

## 1、Collection接口中的方法的使用

### 2、方法测试

#### 1、Collection接口中的方法的使用

1. add(Object e):将元素e添加到集合coll中
2. size():获取添加的元素个数
3. addAll(Collection colls):将colls集合中的元素添加到当前的集合中
4. clear():清空集合元素
5. isEmpty():判断当前集合是否为空
- 6.1 contains(Object obj):判断当前集合中是否包含obj, 判断时会调用obj对象所在类的equals()。
- 6.2 containsAll(Collection coll1):判断形参coll1中的所有元素是否都存在于当前集合中。
- 7.1 remove(Object obj):从当前集合中移除obj元素。
- 7.2 removeAll(Collection colls):差集: 从当前集合中移除colls中所有的元素。
8. retainAll(Collection coll1):交集: 获取当前集合和coll1集合的交集, 并返回给当前集合
9. equals(Object obj):要想返回true, 需要当前集合和形参集合的元素都相同。
10. hashCode():返回当前对象的哈希值
11. 集合 ---> 数组: toArray()
- 拓展: 数组 ---> 集合:调用Arrays类的静态方法asList()
12. iterator():返回Iterator接口的实例, 用于遍历集合元素。

总结: 向Collection接口的实现类的对象中添加数据obj时, 要求obj所在类要重写equals()。

#### 2、方法测试

```
1 public class CollectionTest {  
2
```

```

3  @Test
4  public void test1(){
5      Collection coll = new ArrayList();
6
7      //add(Object e):将元素e添加到集合coll中
8      coll.add("AA");
9      coll.add("BB");
10     coll.add(123); //自动装箱
11     coll.add(new Date());
12
13     //size():获取添加的元素个数
14     System.out.println(coll.size()); //4
15
16     //addAll(Collection coll1):将coll1集合中的元素添加到当前的集合中
17     Collection coll1 = new ArrayList();
18     coll1.add(456);
19     coll1.add("CC");
20     coll.addAll(coll1);
21
22     System.out.println(coll.size()); //6
23     System.out.println(coll);
24
25     //clear():清空集合元素
26     coll.clear();
27
28     //isEmpty():判断当前集合是否为空
29     System.out.println(coll.isEmpty());
30
31 }
32 }
33
34

```

```

1  @Test
2  public void test1(){
3      Collection coll = new ArrayList();
4      coll.add(123);
5      coll.add(456);
6      // Person p = new Person("Jerry",20);
7      // coll.add(p);
8      coll.add(new Person("Jerry",20));

```

```

9  coll.add(new String("Tom"));
10 coll.add(false);
11 //1.contains(Object obj):判断当前集合中是否包含obj
12 //我们在判断时会调用obj对象所在类的equals()。
13 boolean contains = coll.contains(123);
14 System.out.println(contains);
15 System.out.println(coll.contains(new String("Tom")));
16 // System.out.println(coll.contains(p)); //true
17 System.out.println(coll.contains(new Person("Jerry", 20))); //false --> true
18
19 //2.containsAll(Collection coll1):判断形参coll1中的所有元素是否都存在于当前
   集合中。
20 Collection coll1 = Arrays.asList(123, 4567);
21 System.out.println(coll.containsAll(coll1));
22 }
23
24 @Test
25 public void test2(){
26 //3.remove(Object obj):从当前集合中移除obj元素。
27 Collection coll = new ArrayList();
28 coll.add(123);
29 coll.add(456);
30 coll.add(new Person("Jerry", 20));
31 coll.add(new String("Tom"));
32 coll.add(false);
33
34 coll.remove(1234);
35 System.out.println(coll);
36
37 coll.remove(new Person("Jerry", 20));
38 System.out.println(coll);
39
40 //4. removeAll(Collection coll1):差集: 从当前集合中移除coll1中所有的元素。
41 Collection coll1 = Arrays.asList(123, 456);
42 coll.removeAll(coll1);
43 System.out.println(coll);
44
45
46 }
47

```

```

48  @Test
49  public void test3(){
50      Collection coll = new ArrayList();
51      coll.add(123);
52      coll.add(456);
53      coll.add(new Person("Jerry",20));
54      coll.add(new String("Tom"));
55      coll.add(false);
56
57      5.retainAll(Collection coll1):交集: 获取当前集合和coll1集合的交集, 并返回给
    当前集合
58      Collection coll1 = Arrays.asList(123,456,789);
59      coll.retainAll(coll1);
60      System.out.println(coll);
61
62      //6.equals(Object obj):要想返回true, 需要当前集合和形参集合的元素都相同。
63      Collection coll1 = new ArrayList();
64      coll1.add(456);
65      coll1.add(123);
66      coll1.add(new Person("Jerry",20));
67      coll1.add(new String("Tom"));
68      coll1.add(false);
69
70      System.out.println(coll.equals(coll1));
71
72
73  }
74
75  @Test
76  public void test4(){
77      Collection coll = new ArrayList();
78      coll.add(123);
79      coll.add(456);
80      coll.add(new Person("Jerry",20));
81      coll.add(new String("Tom"));
82      coll.add(false);
83
84      //7.hashCode():返回当前对象的哈希值
85      System.out.println(coll.hashCode());
86
87      //8.集合 --->数组: toArray()

```

```
88  Object[] arr = coll.toArray();
89  for(int i = 0;i < arr.length;i++){
90  System.out.println(arr[i]);
91  }
92
93  //拓展: 数组 --->集合:调用Arrays类的静态方法asList()
94  List<String> list = Arrays.asList(new String[]{"AA", "BB", "CC"});
95  System.out.println(list);
96
97  List arr1 = Arrays.asList(new int[]{123, 456});
98  System.out.println(arr1.size());//1
99
100 List arr2 = Arrays.asList(new Integer[]{123, 456});
101 System.out.println(arr2.size());//2
102
103 //9.iterator():返回Iterator接口的实例，用于遍历集合元素。
104 }
```