归并排序

# 归并思想分析

需要复制一个数组，因此空间复杂度为O(n);

递归实现。

/\*\*\*\*\*\*\*归并排序：使用固定的辅助数组\*\*\*\*\*\*\*\*\*\*\*\*\*/

public void **mergeSort**(int[] nums){//经过测试，无论nums数组是奇数还是偶数，正确结果都存在nums中，而copyNums差一次

int[] copyNums = Arrays.copyOf(nums,nums.length);

**mergeSort**(nums,copyNums,0,nums.length-1);

}

public void **mergeSort**(int[] nums,int[] copyNums,int low,int high){

if(nums == null|| nums.length <= 1||low >= high) return ;

int mid = (low + high)>>1;

**mergeSort(copyNums,nums,low,mid);//交替使用nums和copyNums**

**mergeSort(copyNums,nums,mid+1,high);**

**//将最终排序结果拷贝至nums中，因此最终正确排序就是nums，而copyNums不是**

**mergeTowSubArray(copyNums,nums,low,mid,high);**

}

/\*\*

\* 合并两个排序子数组为一个排序的子数组

\* （归并思想，与合并两个有序链表是一致的）

\*/

public void **mergeTowSubArray**(int[] nums,int[] copyNums,int low,int mid,int high){

int i = mid,j = high,indexCopy = high;

while(i >= low&&j >= mid+1){//其中之一遍历结束，则结束while

copyNums[indexCopy--] = nums[i]>nums[j]?nums[i--]:nums[j--];

}

while(i>=low) copyNums[indexCopy--] = nums[i--];

while(j>=mid+1) copyNums[indexCopy--] = nums[j--];

}