|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 2018 | 年 | 11 | 月 | 12 | 日 |

实验编号：006 **四川师大Java编程实验报告**

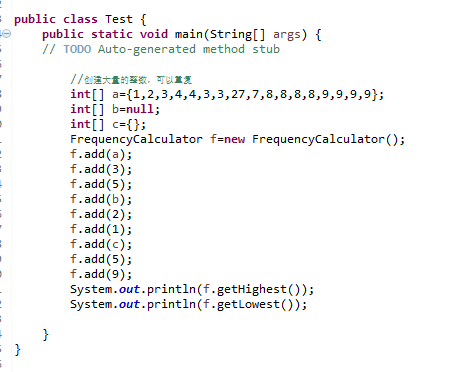
|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **计算机科学学院** | | 2016 | | 级 | | 3 | 班 | 实验名称： | 集合框架 | | |
| 姓名： | 何丹阳 | | 学号： | | 2016110308 | | | 指导老师： | 夏羽 | 实验成绩： |  |

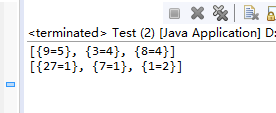
**实验六 集合框架**

1. **实验目的及要求**
2. 掌握Java集合框架的结构和内容。
3. 掌握Java集合框架中常用的类。
4. 掌握List、Set、Map接口及其实现。
5. 了解Java泛型的概念和使用。
6. **实验内容**

根据实验说明的要求编写程序并回答实验报告中的问题。

1. 实验主要流程、基本操作或核心代码、算法片段
2. 请将任务一的运行结果截图。





1. Comparable接口和Comparator接口之间有什么不同？他们分别属于哪一个包？他们各自有什么作用？实现类必须实现接口里面的所有方法，但是我们为什么可以不实现Comparator接口中的equals方法？

区别：Comparable 定义在类的内部，Comparator 是定义在类的外部的, Comparable 接口的对象直接就成为一个可以比较的对象,但是需要修改源代码, Comparator是不需要修改源代码, 而是另外实现一个比较器, 当某个自定义的对象需要作比较的时候,把比较器和对象一起传递过去就可以比大小了。可以说一个是自已完成比较，一个是外部程序实现比较的差别而已。

Comparator位于包java.util下，而Comparable位于包java.lang下

作用：用来实现集合中元素的比较和排序。Comparable强行对实现它的每个类的对象进行整体排序。这种排序被称为类的自然排序，类的 compareTo 方法被称为它的自然比较方法。Comparator强行对某个对象 collection 进行整体排序 的比较函数。可以将 Comparator 传递给 sort 方法（如 Collections.sort 或 Arrays.sort），从而允许在排序顺序上实现精确控制。还可以使用 Comparator 来控制某些数据结构（如有序 set或有序映射）的顺序，或者为那些没有自然顺序的对象 collection 提供排序。

所有类默认继承Object，所以该类已有了Object的equals方法，相当于重写了equals方法，equals是Object提供的。

1. 假设有两个HashSet<String>：set1和set2，执行下面的代码：

|  |
| --- |
| System.out.println(set1);  Systemout.println(set2); |

有以下结果：

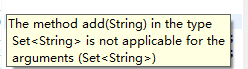
|  |
| --- |
| [red, yellow, green]  [red, yellow, blue] |

执行以下代码之后，在执行打印语句，输出的结果应该是什么？为什么？

1) set1.addAll(set2);

addAll传入的是一个集合。将会存下set2中的所有元素。又因为set是不可重复的。所以结果为[red, green, blue, yellow]。

2) set1.add(set2);

ddd传参为String类型，不能传入Set<String>类型

3) set1.removeAll(set2);

 removeAll是传入一个集合，例如set2，如果set1中存在set2中含有的元素，则移除。所以最后的结果为[green]。

4) set1.remove(set2);

remove传入一个对象，如果set1中含有这个对象，则移除。因为set1中没有[red, blue, yellow]这个元素，所以最后的结果为[red, green, yellow]。

5) set1.retainAll(set2);

retainAll相当于取交集

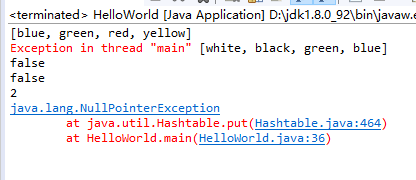
6) set1.clear();

 clear为清除所有映射单元

1. 假设有如下程序：

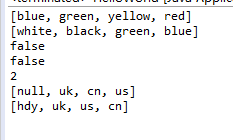
|  |
| --- |
| UseCollections.java |
| import java.util.\*;  public class UseCollections {  public static void main(String[] args) {  List<String> list =  Arrays.asList(“yellow”, “red”, “green”, “blue”);  Collections.reverse(list);  System.out.println(list);  List<String> list1 =  Arrays.asList(“yellow”, “red”, “green”, “blue”);  List<String> list2 = Arrays.asList(“white”, “black”);  Collections.copy(list1, list2);  System.out.println(list1);    Collection<String> c1 = Arrays.asList(“red”, “cyan”);  Collection<String> c2 = Arrays.asList(“red”, “blue”);  Collection<String> c3 = Arrays.asList(“red”, “tan”);  System.out.println(Collections.disjoint(c1, c2));  System.out.println(Collections.disjoint(c1, c3));  Collection<String> collection =  Arrays.asList(“red”, “cyan”, “red”);  System.out.println(Collections.frequency(collection, “red”));  Map<String, String> map1 = new HashMap<String, String>();  Hashtable<String, String> map2 = new Hashtable<String, String>();  map1.put(“cn”, “China”);  map2.put(“cn”, “China”);  map1.put(“us”, “United States”);  map2.put(“us”, “United States”);  map1.put(“uk”, “United Kingdom”);  map2.put(“uk”, “United Kingdom”);  map1.put(null, “Unknown”);  map2.put(null, “Unknown”);  System.out.println(map1.keySet());  System.out.println(map2.keySet());  }  } |

程序的输出是什么？若程序运行出现错误，请对程序进行修改（如何修改？），然后对每一个输出进行解释。



map2.put(null, “Unknown”),HashTable不允许null健。





第一个输出：Collections.reverse反转集合中元素的顺序

第二个输出：Collections.copy只有当list1的 size 大于或者等于list2的size时才进行拷贝，拷贝把相同位置给覆盖

第三个输出：Collections.disjoint用于为'true'如果两个指定collection中没有相同的元素，因为都有red，所以返回false

第四个输出：Collections.disjoint用于为'true'如果两个指定collection中没有相同的元素，因为都有red，所以返回false

第五个输出：frequency 是 Collections 的一个方法，可找出一个单词在list中出现的次数

第六个输出：HashMap中元素不会排序，且允许有null健，但不会有相同的健

第七个输出：HashTable中元素不会排序，不允许有null健，也不会有相同的健

1. **实验结果的分析与评价**

在这次实验中，关于接口的使用更深入了一些，对collections的一些函数也是有了新的认识。但是感觉自己还是没有很好地使用API文档，很少回去看，要么百度要么问同学，觉得自己应该多去看看API文档，还可以学习英语。

注：实验成绩等级分为（90－100分）优，（80－89分）良，(70-79分)中，（60－69分）及格，（59分）不及格。