**06\_EnhancedFor\_and\_Set\_笔记**

**一、增强for循环 ﻿**

**1. 例题:for循环遍历 ﻿**

A computer screen shot of a computer

AI-generated content may be incorrect.

* 语法形式：JDK1.5新增的for循环语法为for(元素类型 变量名 : 集合/数组)，可简化数组和Collection的遍历
* 数组遍历示例：
* 集合遍历示例：
* 执行原理：内部通过迭代器(Iterator)实现，相当于隐式调用了集合的iterator()方法

**2. 增强for循环总结**

A screenshot of a computer

AI-generated content may be incorrect.

* 优点：
  + 语法简洁，遍历数组和集合时代码更紧凑
  + 无需手动处理迭代器或下标，减少样板代码
* 数组缺陷：
  + 无法直接访问当前元素的下标值
  + 示例：需要特殊处理第3个元素时，需额外维护计数器变量
* 集合缺陷：
  + 相比显式使用Iterator，无法调用remove()删除元素
  + 底层仍使用Iterator，会锁定集合导致并发修改异常
* 使用建议：
  + 仅推荐用于简单的只读遍历场景
  + 需要下标操作或修改集合时，应使用传统for循环或显式Iterator
  + 实际开发中应避免过度依赖增强for循环

**二、Set接口**

A screenshot of a computer

AI-generated content may be incorrect.

* 继承关系: Set接口是Collection的子接口，没有提供额外的方法
* 特性要求:
  + 无序性: 实现Set接口的容器类元素没有顺序
  + 唯一性: 元素不可重复，对应数学中的"集合"概念
* 实现类:
  + HashSet: 基于哈希表实现
  + TreeSet: 基于树结构实现

**1. 例题:set添加元素**

* A screenshot of a computer program

  AI-generated content may be incorrect.
* 示例代码解析:
  + 创建HashSet并添加元素："hello"、"world"、Name对象("f1","f2")、Integer(100)
  + 重复元素处理:
    - 再次添加"hello"和相同Name对象时会被自动过滤
    - 输出结果：[100, hello, world, f1 f2]
* 底层原理:
  + equals方法重写:
    - String类已重写equals方法
    - Name类自定义equals逻辑：当firstName和lastName都相同时视为相等
  + hashCode方法:
    - Name类基于firstName生成hashCode
    - 哈希值相同才会进一步比较equals
* 集合操作:
  + 交集(retainAll): 保留两个集合共有的元素
  + 并集(addAll): 合并两个集合，自动去重
  + 示例说明:
    - 集合s1包含{a,b,c}，s2包含{a,b,d}
    - 交集结果为{a,b}
    - 并集结果为{a,b,c,d}（无序）
* 编程建议:
  + 课后必须独立重写示例代码
  + 阅读API文档理解Set方法
  + 避免机械打字，理解后编码
  + 培养快速编码能力以适应项目开发节奏

**三、知识小结**

|  |  |  |  |
| --- | --- | --- | --- |
| 知识点 | 核心内容 | 考试重点/易混淆点 | 难度系数 |
| JDK 1.5增强for循环 | 提供简化数组/集合遍历的语法：for(int i : array) | 无法直接访问下标，删除集合元素受限 | ⭐⭐ |
| Set接口特性 | 无序且不可重复的集合，继承自Collection接口 | 重复元素判定依据equals()方法 | ⭐⭐ |
| HashSet实现类 | 基于哈希表实现，允许null元素 | add()方法对重复元素的处理机制 | ⭐⭐ |
| 集合运算 | retainAll()求交集，addAll()求并集 | 并集操作自动去重特性 | ⭐⭐⭐ |
| 集合遍历限制 | 增强for循环底层使用Iterator，修改集合会触发ConcurrentModificationException | 遍历时只能通过Iterator的remove()方法删除元素 | ⭐⭐⭐⭐ |

该部分主要讲述了Java中的Set接口及其相关容器类的特点和使用方法。Set接口不允许存储重复元素，其容器类有HashSet和TreeSet等。

HashSet基于哈希表实现，数据无序；TreeSet基于树结构，数据有序。视频通过示例展示了Set的特点，并介绍了如何求交集。最后强调了培养写代码的感觉和习惯的重要性。

* **分段总结**

**增强for循环介绍**

1.JDK1.5引入了增强for循环，简化了数组和集合的遍历过程。 2.增强for循环的语法为：for(type element:collection){statement}，其中type表示元素类型，element表示当前元素，collection表示要遍历的集合。 3.增强for循环可以方便地遍历数组和集合，并将每个元素依次取出进行处理。

**增强for循环的示例**

1.示例1：遍历数组。使用增强for循环可以简洁地遍历数组并打印每个元素。 2.示例2：遍历集合。使用增强for循环可以方便地遍历集合中的每个元素并将其打印出来。 3.增强for循环在遍历数组时无法方便地访问下标值，在遍历集合时无法方便地删除集合中的内容。

**Set接口介绍**

1.Set接口是Collection接口的子接口，所有实现Set接口的容器类都必须满足无序且不包含重复元素的要求。

2.Set接口没有提供额外的方法，但实现了该接口的容器类通常使用哈希表或树结构来实现。

3.常见的Set实现类有HashSet和TreeSet。

**Set接口的示例**

1.示例1：添加元素到Set集合中，重复元素不会被添加。

2.示例2：合并两个Set集合，返回它们的并集、交集和差集。

3.Set集合中的元素没有顺序，打印出来的结果可能会不同，但只要元素被正确打印出来即可。