算法与数据结构体系课程

liuyubobobo

基于比较的排序算法总结

liuyubobobo

基于比较的排序算法总结

选择排序法

插入排序法

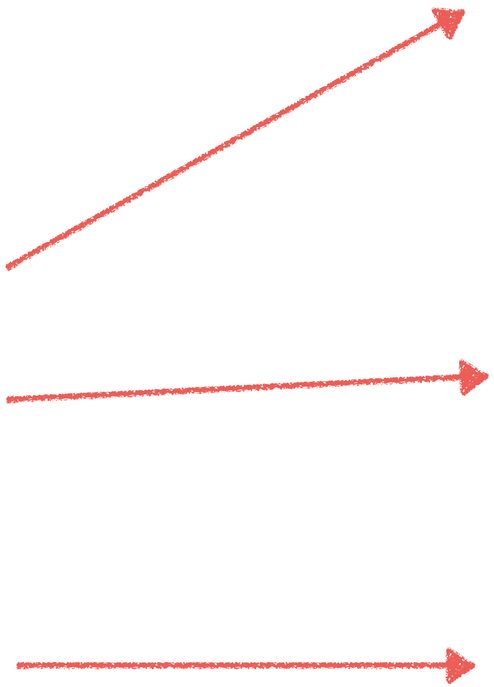
冒泡排序法

归并排序法

快速排序法

堆排序法

希尔排序法



自顶向下的归并排序法

自底向上的归并排序法

单路快速排序法

双路快速排序法

三路快速排序法

不同的步长序列

基于比较的排序算法总结

选择排序法

插入排序法

冒泡排序法

归并排序法

快速排序法

堆排序法

希尔排序法

自顶向下的归并排序法

自底向上的归并排序法

单路快速排序法

双路快速排序法

三路快速排序法

不同的步长序列

基于比较的排序算法总结

	时间
选择排序法	O(n^2)
插入排序法	O(n^2)
冒泡排序法	O(n^2)
归并排序法	O(nlogn)
快速排序法	O(nlogn)*
堆排序法	O(nlogn)
希尔排序法	O(nlogn) - O(n^2)

基于比较的排序算法总结

	时间	空间
选择排序法	O(n^2)	O(1)
插入排序法	O(n^2)	O(1)
冒泡排序法	O(n^2)	O(1)
归并排序法	O(nlogn)	O(n)
快速排序法	O(nlogn)*	O(1)
堆排序法	O(nlogn)	O(1)
希尔排序法	O(nlogn) - O(n^2)	O(1)

基于比较的排序算法总结

	时间	空间	特殊数据
选择排序法	O(n^2)	O(1)	
插入排序法	O(n^2)	O(1)	完全有序数组，时间 O(n)
冒泡排序法	O(n^2)	O(1)	完全有序数组，时间 O(n)
归并排序法	O(nlogn)	O(n)	完全有序数组，时间 O(n)
快速排序法	O(nlogn)*	O(1)	含有相同元素数组，三路快排时间 O(n)
堆排序法	O(nlogn)	O(1)	
希尔排序法	O(nlogn) - O(n^2)	O(1)	

