# 算法

## 快速排序

```
function quickSort(arr) {
  // 去除特殊情况
  if(arr.length <= 1){
    return arr
  }
  // 找基准pivot
  var pivotIndex = Math.floor(arr.length/2)
  var pivot = arr.splice(pivotIndex, 1)[0]
  var left = []
  var right = []
  // 分治
  for (var i=0; i<arr.length; i++) {
    if(arr[i] < pivot){
      left.push(arr[i])
    }else{
      right.push(arr[i])
    }
  }
  // 递归排序左右子数组
  return quickSort(left).concat([pivot], quickSort(right))
}
```

## 插入排序&希尔排序

```
// 插入排序
function insertSort(arr) {
  for(var i =1;i<arr.length;i++){
    var key = arr[i]
    var j = i
    while(j>0&&arr[j-1]>key){
      arr[j] = arr[j-1]
      j--
    }
    arr[j] = key
  }
  return arr
}
// 希尔排序
function hillSort(arr) {
  var gap = Math.floor(arr.length/2)
  while(gap!==0){
    for(var i =gap;i<arr.length;i++){
```

```
      var temp = arr[i]
      var j = i -gap
      for(;j>=0&&temp<arr[j];j-=gap){
        arr[j+gap] = arr[j]
      }
      arr[j+gap] = temp
    }
    gap = Math.floor(gap/2)
  }
  return arr
}
```

## 平铺数组

```
function deepFlatten(arr){
  return [].concat(...arr.map((item)=>{
    return !Array.isArray(item) ? item : deepFlatten(item)
  }))
}
```