13216　Java 语言程序设计（实践）考试大纲

自学用书：Java 语言程序设计 ，全国国等教育自学考试指导委员会组编，机械工业出版社，2024 年 9 月第一版

一、 课程性质与设置目的要求

《Java 语言程序设计（实践）》课程是与《Java 语言程序设计》课程所对应的一门实践课。Java 语言具有面向对象、与平台无关、安全、稳定和多线程等特点。通过本课程的实践环节训练，使应考者对使用 Java 语言进行面向对象程序设计有一个完整的认识，基本掌握用 Java 语言进行程序设计的基本方法和技术，为今后各种应用编程特别是网络编程打下基础。

《文书学》教材内容共分四篇。基础篇主要阐述文书学的基础理论；写作篇讲授公文写作的方法与技巧；处理篇介绍文书处理程序及特殊类型、新型载体文件的处理规范；立卷篇主要阐释文书立卷的原理与操作规程。全书在系统探讨相关理论的同时突出了内容的实用性。在自学考试命题中应充分体现本课程的性质和特点。

设置《Java 语言程序设计（实践）》课程的目的，在于与《Java 语言程序设计》理论课程紧密衔接，依托 Java 语言面向对象、平台无关、安全、稳定及多线程等技术特性，通过系统化的实践训练，帮助应考者建立对 Java 面向对象程序设计的完整认知，熟练掌握 Java 程序设计的基础方法与核心技术，全面强化实践能力与应用思维，从而为后续各类应用开发尤其是网络编程领域的学习与实践奠定坚实基础。。

学习《Java 语言程序设计（实践）》课程，要求学生能够与《Java 语言程序设计》理论教学紧密配合，依托 Java 语言面向对象、平台无关、安全、稳定及多线程等技术特性，在实验中独立完成 Java 开发与运行环境的搭建，熟练掌握 Java 基本语法与编程思想，重点理解并实践面向对象编程中继承与多态的核心机制，具备利用常用 GUI 组件开发图形用户界面应用程序的能力，同时掌握文件操作与多线程编程的原理及实现方法，最终能够综合运用所学知识分析问题、设计方案并完成一般性应用程序的开发，从而为后续各类应用开发尤其是网络编程奠定坚实的实践基础。

二、考核内容及目标

第 1 章 熟悉 Java 程序开发环境

1、学习目的和要求

本节旨在让学生掌握 Java 开发环境的基本配置与使用，理解 Java 程序开发的核心流程。你将熟悉常用开发工具的操作，并熟练掌握 Java 中数据的输入和输出方法。

学生应能够独立完成 JDK 的安装与环境变量配置，熟练掌握 Java 程序从编写、编译到运行的完整流程；同时需了解 Eclipse 集成开发环境的基本操作，熟悉 Java Application 的基本结构并能编写简单的应用程序，学会运用 Scanner 类、System.out 对象以及图形界面对话框等多种方式实现数据的输入与输出。