基于比较的排序算法总结

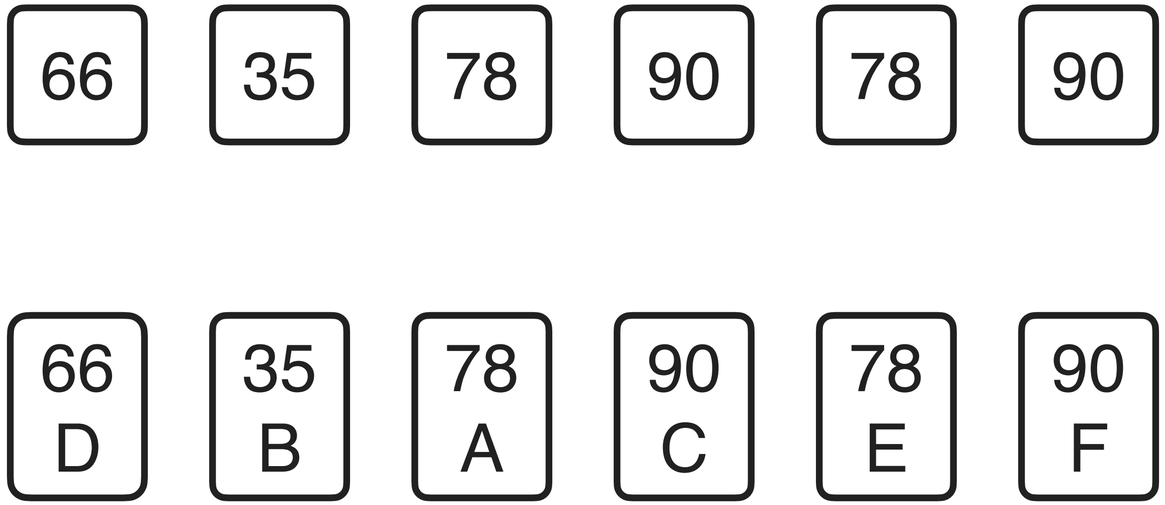
	时间	空间	其他
选择排序法	O(n^2)	O(1)	
插入排序法	O(n^2)	O(1)	
冒泡排序法	O(n^2)	O(1)	
归并排序法	O(nlogn)	O(n)	O(nlogn) 求解数组中逆序对个数
快速排序法	O(nlogn)*	O(1)	O(n) 求解 selectK, topK 问题
堆排序法	O(nlogn)	O(1)	堆；优先队列
希尔排序法	O(nlogn) - O(n^2)	O(1)	分组的思想

