

Java基础(6)¹

✧ 数组元素反转

- 1 需求：某个数组有5个数据：10,20,30,40,50，请将这个数组中的数据进行反转。
- 2 [10, 20, 30, 40, 50] 反转后 [50, 40, 30, 20, 10]

分析

- 1 1.每次交换，需要有左右两边的两个索引，我们可以用i和j表示刚开始i=0, j=数组长度-1;
- 2 2.每次让i和j索引位置的两个元素互换位置arr[i]和arr[j]互换位置
- 3 3.每次还完位置之后，让i往右移动一位，让j往前移动一位

```

1 public static void main(String[] args) {
2     // 现有一个 int 数组，数组中有十个元素。将数组反转后输出。
3     int[] arr = new int[]{9, 1, 3, 4, 54, 56, 23, 22, 20, 43};
4     // for (int i = arr.length - 1; i ≥ 0; i--) {
5         // System.out.println(arr[i]);
6     // }
7     for (int i = 0, j = arr.length - 1; i < arr.length / 2; i++, j--
- ) {
8         // int temp = arr[i];
9         // arr[i] = arr[j];
10        // arr[j] = temp;
11        arr[i] = arr[i] ^ arr[j]; // 1100 0000 1100 1100 0000
12        arr[j] = arr[i] ^ arr[j]; // arr[i] ^ arr[j] ^ arr[j] =
13        arr[i] ^ 0 = arr[i]
14        arr[i] = arr[i] ^ arr[j]; // arr[i] ^ arr[j] ^ arr[i] = 0
15        ^ arr[j] = arr[j]
16    }
17    for (int a : arr) {
18        System.out.println(a);
19    }
19 }
```

✧ 排序

排序算法 有多种，常用的排序算法有冒泡排序、插入排序、选择排序、快速排序、堆排序、归并排序、希尔排序、二叉树排序、计数排序等。

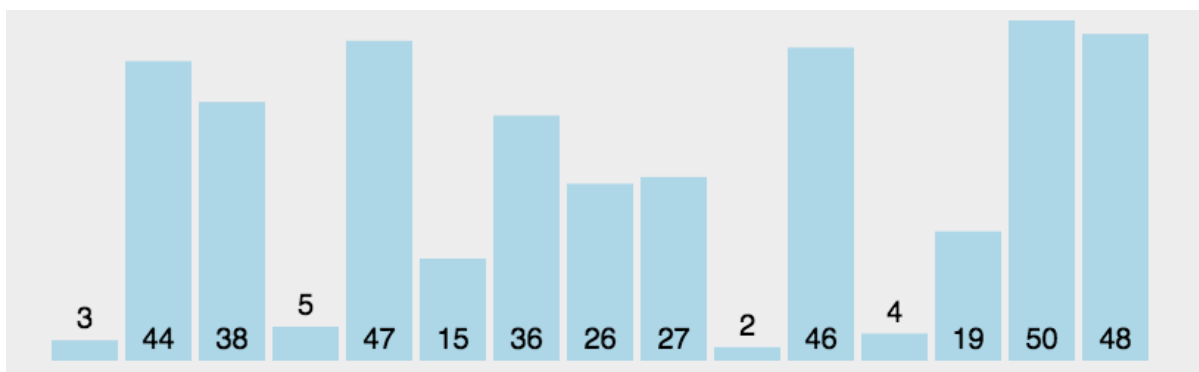
选择排序

表现最稳定的排序算法之一，因为无论什么数据进去都是 $O(n^2)$ 的时间复杂度，所以用到它的时候，数据规模越小越好。唯一的好处可能就是不占用额外的内存空间。

i

选择排序(Selection-sort)是一种简单直观的排序算法。它的工作原理：首先在未排序序列中找到最小（大）元素，存放到排序序列的起始位置，然后，再从剩余未排序元素中继续寻找最小（大）元素，然后放到已排序序列的末尾。以此类推，直到所有元素均排序完毕。

- 初始状态：无序区为 $R[1..n]$ ，有序区为空；
- 第 i 趟 排序 ($i=1,2,3...n-1$) 开始时，当前有序区和无序区分别为 $R[1..i-1]$ 和 $R(i..n)$ 。该趟排序从当前无序区中-选出关键字最小的记录 $R[k]$ ，将它与无序区的第1个记录 R 交换，使 $R[1..i]$ 和 $R[i+1..n]$ 分别变为记录个数增加1个的新有序区和记录个数减少1个的新无序区；
- $n-1$ 趟结束，数组有序化了。



✧ 插入排序

插入排序的算法描述是一种简单直观的排序算法。它的工作原理是通过构建有序序列，对于未排序数据，在已排序序列中从后向前扫描，找到相应位置并插入。插入排序在实现上，需要反复把已排序元素逐步向后挪位，为最新元素提供插入空间。算法描述：

- 从第一个元素开始，该元素可以认为已经被排序
- 取出下一个元素，在已经排序的元素序列中从后向前扫描
- 如果该元素(已排序)大于新元素，将该元素移到下一位置
- 重复步骤3，直到找到已排序的元素小于或者等于新元素的位置
- 将新元素插入到该位置后;0
- 重复步骤2~5。