2、考核知识点

（一）JDK 安装与配置：

JDK 的下载、安装步骤。

环境变量的设置方法。

命令行验证安装结果。

（二）Java 程序开发流程：

源文件编写、编译、运行的步骤。

JVM 的基本作用。

常见编译和运行错误的识别。

（三）Eclipse 使用：

创建和管理 Java 项目。

代码编写与调试的基本操作。

（四）Java 程序结构：

类和 main 方法的定义。

变量声明、数据类型、运算符。

条件语句和循环结构。

（五）输入输出：

Scanner 类的使用。

System.out 的输出方法。

JOptionPane 对话框的实现。

格式化输出的应用

3、考核目标要求

（一）Java 开发环境的构建与配置

(1)识记

准确复述 JDK、JRE 与 JVM 的定义及区别；列举 Java 开发环境搭建所需的主要步骤。

(2)理解

解释环境变量（如 PATH 和 CLASSPATH）的作用及其对系统运行 Java 命令的影响；阐明 JVM 在跨平台特性中的核心作用。

(3)简单应用

能够在 Windows/Linux 系统上独立完成 JDK 的安装，并正确配置环境变量；能够启动 Eclipse 并进行基本的视图切换。

(4)综合应用

针对安装过程中出现的常见错误（如“javac 不是内部或外部命令”）进行分析与排查,并根据项目需求配置 Eclipse 的工作空间与项目属性。

（二）Java 程序运行机制与基础语法结构

（1） 识记

记忆 Java Application 的基本结构（类定义、main 方法）；熟记 System.out.print 与 Scanner 类的常用方法名。

（2） 理解

阐述 Java 程序从源代码（.java）到字节码（.class）再到由 JVM 执行的整个过程；理解 API 文档的组织结构及查阅方法。

（3） 简单应用

能够编写包含单行/多行注释的简单程序；熟练使用 Scanner 类接收键盘输入，并使用 System.out 输出结果。

（4） 综合应用

综合运用输入输出流与图形界面（JOptionPane），编写具备人机交互功能的控制台或桌面小程序，处理不同类型数据的混合输入。

（三）面向对象编程基础与开发工具使用

（1） 识记

记忆 Eclipse 中常用的快捷键；识记 Java 程序的基本命名规范与语法规则。

（2） 理解

理解类与对象的概念;说明 Eclipse 中“Project”、“Package”、“Class”之间的层级关系；解释 API 文档中类与方法的含义。

（3） 简单应用

能够在 Eclipse 中创建项目、包及类，并成功编译运行第一个 Java 程序；能够使用 JOptionPane 弹出简单的消息框或输入框。

（4） 综合应用

结合 API 文档，在 Eclipse 中开发一个包含标准输入输出与图形界面交互的小型综合案例，展示对程序结构与开发工具的融合运用能力。。

4、本章重点

本章重点聚焦于 Java 开发入门与基础环境的构建，核心在于通过动手实践确立学生对 Java 语言的完整认知。首先，要求学生掌握 JDK 的安装与环境变量配置，透彻理解 JVM 的跨平台原理及 Java 程序从编写、编译到运行的完整生命周期；其次，需熟悉 Eclipse 集成开发环境，掌握项目、包及类的创建与管理，并学会查阅 API 文档以解决技术问题；最后，通过熟练掌握 Scanner 键盘输入、System.out 控制台输出及 JOptionPane 图形对话框三种交互方式，具备编写并运行包含人机交互功能的简单应用程序的综合能力，为后续深入学习打下坚实基础。

第 2 章 数据和表达式

1、学习目的和要求

本章重点围绕 Java 语言基础语法与数据运算展开，旨在帮助学生建立严谨的编码规范意识与逻辑计算能力。具体内容包括：一是代码规范，要求正确认识并严格遵循标识符的命名规则，准确使用关键字，并规范地编写单行与多行注释；二是数据类型与变量，熟练掌握基本数据类型的特征、常量的定义及转义字符的用法，重点理解变量的声明、赋值机制及其作用域范围；三是运算与求值，深入理解各类常用运算符的优先级与结合性，规范书写表达式，并能熟练调用 Math 类的随机数与数学函数进行复杂的表达式求值与逻辑计算。