什么是排序算法的稳定性

liuyubobobo

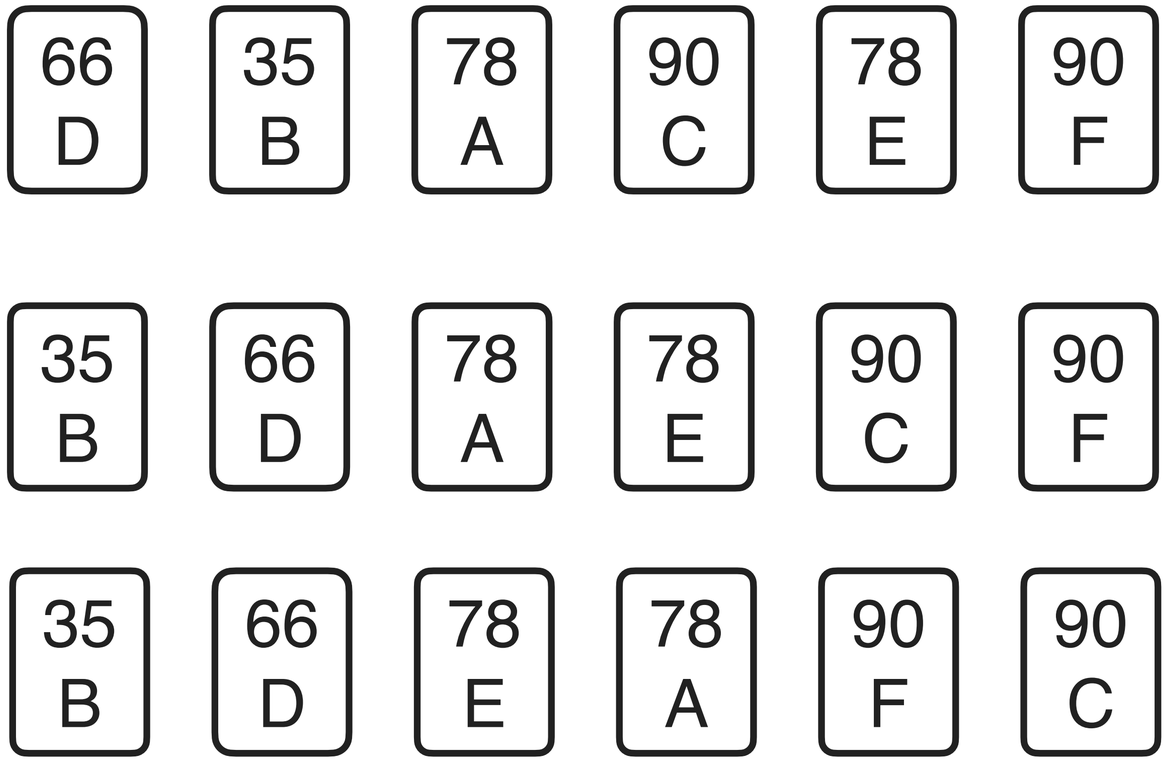
排序算法的稳定性

排序的稳定性：排序前相等的两个元素，排序后相对位置不变



排序算法的稳定性

排序的稳定性：排序前相等的两个元素，排序后相对位置不变

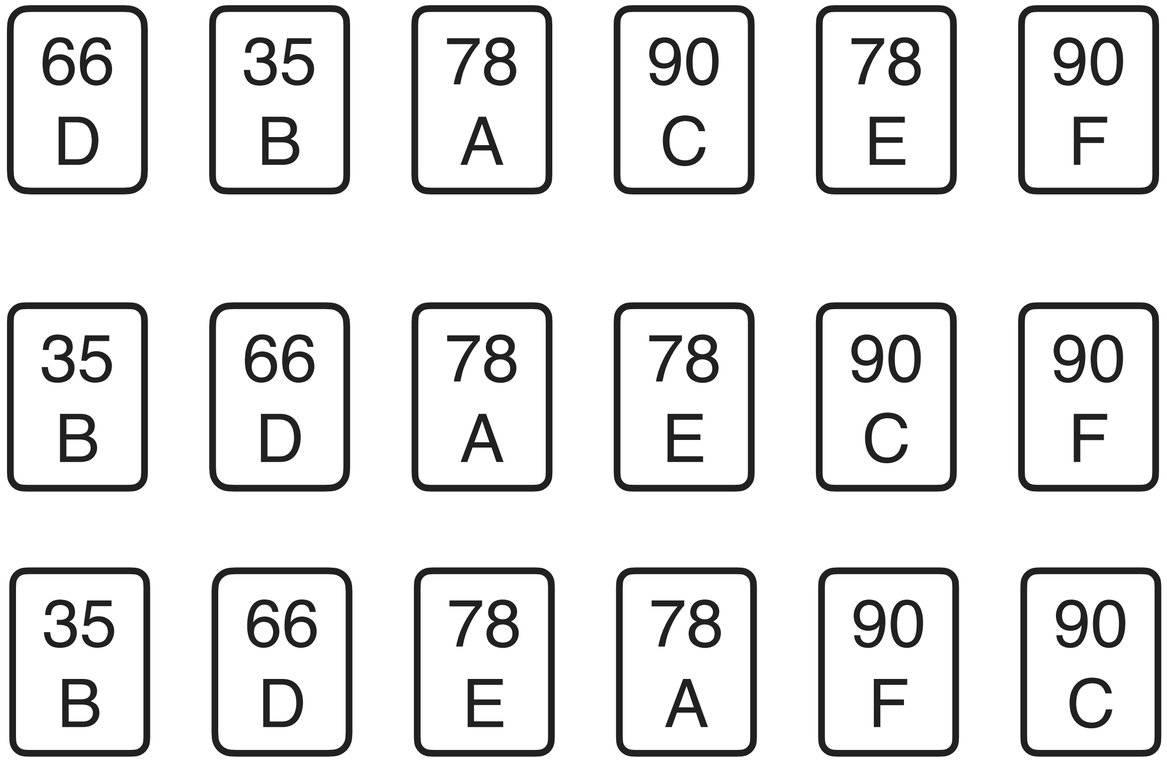


排序算法的稳定性

liuyubobobo

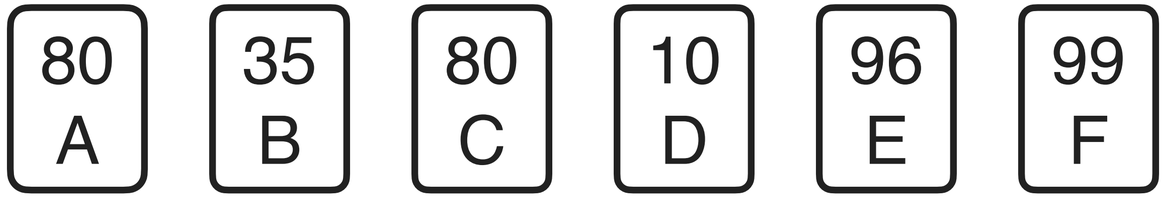
排序算法的稳定性

排序的稳定性：排序前相等的两个元素，排序后相对位置不变



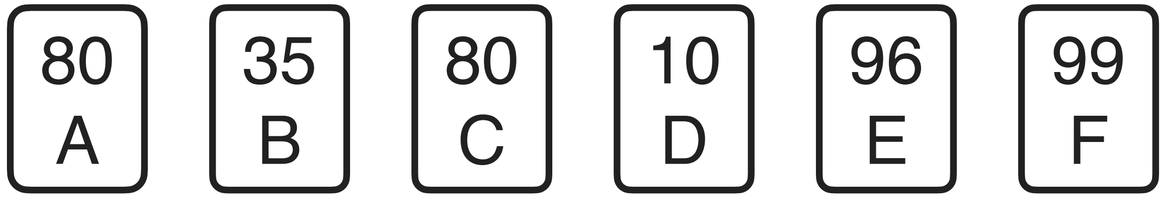
排序算法的稳定性

选择排序法是不稳定的



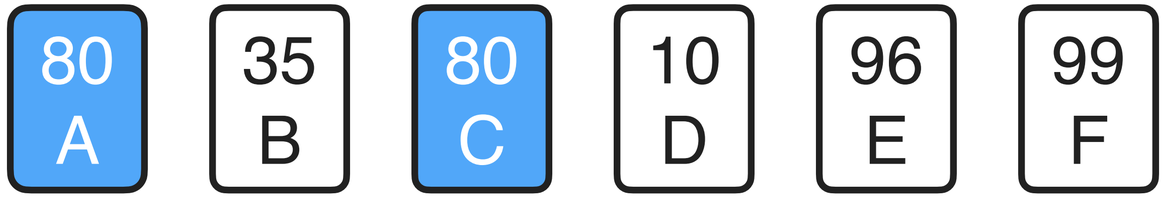
排序算法的稳定性

选择排序法是不稳定的



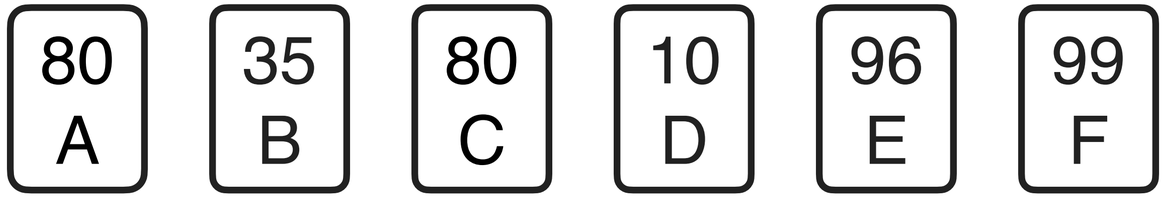
排序算法的稳定性

选择排序法是不稳定的



