### Java第一阶段—DAY15-JAVA作业答案

1. 请简述集合框架。

集合按照其存储结构可以分为两大类，分别是单列集合java.util.Collection和双列集合java.util.Map。Collection：单列集合类的根接口，用于存储一系列符合某种规则的元素，它有两个重要的子接口，分别是java.util.List和java.util.Set。其中，List的特点是元素有序、元素可重复。Set的特点是元素无序，而且不可重复。List接口的主要实现类有java.util.ArrayList和java.util.LinkedList，Set接口的主要实现类有java.util.HashSet和java.util.TreeSet

1. 给定以下代码，请定义方法listTest()统计集合中指定元素出现的次数， 如"a": 2,"b": 2,"c" :1, "xxx":0。

Collection<String> list = new ArrayList<>();

list.add("a");

list.add("a");

list.add("b");

list.add("b");

list.add("c");

System.out.println("a:"+listTest(list, "a"));

System.out.println("b:"+listTest(list, "b"));

System.out.println("c:"+listTest(list, "c"));

System.out.println("xxx:"+listTest(list, "xxx")); 。

**public class** CollectionTest01{  
 **public static void** main(String[] args) {  
 Collection<String> list = **new** ArrayList<>();  
 list.add(**"a"**);  
 list.add(**"a"**);  
 list.add(**"b"**);  
 list.add(**"b"**);  
 list.add(**"c"**);  
 System.**out**.println(**"a:"**+listTest(list, **"a"**));  
 System.**out**.println(**"b:"**+listTest(list, **"b"**));  
 System.**out**.println(**"c:"**+listTest(list, **"c"**));  
 System.**out**.println(**"xxx:"**+listTest(list, **"xxx"**));  
 }  
  
 //定义方法统计集合中指定元素出现的次数  
 **public static int** listTest(Collection<String> list,String s){  
 //定义计数器，初始化为0  
 **int** count = 0;  
 //增强for遍历集合  
 **for** (String string : list) {  
 //判断传入方法的字符与遍历集合的是否一致  
 **if** (s.equals(string)) {  
 //如果一致，加1  
 count++;  
 }  
 }  
 **return** count;  
 }  
}

1. 定义一个方法，要求此方法把int数组转成存有相同元素的集合(集合里面的元素是Integer)，并返回。

**public class** CollectionTest02 {  
 **public static void** main(String[] args) {  
 //定义int数组  
 **int**[] arr = {1,2,3,4,5};  
 ArrayList<Integer> list = listTest(arr);  
 System.**out**.println(list);  
 }  
  
 **public static** ArrayList<Integer> listTest(**int**[] arr) {  
 //定义集合  
 ArrayList<Integer> list = **new** ArrayList<Integer>();  
 //遍历数组，把元素依次添加到集合当中  
 **for** (**int** a : arr) {  
 list.add(a);  
 }  
 **return** list;  
 }  
}

1. 简述迭代器的实现原理。

当遍历集合时，首先通过调用集合的iterator()方法获得迭代器对象，然后使用hashNext()方法判断集合中是否存在下一个元素，如果存在，则调用next()方法将元素取出，否则说明已到达了集合末尾，停止遍历元素。Iterator迭代器对象在遍历集合时，内部采用指针的方式来跟踪集合中的元素，在调用Iterator的next()方法之前，迭代器的索引位于第一个元素之前，不指向任何元素，当第一次调用迭代器的next方法后，迭代器的索引会向后移动一位，指向第一个元素并将该元素返回，当再次调用next方法时，迭代器的索引会指向第二个元素并将该元素返回，依此类推，直到hasNext方法返回false，表示到达了集合的末尾，终止对元素的遍历。

1. **定义人类，包含姓名和年龄属性。创建4个人存储到HashSet中，姓名和年龄相同的人看做同一人不存储。**

// 1.定义Person类.包好姓名年龄属性,重写hashCode()和equals()方法  
**public class** Person {  
 **private** String **name**;  
 **private int age**;  
  
 **public** Person() {  
 }  
  
 **public** Person(String name, **int** age) {  
 **this**.**name** = name;  
 **this**.**age** = age;  
 }  
  
 @Override  
 **public boolean** equals(Object o) {  
 **if** (**this** == o) **return true**;  
 **if** (!(o **instanceof** Person)) **return false**;  
  
 Person person = (Person) o;  
  
 **if** (**age** != person.**age**) **return false**;  
 **return name** != **null** ? **name**.equals(person.**name**) : person.**name** == **null**;  
 }  
  
 @Override  
 **public int** hashCode() {  
 **int** result = **name** != **null** ? **name**.hashCode() : 0;  
 result = 31 \* result + **age**;  
 **return** result;  
 }  
  
 @Override  
 **public** String toString() {  
 **return "Person{"** +  
 **"name='"** + **name** + **'\''** +  
 **", age="** + **age** +  
 **'}'**;  
 }  
}

测试类

**public class** HashSetTest01 {  
 **public static void** main(String[] args) {  
 // 2.创建HashSet用于存储Person类型  
 HashSet<Person> hashSet = **new** HashSet<Person>();  
  
 // 3.添加多个Person到HashSet中  
 hashSet.add(**new** Person(**"王昭君"**, 21));  
 hashSet.add(**new** Person(**"西施"**, 21));  
 hashSet.add(**new** Person(**"杨玉环"**, 20));  
 hashSet.add(**new** Person(**"貂蝉"**, 19));  
 hashSet.add(**new** Person(**"杨玉环"**, 20));  
 hashSet.add(**new** Person(**"貂蝉"**, 19));  
  
 // 4.遍历获取HashSet中的内容  
 **for** (Person p : hashSet) {  
 System.**out**.println(p);  
 }  
 }  
}