2、考核知识点

本实验围绕 Java 语言基础知识的掌握情况进行考核，具体知识点包括：

（1） Java 程序的基本语法结构

类的声明与 main 方法的格式

注释、空白符与代码缩进规范

（2） 标识符与关键字使用规范

命名规则（驼峰命名、不能使用关键字等）

合法与非法标识符的区分

（3） 基本数据类型与变量声明

Java 八种基本数据类型及其关键字

变量的定义、初始化与作用域

（4） 表达式与运算符使用

算术运算、关系运算、逻辑运算的使用

运算符优先级与表达式求值顺序

表达式中的类型提升与强制类型转换

（5） 控制台输入输出语句的应用

System.out.print/println/printf 的用法

格式化输出中的格式控制符使用

（6） 数学函数库的调用方法

Math 类的常用函数：如 sqrt、round、sin、cos 等

表达式嵌套调用的语法规范

3、考核目标要求

（一）Java 编码规范与程序结构

（1） 识记

记忆 Java 标识符的命名规则；识别常见的关键字；熟记单行、多行及文档注释的格式符号。

（2） 理解

理解非法标识符与关键字误用的后果；理解代码缩进与格式规范对程序可读性的重要性。

（3） 简单应用

能够编写一个包含 main 方法的完整 Java 类；在代码中正确使用注释说明逻辑；避免使用非法标识符。

（4） 综合应用

能够编写结构清晰、格式标准、无任何语法错误的综合性程序，并体现出良好的代码风格与规范意识。

（二）变量声明、作用域与类型转换

（1） 识记

记忆基本数据类型的分类及占用的字节数；记忆强制类型转换的语法格式（如 (int)）。

（2） 理解

理解变量作用域的范围（块级作用域）；理解自动类型提升与强制类型转换的区别；理解 float 与 double 混合计算时的精度差异。

（3） 简单应用

能够正确声明并初始化变量，确保类型匹配；能在不同作用域内定义同名变量并理解其覆盖关系。

（4） 综合应用

在复杂表达式中灵活运用隐式转换与显式转换，正确处理浮点数与整数混合运算带来的精度丢失问题，确保计算结果符合预期。

（三）运算符、表达式与 Math 类应用

（1） 识记

记忆算术、关系、逻辑运算符的符号；熟记 Math 类中常用静态方法（如 sqrt, round, sin）的方法名。

（2） 理解

理解运算符的优先级与结合性对表达式求值顺序的影响；理解 Math 类函数返回值的类型与意义。

（3） 简单应用

能根据需求构造正确的算术或逻辑表达式；能调用 Math 类完成单一功能的数学计算（如求绝对值、开方）。

（4） 综合应用

能够综合运用多种运算符构造复杂的复合表达式，并结合 Math 类函数解决实际的数学建模问题（如几何计算、物理模拟等），确保运算结果准确无误。

4、本章重点

本章重点在于夯实 Java 编程的基础语法与规范，核心任务是培养学

生编写结构严谨、逻辑清晰的基础程序能力。

内容涵盖：一是代码规范与结构，要求学生养成良好习惯，确保程序结构完整、命名合规、注释规范且无语法错误；

二是变量与数据类型管理，重点掌握变量的正确定义、初始化及作用域划分，深刻理解基本数据类型的匹配机制及 float 与 double 混合运算中的精度问题；

三是运算与表达式处理，要求学生熟练掌握运算符优先级，灵活运用(int)等强制类型转换控制数据精度，并能准确调用 Math 类的数学函数进行平方根、四舍五入等复杂计算，最终确保各类表达式求值结果的准确性。

第 3 章 流程控制语句与异常处理

1、学习目的和要求

通过本章学习，掌握 Java 的流程控制语句使用方法，包括条件判断（if-else、switch）、循环控制（for、while、do-while）等基本语法结构，理解其适用场景及执行流程。掌握 Java 异常处理机制，理解异常的概念、常见异常类型及其处理流程，能够使用 try-catch-finally 结构提高程序健壮性和可维护性。通过实际编程练习，提高分析问题和解决问题的能力，增强程序结构设计能力，提升代码规范性与错误处理意识。

2、考核知识点

（1） 条件分支语句（if-else / switch）

（2） 循环语句（for、while、do-while）

（3） 循环控制语句（break / continue）

（4） 异常类与异常处理机制

（5） 多个 catch 结构的使用

（6） finally 结构的理解与使用

3、考核目标要求

（一）流程控制与逻辑构建

（1） 识记

记忆 if-else、switch、for、while、do-while 等关键字及基本语法格式；熟记循环控制语句（break, continue）的作用。

（2） 理解

理解不同流程控制结构的适用场景（如 switch 的多路选择、while 的未知次数循环）；理解嵌套循环的执行逻辑与算法复杂度。

（3） 简单应用

能够根据需求选择合适的结构（如用 if 判断区间、用 for 遍历数组）；能编写单层循环或简单的分支判断程序，并输出正确结果。

（4） 综合应用

能够综合运用多重循环与复杂条件判断，构建具备一定算法逻辑的程序（如排序、查找、图形打印等），确保逻辑严密且结果完全符合预期。

（二）异常处理与资源管理

（1） 识记

记忆异常处理的语法结构（try-catch-finally）；识别常见的运行时异常类型（如 ArithmeticException, InputMismatchException）。