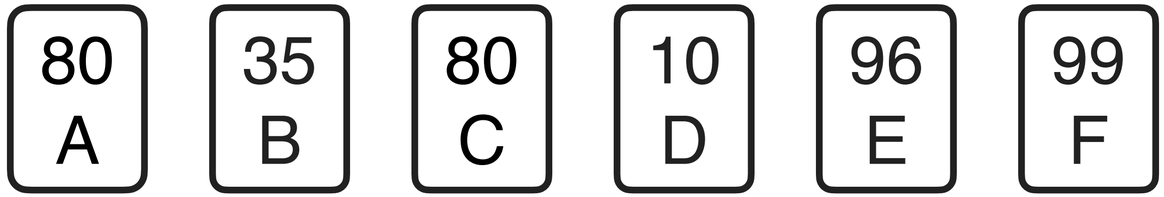
排序算法的稳定性

插入排序法是稳定的



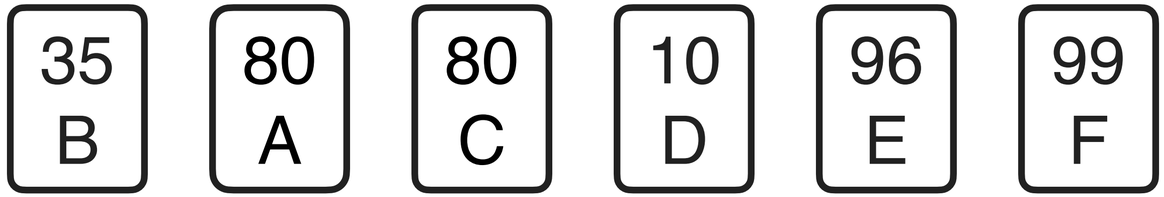
排序算法的稳定性

插入排序法是稳定的



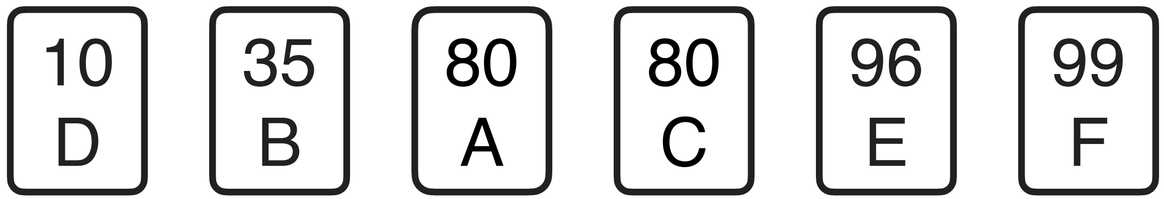
排序算法的稳定性

插入排序法是稳定的



排序算法的稳定性

插入排序法是稳定的



相同大小的元素没有机会“跳跃”

排序算法的稳定性

插入排序法是稳定的　　依赖具体实现

```
public static <E extends Comparable<E>> void sort(E[] arr){

    for(int i = 0; i < arr.length; i ++){

        // 将 arr[i] 插入到合适的位置
        E t = arr[i];
        int j;
        for(j = i; j - 1 >= 0 && t.compareTo(arr[j - 1]) < 0; j --){
            arr[j] = arr[j - 1];
        }
        arr[j] = t;
    }
}
```

排序算法的稳定性

插入排序法是稳定的　　依赖具体实现

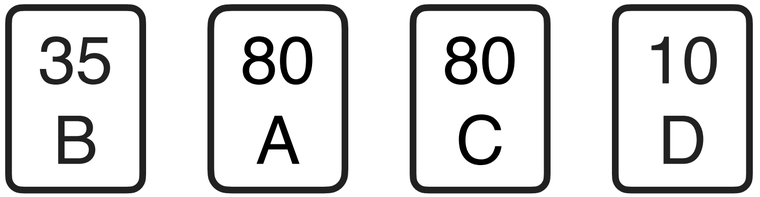
```
public static <E extends Comparable<E>> void sort(E[] arr){

    for(int i = 0; i < arr.length; i ++){

        // 将 arr[i] 插入到合适的位置
        E t = arr[i];
        int j;
        for(j = i; j - 1 >= 0 && t.compareTo(arr[j - 1]) < 0; j --){
            arr[j] = arr[j - 1];
        }
        arr[j] = t;
    }
}
```

