**一、容器**

**1. 容器的概念**

A screenshot of a computer

AI-generated content may be incorrect.

* 本质：用于存储和管理其他对象的对象集合
* 与传统数组对比：
  + 数组长度固定，容器长度动态可变
  + 数组空间利用率低（预分配过大空间浪费，过小则扩展困难）
  + 容器自动扩容缩容，内存管理更高效

**1）例题:人名管理**

A screenshot of a computer

AI-generated content may be incorrect.

* 问题场景：管理公司员工姓名（Name类包含firstName和lastName）
* 传统方案缺陷：
  + 数组方案：需预分配固定空间（如100人），实际使用可能远小于或超过
  + 扩展困难：数组扩容需创建新数组并拷贝数据
* 容器优势：
  + 动态增减：随元素数量自动调整存储空间
  + 灵活管理：无需预先确定容量上限

**2. 容器API ﻿**

**1）容器API的类图结构 ﻿**

A screenshot of a computer

AI-generated content may be incorrect.

* 组织架构：
  + 位于java.util包
  + 核心接口：Collection（单元素存储）和Map（键值对存储）
* 设计理念：
  + 接口定义标准：统一操作规范（类比高通专利模式）
  + 实现分离：具体实现类（ArrayList/HashSet等）可自由替换
* 编程实践：
  + 推荐使用父接口声明（如Collection c = new ArrayList()）
  + 优势：提高代码灵活性，便于后续实现类替换

**2）Collection接口**

**﻿**

A screenshot of a computer

AI-generated content may be incorrect.

* 核心方法：
  + 容量操作：size()/isEmpty()/clear()
  + 元素操作：add()/remove()/contains()（基于equals判断）
  + 批量操作：addAll()/removeAll()/retainAll()（集合运算）
  + 转换方法：toArray()转为对象数组
* 子接口特性：
  + Set：无序且元素唯一（重复判定：equals返回true）
  + List：有序且允许重复
* 实现类体系：
  + Set实现：HashSet等
  + List实现：ArrayList/LinkedList等
* A screenshot of a computer program

  AI-generated content may be incorrect.例题:Collection用法
  + 关键实现细节：
    - 可存储异构对象（String、自定义类、包装类等）
    - toString()实现：自动调用元素toString并用逗号连接
  + 注意事项：
    - 只能存储对象（不能存基本类型，需使用包装类）
    - 打印输出格式：[元素1, 元素2,...]
  + 典型输出：

**二、知识小结**

|  |  |  |  |
| --- | --- | --- | --- |
| 知识点 | 核心内容 | 考试重点/易混淆点 | 难度系数 |
| 容器概念 | 容器是用于存储和管理其他对象的对象，类似于现实中的容器 | 数组也是容器，但长度固定不灵活 | ⭐⭐ |
| Collection接口 | 定义了一组存取对象的方法，是容器API的基础接口 | 为什么要用接口定义容器标准 | ⭐⭐⭐ |
| List与Set区别 | List有顺序可重复，Set无顺序不可重复 | 重复判断标准是equals()方法 | ⭐⭐⭐⭐ |
| ArrayList特性 | 基于数组实现的动态扩容List | 与数组相比的优势(自动扩容)和劣势(插入删除效率) | ⭐⭐⭐ |
| 泛型与自动装箱 | 容器只能存储对象，基础类型需自动装箱 | Integer等包装类的使用场景 | ⭐⭐⭐⭐ |
| Map接口 | 存储键值对的容器，与Collection并列 | 键的唯一性判断标准 | ⭐⭐⭐⭐ |
| 容器API组织 | 位于java.util包，分为Collection和Map两大体系 | "1136"记忆法(1图3知识点6接口) | ⭐⭐⭐⭐ |
| 接口编程优势 | 使用Collection接口声明变量提高灵活性 | 可随时更换具体实现类而不影响业务逻辑 | ⭐⭐⭐⭐⭐ |

该部分主要讲述了Java中的接口和集合框架。接口定义了存取对象的方法，子接口如List和Set定义了存储方式中的数据。实现Collection接口的类应实现一系列方法，如isEmpty、clear、contains、add和addAll等。

此外，还介绍了Array List、HashSet、LinkedList、HashMap等实现了不同接口的集合类。使用父类引用指向子类对象可实现代码的灵活性和可维护性。在添加元素时，只能添加对象，不能添加基本引用类型或基本数据类型。最后，演示了如何使用toString方法打印集合对象的内容。

容器的概念

1.容器是装其他东西的器皿，用于存储对象。

2.容器在编程中用于存储和管理对象引用。

3.容器的基本概念包括数组、集合、映射等。

容器API概述

1.Java提供了丰富的容器API，用于处理各种数据结构。

2.容器API包括Collection接口、Set接口、List接口、Map接口等。

3.这些接口定义了容器的基本操作和方法。

容器的应用场景

1.容器广泛应用于各种编程场景，如人事管理系统。

2.容器用于存储和管理人员信息，如姓名、职位等。

3.容器提供了灵活的数据存储和访问方式。

容器分类

1.容器分为Collection和Map两大类。

2.Collection用于存储单个对象，如数组列表、哈希集合等。

3.Map用于存储键值对，如哈希映射、树映射等。

容器API组织结构

1.Java容器的API组织结构清晰，便于理解和使用。

2.容器API位于java.util包中，包含各种工具类和接口。

3.通过类图可以了解JDK提供的容器及其组织关系。

Collection接口

1.Collection是所有单元素容器的根接口。

2.Collection接口定义了容器的基本操作，如添加、删除、大小查询等。

3.Collection接口的子接口包括Set和List。

Set接口

1.Set接口表示无序且不包含重复元素的集合。

2.Set接口的实现类如HashSet，不保证元素的顺序。

3.Set中的元素通过equals方法进行比较是否重复。

List接口

1.List接口表示有序且可包含重复元素的集合。

2.List接口的实现类如ArrayList和LinkedList。

3.List维护元素的插入顺序，允许重复元素。

Map接口

1.Map接口用于存储键值对数据。

2.Map接口的实现类如HashMap和TreeMap。

3.Map中的键是唯一的，值可以是任意对象。

容器的实现类

1.Java提供了多种实现Collection、Set、List和Map接口的类。

2.如ArrayList、LinkedList、HashSet等。

3.这些类实现了容器接口定义的方法，用于存储和管理数据。

容器的使用方式

1.通过父类引用指向子类对象的方式使用容器，增加灵活性。

2.例如，使用Collection接口引用ArrayList或LinkedList对象。

3.这种方式允许在运行时动态更换容器的实现，而无需修改代码逻辑。