（2） 理解

理解异常传播机制；理解多个 catch 块的捕获顺序（先子类后父类）；理解 finally 块无论是否发生异常都会执行的特性。

（3） 简单应用

能使用 try-catch 包裹可能出错的代码（如除法、类型转换）；能在 finally 中编写关闭资源（如 Scanner）的代码。

（4） 综合应用

能够根据业务逻辑设计多层异常处理机制，针对不同的异常类型给出精准的错误提示，并在 finally 中确保系统资源的安全释放，防止内存泄漏。

（三）错误诊断与防御性编程

（1） 识记

识记编译期错误（语法错误）与运行期错误的区别；记忆常见的错误提示信息（如 ArrayIndexOutOfBoundsException）。

（2） 理解

理解程序崩溃的根本原因（如除数为零、空指针）；理解防御性编程的重要性，即程序应对非法输入保持宽容。

（3） 简单应用

能读懂编译器报错信息并修复语法错误；能识别导致程序中断的运行时错误位置。

（4） 综合应用

能够开发具备“防呆设计”的程序，利用异常处理机制主动拦截并过滤非法输入（如输入非数字、负数下标等），确保程序在任何异常输入下均能平稳运行或优雅退出。

4、本章重点

本章重点聚焦于程序的流程控制与健壮性保障，旨在培养学生构建逻辑严密、运行稳定的 Java 程序能力。核心内容包括：一是逻辑构建，要求学生熟练掌握 if-else、switch 及各类循环结构，能够依据业务需求灵活组合分支与循环，编写出逻辑清晰、输出准确的算法代码；二是异常处理，重点理解 try-catch-finally 机制，学会针对不同类型的运行时错误（如除数为零、非法输入）设置多路捕获，并合理利用 finally 块进行资源释放，有效防止程序崩溃；三是调试与防御，具备识别语法错误与运行时错误的能力，能够主动运用异常处理技术应对非法输入，提升程序的容错率与稳定性。

第 4 章 类和对象

1、学习目的和要求

本节学习旨在让学生全面掌握 面向对象编程 (OOP) 的核心基础。学生将能够理解类与对象、构造方法的基本概念，并独立定义 Java 类，包括正确使用访问控制符来声明成员变量、构造方法和成员方法。此外，学生将熟练掌握对象的创建与初始化，能够正确调用和重载构造方法。本节还将涵盖方法的定义与重载规则，以及如何通过方法访问和控制成员变量。最后，学生将理解静态成员（变量与方法）的特点与使用场景，并掌握包装类及其自动装箱与拆箱机制在实际开发中的应用