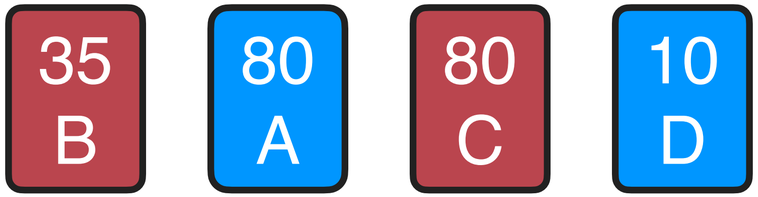
排序算法的稳定性

希尔排序是不稳定的



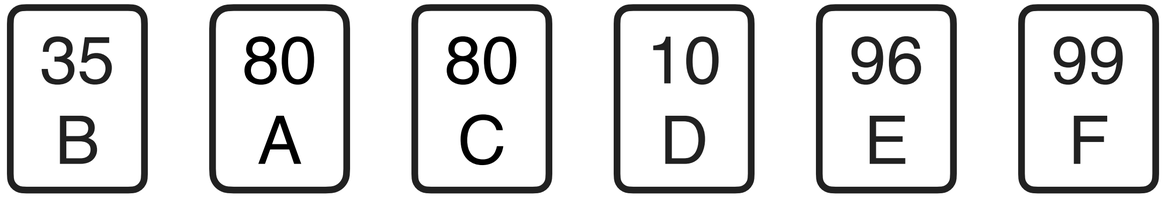
排序算法的稳定性

希尔排序是不稳定的



排序算法的稳定性

冒泡排序法是稳定的

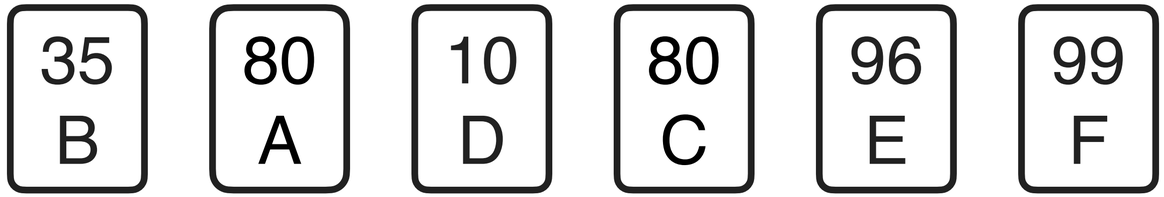


每次只比较相邻元素

相同大小的元素没有机会“跳跃”

排序算法的稳定性

冒泡排序法是稳定的

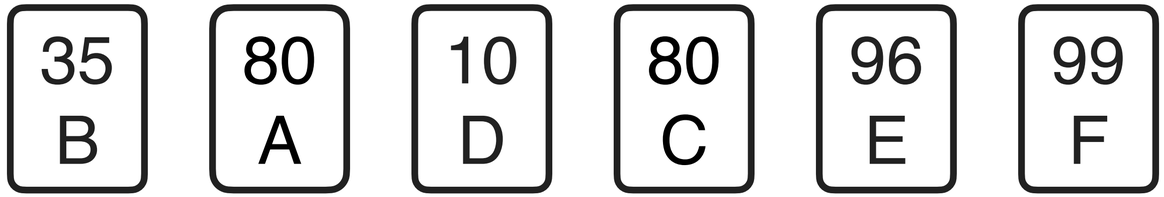


每次只比较相邻元素

相同大小的元素没有机会“跳跃”

排序算法的稳定性

冒泡排序法是稳定的

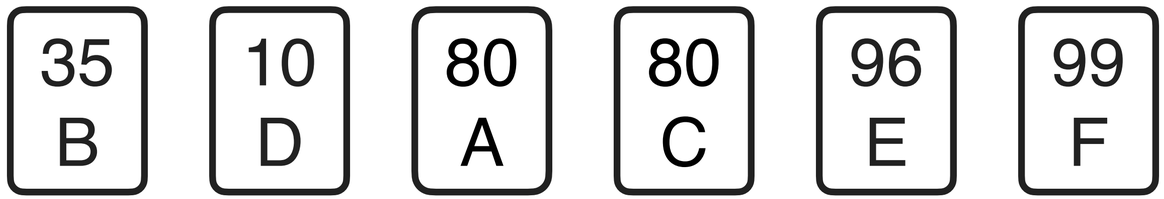


每次只比较相邻元素

相同大小的元素没有机会“跳跃”

排序算法的稳定性

冒泡排序法是稳定的



每次只比较相邻元素

相同大小的元素没有机会“跳跃”

排序算法的稳定性

冒泡排序法是稳定的　　依赖具体实现

```
public static <E extends Comparable<E>> void sort(E[] data){

    for(int i = 0; i + 1 < data.length; ){

        int lastSwappedIndex = 0;
        for(int j = 0; j < data.length - i - 1; j ++)
            if(data[j].compareTo(data[j + 1]) > 0){
                swap(data, j, j + 1);
                lastSwappedIndex = j + 1;
            }

        i = data.length - lastSwappedIndex;
    }
}
```

