### JAVA第五阶段—DAY07-JAVA案例

1. 有int数组{ 4, 2, 8, 9, 5, 7, 6, 1, 3 },请使用冒泡排序对其进行排序。

*//冒泡排序*public class BubbleSort {  
  
 public static void sort(int[] array) {  
 *// 这里for循环表示总共需要比较多少轮* for (int i = 1; i < array.length; i++) {  
 *// 这里for循环表示每轮比较参与的元素下标  
 // 对当前无序区间array[0 length-i]进行排序  
 // j的范围很关键，这个范围是在逐步缩小的,因为每轮比较都会将最大的放在右边* for (int j = 0; j < array.length - i; j++) {  
 if (array[j] > array[j + 1]) {  
 int temp = array[j];  
 array[j] = array[j + 1];  
 array[j + 1] = temp;  
 }  
 }  
 }  
 }  
  
 public static void main(String[] args) {  
 int[] array = { 4, 2, 8, 9, 5, 7, 6, 1, 3 };  
 *// 未排序数组顺序为* System.out.println("未排序数组顺序为：");  
 System.out.println(Arrays.toString(array));  
  
 sort(array);  
 System.out.println("经过冒泡排序后的数组顺序为：");  
 System.out.println(Arrays.toString(array));  
 }  
}

1. 有序数组{ 4, 2, 8, 9, 5, 7, 6, 1, 3}，请使用选择排序进行排序。

*// 选择排序*public class ChoiceSort {  
  
 public static void sort(int[] array) {  
 *// 总共要经过N-1轮比较* for (int i = 0; i < array.length - 1; i++) {  
 int min = i;  
 *// 每轮需要比较的次数* for (int j = i + 1; j < array.length; j++) {  
 if (array[j] < array[min]) {  
 min = j;*// 记录目前能找到的最小值元素的下标* }  
 }  
 *// 将找到的最小值和i位置所在的值进行交换* if (i != min) {  
 int temp = array[i];  
 array[i] = array[min];  
 array[min] = temp;  
 }  
  
 }  
 }  
  
 public static void main(String[] args) {  
 int[] array = { 4, 2, 8, 9, 5, 7, 6, 1, 3};  
 *// 未排序数组顺序为* System.out.println("未排序数组顺序为：");  
 System.out.println(Arrays.toString(array));  
  
 sort(array);  
 System.out.println("经过选择排序后的数组顺序为：");  
 System.out.println(Arrays.toString(array));  
 }  
}

1. 有数组{ 4, 2, 8, 1, 3}，请使用插入排序进行排序。

*//插入排序*public class InsertSort {  
  
 public static int[] sort(int[] array) {  
 *// 从下标为1的元素开始选择合适的位置插入，因为下标为0的只有一个元素，默认是有序的* for (int i = 1; i < array.length; i++) {  
 int tmp = array[i];*// 记录要插入的数据* int j = i;  
 while (j > 0 && array[j - 1]>tmp) {  
 array[j] = array[j - 1];  
 *// 向后挪动* j--;  
 }  
 array[j] = tmp;*// 存在比其小的数，插入* }  
 return array;  
 }  
  
 public static void main(String[] args) {  
 int[] array = { 4, 2, 8, 1, 3};  
 *// 未排序数组顺序为* System.out.println("未排序数组顺序为：");  
 System.out.println(Arrays.toString(array));  
  
 sort(array);  
  
 System.out.println("经过插入排序后的数组顺序为：");  
 System.out.println(Arrays.toString(array));  
 }  
  
}

1. 有数组{ 4, 2, 8, 9, 5, 7, 6, 1, 3}，请使用快速排序对其进行排序。

*//快速排序***public class** QuickSort {  
  
 **public static void** main(String[] args) {  
  
 **int**[] array = { 4, 2, 8, 9, 5, 7, 6, 1, 3};  
 quickSort(array, 0, array.length - 1);  
 System.out.println(Arrays.toString(array));  
  
 }  
  
 **static void** quickSort(**int** s[], **int** left, **int** right) {  
 **if** (left < right) {  
 **int** i = left, j = right, x = s[left];  
 **while** (i < j) {  
 **while** (i < j && s[j] >= x) { *// 从右向左找第一个小于x的数* j--;  
 }  
 **while** (i < j && s[i] <= x) {*// 从左向右找第一个大于等于x的数* i++;  
 }  
 *//将找到的第一个小于x跟大于x的数据进行交换* **int** temp = s[j];  
 s[j] = s[i];  
 s[i] = temp ;  
 }  
*// 把目标交换到合适位置，保证前面的所有数都小于它，后面的数都大于它* **int** temp = s[i] ;  
 s[i] = s[left];  
 s[left] = temp ;  
  
 quickSort(s, left, i - 1); *// 递归调用* quickSort(s, i + 1, right);  
 }  
 }  
  
}

1. 有序数组{1,2,3,4,5,6,7,8,9,10}，请使用2分查找目标元素7，找到返回此元素的索引，没找到返回-1.

public class BinarySearch {  
  
 public static void main(String[] args) {  
 int [] arr = {1,2,3,4,5,6,7,8,9,10};  
 int result = binarySearch(arr,7);  
 System.out.println(result);  
  
 *// 在数组中查找目标为4的索引位置* System.out.println(Arrays.binarySearch(arr, 4));  
  
 }  
  
 public static int binarySearch(int [] srcArray, int des) {  
 *//定义初始最小、最大索引* int start = 0;  
 int end = srcArray.length - 1;  
 *//确保不会出现重复查找，越界* while (start <= end) {  
 *//计算出中间索引值* int middle = (end + start)>>>1 ;*//防止溢出* if (des == srcArray[middle]) {  
 return middle;  
 *//判断下限* } else if (des < srcArray[middle]) {  
 end = middle - 1;  
 *//判断上限* } else {  
 start = middle + 1;  
 }  
 }  
 *//若没有，则返回-1* return -1;  
 }  
}