2、考核知识点

（1） 类与对象

类和对象的概念及关系。

类的声明语法，使用访问修饰符（public, private）。

对象的创建与初始化过程。

构造方法的定义、作用及类型（无参、有参）。

（2） 成员方法

成员方法的定义、实现及访问成员变量。

方法重载的定义、语法及调用规则。

方法的访问权限控制（通过修饰符）。

（3） 静态成员

静态变量和方法的定义、用途及特点。

静态成员与实例成员的区别。

静态成员的访问方式（通过类名）。

（4） 包装类

包装类（如 Integer, Double）的作用及常见方法。

自动装箱与拆箱的机制。

字符串与基本类型转换（如 Double.parseDouble）。

（5） 代码规范

合法标识符命名规则（驼峰命名法）。

注释类型（单行、多行、文档注释）及作用。

代码缩进、结构清晰性。

3、考核目标要求

（一）类与对象的基础构建

（1）识记

记忆类（Class）与对象（Object）的定义；熟记构造方法的名称规则（与类名相同）及无返回值特性；识别 private 与 public 关键字。

（2）理解

理解构造方法在对象实例化过程中的作用；理解访问修饰符对成员变量可见性的控制机制（封装性）；理解 this 关键字指代当前对象的含义。

（3）简单应用

能够独立声明一个包含成员变量的类；能编写无参及带参构造方法；能通过 new 关键字创建对象并调用构造方法进行初始化。

（4）综合应用

能够在同一个类中编写多个参数列表不同的构造方法（构造器重载），并能在创建对象时根据传入参数的类型和数量自动匹配对应的构造方法。

（二）方法定义、重载与静态成员

（1） 识记

记忆方法定义的五要素（修饰符、返回值、方法名、参数表、方法体）；记忆 static 关键字的含义；识别方法重载（Overload）的特征（同名不同参）。

（2） 理解

理解方法重载是实现多态的一种形式；理解 static 成员属于类而非特定对象，被所有实例共享；理解实例方法与非实例方法（静态方法）的调用区别。

（3） 简单应用

能正确定义具有特定功能的成员方法；能编写同名但参数不同的方法进行重载；能使用 static 关键字定义类变量或类方法。

（4） 综合应用

能够设计一个包含多个重载方法的工具类（Utility Class），并合理规划 static 成员以存储全局共享数据，实现对象间的数据统计或通信。

（三）高级特性与数据封装机制

（1） 识记

记忆 Java 基本数据类型与其对应包装类（Wrapper Classes）的名称（如 int 与 Integer）；识记自动装箱与拆箱的语法现象。

（2） 理解

理解封装（Encapsulation）的核心思想是将数据隐藏，通过公共方法访问；理解自动装箱是编译器自动将基本类型转换为对象，拆箱则相反。

（3） 简单应用

能将所有成员变量声明为 private，并提供对应的 getter/setter 方法进行访问控制；能在集合类（如 ArrayList）中识别必须使用包装类的情况。

（4） 综合应用

能够综合运用私有化封装、静态计数器（Static Counter）及自动装箱/拆箱机制，实现一个具有安全数据访问、自动编号且能融入Java集合框架的完整实体类（Entity Class）。

4、本章重点

本章重点在于面向对象编程（OOP）的核心基础与封装实现。核心是掌握类与对象的构建，能够定义成员变量与构造方法（含重载），并通过new关键字实例化对象；同时熟练运用方法重载与static关键字，理解实例成员与类级属性的共享区别；最终落实封装思想，利用private保护数据安全，结合自动装箱/拆箱机制，编写出结构严谨、数据安全的Java实体类。

第5章 数组和字符串

1、学习目的和要求

本模块旨在让学生全面掌握 Java 中常用的基本数据结构。学生将深入理解数组（包括一维和二维数组）的概念，学习如何定义、初始化和高效访问数组元素。接着，我们将聚焦于字符串，掌握其定义、初始化，以及字符串与基本数据类型之间的灵活转换，并能熟练运用 String 和 StringBuffer 类进行各种字符串操作。此外，学生还将理解向量（Vector）这种动态数组的特性和用法，包括其元素的添加、删除、修改和查找等核心操作。通过本模块的学习，学生将能够有效选择并运用这些基本数据结构来解决实际编程问题。

2、考核知识点

（1） 一维数组与二维数组的定义与使用
（2） 数组元素访问与遍历方法
（3） 字符串的定义与初始化
（4） 字符串常用方法的使用
（5） StringBuffer 类的使用
（6） 字符串与基本类型的转换
（7） 向量（Vector）的定义
（8） 向量元素的添加与删除
（9） 向量元素的修改与访问