排序算法的稳定性

插入排序法是稳定的

冒泡排序法是稳定的

选择排序法是不稳定的

希尔排序法是不稳定的

依赖具体实现

一个稳定排序算法

可能被实现成不稳定的排序算法

高级排序算法的稳定性

liuyubobobo

排序算法的稳定性

插入排序法是稳定的

冒泡排序法是稳定的

选择排序法是不稳定的

希尔排序法是不稳定的

依赖具体实现

一个稳定排序算法

可能被实现成不稳定的排序算法

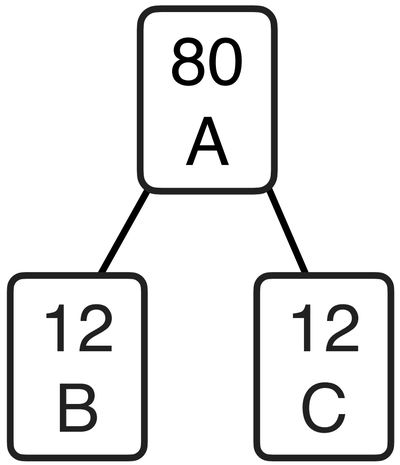
排序算法的稳定性

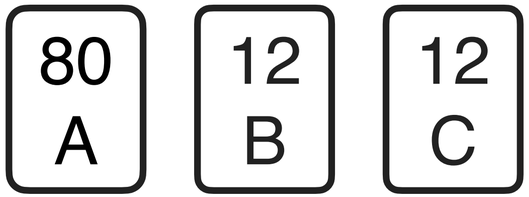
快速排序法是不稳定的

随机化标定点直接打乱了顺序

排序算法的稳定性

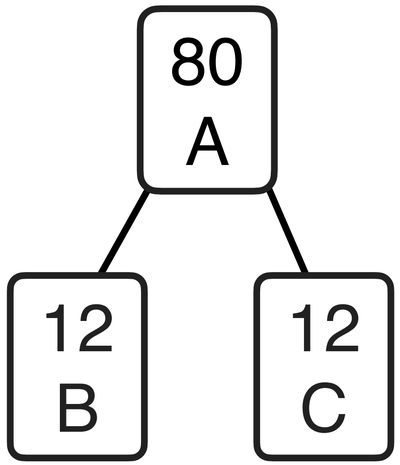
堆排序法是不稳定的

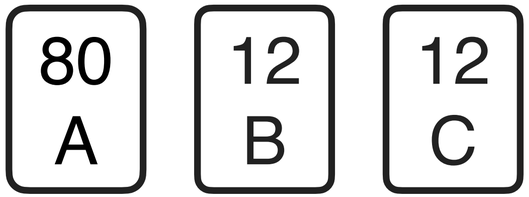




排序算法的稳定性

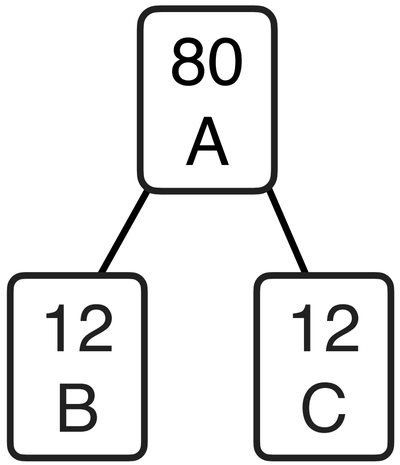
堆排序法是不稳定的

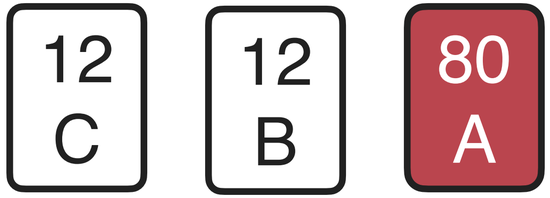




排序算法的稳定性

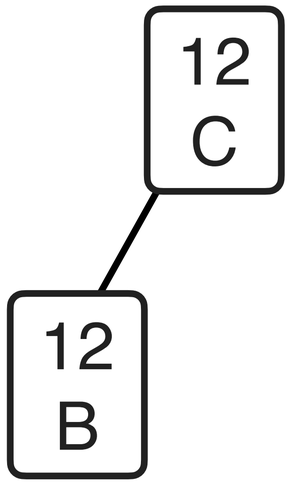
堆排序法是不稳定的

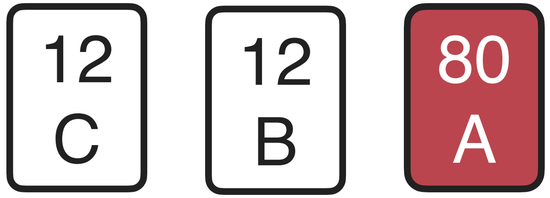




排序算法的稳定性

堆排序法是不稳定的



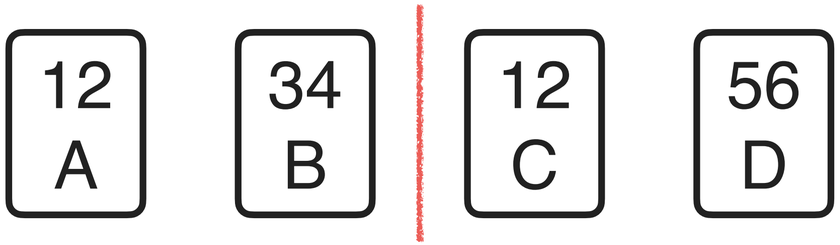


排序算法的稳定性

归并排序法是稳定的

归并排序的元素位置移动，完全在 merge 的过程中

归并过程中，相同元素没机会“跳”到前面去



排序算法的稳定性

归并排序法是稳定的

```
private static <E extends Comparable> void merge(E[] arr, int l, int mid, int r, E[] temp){

    System.arraycopy(arr, l, temp, l, r - l + 1);

