```java
//快速排序
public int[] QuickSort(int []nums,int left,int right){
    if(left<right){
     int middle = getMiddle(nums,left,right);
     QuickSort(nums,left,middle-1);
     QuickSort(nums,middle+1,right);
    }
    return nums;
}
public int getMiddle(int []nums,int left,int right){
    int tmp = nums[left];//基准的初始位置为子数组的最左元素
    while(left<right){
        while(left<right&&nums[right]>=tmp){
            right--;
        }
        nums[left] = nums[right];
        while(left<right&&nums[left]<=tmp){
            left++;
        }
        nums[right] = nums[left];
    }
    nums[left] = tmp;
    return left;
}
```

```java
//归并排序
public int[] sort(int []nums,int left,int right){
    int mid = left+(right-left)/2;
    if(left<right){
        sort(nums,left,mid);
        sort(nums,mid+1,right);
        merge(nums,left,mid,right);
    }
    return nums;
}
public int[] merge(int []nums,int left,int mid,int right){
    int []tmp = new int[right-left+1];
    int i = left;
    int j = mid+1;
    int k = 0;
    while(i<=mid&&j<=right){
        if(nums[i]<nums[j])tmp[k++] = nums[i++];
        else tmp[k++] = nums[j++];
    }
    while(i<=mid){
```

```
        tmp[k++] = nums[i++];
    }
    while(j<=right){
        tmp[k++] = nums[j++];
    }
    for(int index = 0;index<tmp.length;index++){
        nums[left+index] = tmp[index];
    }
    return nums;
}
```

```
//希尔排序
    public int[]method4(int []nums){
        int length = nums.length;
        int h  = 1;
        while(h<length/3){
            h = h*3 +1;
        }
        while(h>=1){
            for(int i =h;i<length;i++){
                for(int j=i;j>=h&&nums[j]<nums[j-h];j-=h){
                     swap(nums,j,j-h);
                }
            }
            h/=3;
        }
        return nums;
    }
```