3、考核目标要求

（一）数组的声明、初始化与遍历处理

（1） 识记

记忆数组的定义（连续内存空间）；熟记数组声明的两种语法（type[] name 与 type name[]）；记忆数组长度属性 length。

（2） 理解

理解静态初始化（已知值）与动态初始化（已知长度）的区别；理解数组下标从 0 开始及 ArrayIndexOutOfBoundsException 的成因。

（3） 简单应用

能够正确声明数组并分别使用静态和动态方式进行初始化；能利用 for 循环或增强型 for 循环遍历数组并打印所有元素。

（4） 综合应用

能够利用嵌套循环处理二维数组，或对数组进行排序、查找、最值计算等复杂数据处理操作。

（二）字符串的构建、操作与修改机制

（1） 识记

记忆 String、StringBuffer 的类名；熟记常用方法名如 length()、substring()、equals()、append()。

（2） 理解

理解 String 的不可变性与 StringBuffer 的可变性的区别；理解 equals()与==的差异。

（3） 简单应用

能正确创建字符串对象并进行基本初始化；能调用 substring、equals、charAt 等方法提取信息；能使用 StringBuffer 的 append 和 insert 方法修改字符串内容。

（4） 综合应用

能够针对具体的性能场景（如大量拼接）选择合适的类，并利用字符串类方法，如替换、分割并结合类型转换（parseInt 等）完成复杂的文本解析与格式化任务。

（三） Vector 向量的操作与综合数据管理

（1） 识记

记忆 Vector 类的声明方式；熟记其核心方法：add(), remove(), set(), get(), contains()。

（2） 理解

理解 Vector 与数组的区别(长度可变、可存任意对象)；理解 Vector 作为动态容器在处理不确定数量数据时的优势。

（3） 简单应用

能够创建 Vector 对象；能够使用 add 方法添加元素，使用 get 方法获取元素，使用 remove 删除元素。

（4） 综合应用

能够结合数组与字符串技术，利用 Vector 设计并实现一个具备完整增删改查（CRUD）功能的小型数据管理系统（如学生信息管理），实现数据的存储、检索与更新。

4、本章重点

本章重点聚焦于 Java 中三类核心数据容器的应用与操作。核心内容包括：一是数组，掌握其静态与动态初始化方式，并能结合循环结构高效处理批量同类型数据；二是字符串，深刻理解 String 的不可变性与 StringBuffer 的可变性，熟练运用 equals、substring 等方法进行文本处理与类型转换；三是 Vector 向量，掌握这种动态数组的增删改查(add, remove, set, get）操作，并能将三者结合，构建具备数据存储与处理能力的小型应用程序。

第 6 章 继承与多态

1、学习目的和要求

本模块旨在让学生全面掌握 Java 面向对象编程 (OOP) 的核心概念。我们将学习如何通过 继承（extends） 实现代码复用，并深入理解对象的转型（向上/向下转型）在多态性中的关键作用。

学生将能够区分方法的重载和方法的覆盖，并熟练运用 super 关键字。同时，我们将掌握 final 和 abstract 关键字的语义，了解它们如何限制或强制类的行为。

此外，模块将详细讲解 接口（interface） 的定义与实现，以及其如何支持 Java 的多态性与面向接口编程思想。最终目标是让学生能够合理设计复杂的类层次结构，并运用继承、抽象类和接口来提高程序的

可扩展性和可维护性。

2、考核知识点

本章重点围绕 Java 面向对象的高级特性——继承、多态与抽象展开，旨在让学生掌握类与类之间、类与接口之间的层次结构与扩展机制。核心内容包括：

（1） 继承与构造机制，能够使用 extends 正确声明子类，并通过 super 调用父类的构造方法和成员方法，理解子类对象的初始化流程；

（2） 多态与转型，掌握方法重载（Overload）与覆盖（Override）的规则，理解并运用对象的向上转型与向下转型，体会父类引用指向子类对象所带来的多态性表现；

（3） final 与抽象机制，理解 final 关键字在声明常量、禁止方法覆盖及禁止类继承方面的作用，并能使用 abstract 定义抽象类与抽象方法，由子类完成具体实现；

（4） 接口与面向接口编程，能够定义接口并由类实现多个接口，掌握接口在解耦和系统扩展性方面的优势，初步形成面向接口编程的思想，为构建松耦合、易维护的系统打下基础。