    int i = l, j = mid + 1;

    //每轮循环为 arr[k] 赋值
    for(int k = l; k <= r; k ++){
        if(i > mid){    arr[k] = temp[j]; j ++; }
        else if(j > r){ arr[k] = temp[i]; i ++; }
        else if(temp[i].compareTo(temp[j]) <= 0){ arr[k] = temp[i]; i ++;  }
        else{ arr[k] = temp[j]; j ++; }
    }
}
```

排序算法的稳定性

插入排序法是稳定的

冒泡排序法是稳定的

选择排序法是不稳定的

希尔排序法是不稳定的

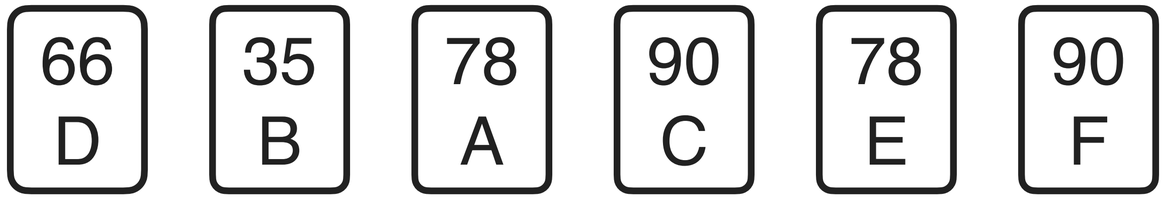
快速排序法是不稳定的

堆排序法是不稳定的

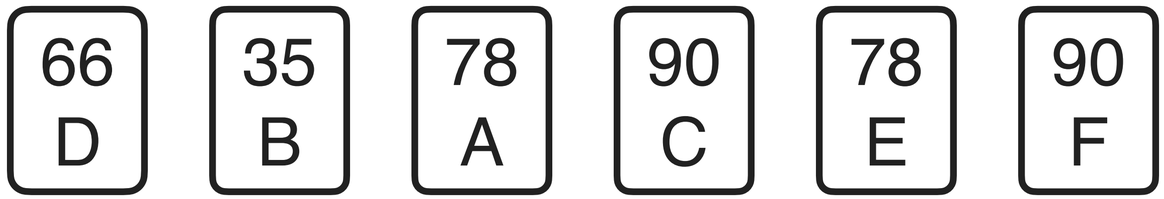
归并排序法是稳定的

排序算法的稳定性

如果元素只有一个域，稳定性没有意义

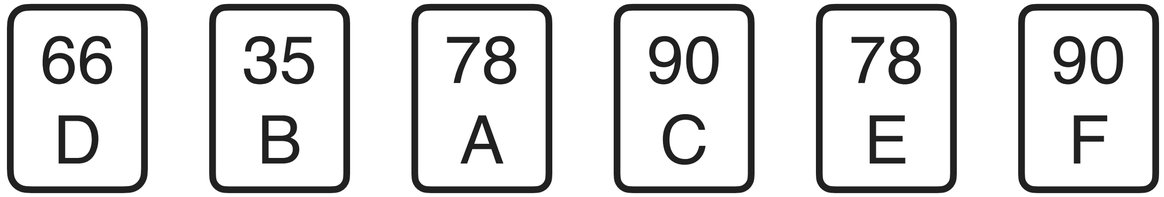


排序算法的稳定性




```
@Override
public int compareTo(Student another){
    return another.score - this.score;
}
```

排序算法的稳定性




```
@Override
public int compareTo(Student another){
    if(another.score != this.score)
        return another.score - this.score;
    else
        return this.name.compareTo(another.name)
}
```

不依赖稳定性

其他

欢迎大家关注我的个人公众号：是不是很酷

算法与数据结构体系课程

liuyubobobo