一、Array和Arrays

1、概念

Array：数组

Arrays：位于java.util包下，是数组的服务类。

提供了一系列的静态方法，用来操作数组。

2、内容

Arrays提供了对数组的：查找、排序、搜索等功能。

```
//asList
List<Integer> list = Arrays.asList(1, 2, 3);

Integer[] data = {1, 2, 3};
List<Integer> list = Arrays.asList(data);


//fill用指定元素填充整个数组（会替换掉数组中原来的元素）
Integer[] data = {1, 2, 3, 4};
Arrays.fill(data, 9);
System.out.println(Arrays.toString(data));

//fill 用指定元素填充数组，从起始位置到结束位置，取头不取尾（会替换掉数组中原来的元素）
Integer[] data = {1, 2, 3, 4};
Arrays.fill(data, 0, 2, 9);
System.out.println(Arrays.toString(data)); // [9, 9, 3, 4]

//sort数组的排序
//Arrays.sort(data)
String[] data = { "1", "4", "3", "2" };
System.out.println(Arrays.toString(data));

Arrays.sort(data);
System.out.println(Arrays.toString(data));

//Arrays.sort(data, Comparator c)
Arrays.sort(data, new Comparator<String>() {

            @Override
            public int compare(String o1, String o2) {
                return -1;
            }
});

//Arrays.sort(Object[] array, int fromIndex, int toIndex)
//指定范围排序
int[] data = { 1,4,3,2,20,8,89};
```

```
        System.out.println(Arrays.toString(data));
        //数组的排序
        Arrays.sort(data,2,6);

        System.out.println(Arrays.toString(data));
//Arrays.sort(Object[] array, int fromIndex, int toIndex, Comparator c)

/**
*Arrays.binarySearch()
注意：在调用该方法之前，必须先调用sort()方法进行排序，如果数组没有排序，
那么结果是不确定的，此外如果数组中包含多个指定元素，则无法保证将找到哪个元素
*/
//Arrays.binarySearch(Object[] array, Object key)
//使用 二分法 查找数组内指定元素的索引值
它的返回值，负数：表示该元素不在数组之内，正数：表示元素存在，并且表示的是该元素的下标。
int[] data = { 1,4,3,2,20,8,89};
//必须先排序
Arrays.sort(data);
System.out.println(Arrays.toString(data));
System.out.println(Arrays.binarySearch(data, 89));

//返回值解析：负数：表示该元素不在数组之内，其数值也是有意义的。
//数组排序以后，有最大值和最小值，最大值和最小值，将整个数轴分成了3部分。
//如果给出的元素，小于最小值，返回-1
//如果给出的元素，大于最大值，返回最大值的下标+1,然后取反（取其负数）。
//如果给出的元素，在范围之间，返回下一个元素的下标+1，然后取反


//copyof  拷贝数组，从下标0开始，如果超过原数组长度，会用null进行填充
Integer[] data1 = { 1, 2, 3, 4 };
        Integer[] data2 = Arrays.copyOf(data1, 2);

        System.out.println(Arrays.toString(data2)); // [1, 2]
        Integer[] data3 = Arrays.copyOf(data1, 5);
        System.out.println(Arrays.toString(data3));

//copyofRange  范围复制 Arrays.copyOfRange(T[] original, int from, int to)
 Integer[] data1 = {1, 2, 3, 4};
        Integer[] data2 = Arrays.copyOfRange(data1, 1, 3);
        System.out.println(Arrays.toString(data2));
        Integer[] data3 = Arrays.copyOfRange(data1, 0, 5);
        System.out.println(Arrays.toString(data3));
```

二、Collection和Collections

1、概念

Collection是java中集合的顶级接口之一。还有一个是Map

Collections位于java.util包，它提供了一些列的静态方法，用来操作集合。

是集合的一个工具类。

2、内容

```
//addAll
List<String> list = new ArrayList<>();
list.add("mazi");
list.add("zhangsan");

//addAll一次性添加多个元素
Collections.addAll(list, "小强", "二狗", "小白");

System.out.println(list);


//copy
List<String> list = new ArrayList<>();
list.add("mazi");
list.add("zhangsan");

List<String> list1 = new ArrayList<>();
list1.add("xiaoqiang");
list1.add("ergou");

Collections.copy(list, list1);
System.out.println(list);
System.out.println(list1);

//emptyList  返回一个空的List集合。但是该集合的类型是：Collections$EmptyList
public class CollectionsDemo {
    public static void main(String[] args) {
        List<String> list = Collections.emptyList();
        System.out.println(list);
    }
}

//同理emptyMap() 、emptySet()  都一样


//集合的排序
List<String> list = new ArrayList<>();
list.add("mazi");
list.add("zhangsan");
list.add("ergou");

System.out.println(list);

//sort 默认情况下，自然排序，升序
Collections.sort(list);
```

```
System.out.println(list);

//sort 指定比较器sort(List<T> list, Comparator<? super T> c)


//集合的反转
Collections.reverse(list);

//集合指定元素的反转
Collections.swap(list, 0, 2);

//集合的线程安全
//默认情况下，除了少数的几个集合是线程安全的之外，其他集合对是线程不安全的
//可以通过Collections中的synchronized开头的方法，让所有的集合都线程安全
```