=0;j<arr.length-1-i;j++){

if(arr[j+1]<arr[j]]){

var temp=arr[j];

arr[j]=arr[j+1];

arr[j+1]=temp;

}

}

}

平均时间复杂度：O(n^2) 稳定

**选择排序**：每次从未排序的序列中选出最大或者最小放到已排序的序列中

for(var i=0;i<arr.length-1;i++){

var minIndex=i;

for(var j=i+1;j<arr.length;j++){

if(arr[j]<arr[minIndex]){

minIndex=j;

}

}

var temp=arr[minIndex];

arr[minIndex]=arr[i];

arr[i]=temp;

}

平均时间复杂度：O(n^2) 不稳定

**插入排序**：每次待插入排序的数按照大小插入到前面已排序好的序列中

for(var i=1;i<arr.length;i++){

var j=i-1;

while(j>=0&&arr[j]>arr[i]){

arr[j+1]=arr[j];

j--;

}

arr[j+1]=arr[i];

}

平均时间复杂度：O(n^2) 稳定

**归并排序**：拆分，有序，归并

平均时间复杂度，最佳和最差情况都是 ****O(nlgn)****

function mergeSort(arr,len){

if(len>1){

var half=Number.parseInt(len/2);

var arr1=[],arr2=[];

for(let i=0;i<half;i++){

arr1[i]=arr[i]

}

for(let i=0,j=half;j<len;i++,j++){

arr2[i]=arr[j]

}

mergeSort(arr1,arr1.length);

mergeSort(arr2,arr2.length);

return merge(arr1,arr1.length,arr2,arr2.length,arr);

}

}

function merge(arr1,length1,arr2,length2,arr){

var i=k=j=0;

while(i<length1&&j<length2){

if(arr1[i]<arr2[j]){

arr[k++]=arr1[i++];

}else{  
 arr[k++]=arr2[j++];

}

}

while(i<length1){

arr[k++]=arr1[i++];

}

while(j<length2){

arr[k++]=arr2[j++]

}

return arr;

}

平均时间复杂度，最佳和最差情况都是 ****O(nlgn) 稳定****

****快速排序：**取一个值，小于这个值分为一组，大于这个值的分为一组，以此类推，直到不能分为止即排好序（递归）**

function quickSort(left,right){

if(left>right) return;

var i=left;

var j=right;

var temp=arr[left];

while(i!=j){

while(arr[j]>=temp&&j>i){

j--

}

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

i++

}

if(j>i){

var t=arr[i];

arr[i]=arr[j];

arr[j]=t

}

}

arr[left]=arr[i];

arr[i]=temp;

quickSort(left,i-1);

quickSort(i+1,right)

}

快速排序的最差时间复杂度和冒泡排序是一样的都是O(N2)，它的平均时间复杂度为O(NlogN) 不稳定

**堆排序：**

大根堆和小根堆：已知某个数的索引=》父节点索引：(i-1)/2(去小数字),左孩子索引：2\*i+1,右孩子索引：2\*i+2 平均时间复杂度：O(nlogn) 不稳定

**希尔排序：**

function shellSort(arr){

var len=arr.length;

for(let gap=Math.floor(len/2);gap>0;gap=Math.floor(len/2)){

for(let i=gap;i<len;i++){

let j=i;

let current=arr[i];

while(j-gap>=0&&current>arr[j-gap]){

arr[j]=arr[j-gap];

j=j-gap;

}

arr[j]=current

}

}

}