3、考核目标要求

(一) 继承机制与父类成员的复用

（1） 识记

记忆继承的关键字 extends；记忆 Object 类是所有 Java 类的根类；识别父类（Superclass）与子类（Subclass）的概念。

（2） 理解

理解继承机制如何实现代码复用和多态前提；理解子类不能继承父类的构造方法和 private 成员；理解 super 关键字用于访问父类成员的必要性。

（3） 简单应用

能正确编写子类继承父类，并继承其属性和方法；能在子类构造方法中通过 super(...)显式调用父类构造器；能在子类中调用父类的普通方法。

（4） 综合应用

能够设计一个合理的类继承体系（如 Animal-> Dog），在子类中重

写父类方法以增强功能，并通过 super 调用父类被覆盖的方法，实现功能的扩展与复用。

（二）多态性与对象类型转换

（1） 识记

记忆向上转型（自动）与向下转型（强制）的语法格式；记忆 instanceof 关键字的作用。

（2） 理解

理解编译时类型（引用类型）与运行时类型（对象类型）的区别；理解多态产生的三个必要条件（继承、重写、父类引用指向子类对象）；理解运行期绑定（动态绑定）的原理。

（3） 简单应用

能够实现对象的向上转型（父类引用 = new 子类()）；能在明确类型的情况下进行安全的向下转型；能使用 instanceof 判断对象类型以避免类型转换异常。

（4） 综合应用

能够设计并使用多态机制，通过父类引用统一调用不同子类的重写方法，实现“一个接口，多种实现”的业务逻辑，提高代码的通用性和可维护性。

（三）方法重载、覆盖与 final 控制

（1） 识记

记忆方法重载（Overload）的特征（同名不同参，与返回值无关）；记忆方法覆盖（Override）的注解@Override；记忆 final 修饰变量、方法、类的效果。

（2） 理解

理解重载发生在同一类中，覆盖发生在父子类间；理解方法覆盖的“两同两小一大”规则；理解 final 关键字保证不可变性的设计意图。

（3） 简单应用

能在同一类中编写多个构造方法或功能相似但参数不同的方法（重载）；能正确覆盖父类的非 final 方法；能使用 final 定义常量或禁止类被继承。

（4） 综合应用

能够在复杂系统中灵活运用重载提供便捷的 API 入口，同时利用覆盖定制子类特有行为，并合理使用 final 保护核心代码不被篡改，确保系统稳定性。

（四）抽象类、接口与面向接口编程

（1） 识记

记忆 abstract 关键字用于定义抽象类和抽象方法；记忆接口定义使用 interface，实现使用 implements；记忆接口中的成员默认修饰符（public static final 和 public abstract）。

（2） 理解

理解抽象类作为模板的作用（部分实现）；理解接口作为规范的作用（完全抽象）；理解接口弥补了 Java 单继承的不足，实现了多重继承的效果；理解面向接口编程带来的松耦合优势。

（3） 简单应用

能定义包含抽象方法的抽象类，并由子类继承实现；能定义接口及其抽象方法；能让类实现单个或多个接口，并重写接口中的所有方法。

（4） 综合应用

能够基于接口进行系统设计，定义统一的标准接口，让不同的业务类实现该接口；能够通过接口类型声明变量，实现模块间的低耦合调用，体现程序设计的扩展性和灵活性。

4、本章重点

本章重点聚焦于 Java 面向对象编程的高级特性——继承、多态与抽象。核心内容包括：

（1）继承与复用，掌握使用 extends 建立类层次结构，理解子类对父类属性与方法的继承机制，并能通过 super 关键字灵活调用父类构造器与成员；

（2）多态与转型，深入理解方法重载（Overload）与覆盖（Override）的区别，掌握对象的向上转型与向下转型，理解编译时类型与运行时类型的差异，体会运行期绑定带来的多态性表现；

（3） final 与抽象机制，理解 final 关键字在定义常量、禁止方法覆盖及禁止类继承方面的控制作用，并能使用 abstract 定义抽象类与抽象方法，由子类完成具体实现；

（4） 接口与面向接口编程，掌握接口的定义与实现（包括多实现），理解接口作为“规范”在解耦和系统扩展性方面的优势，初步形成面向接口编程的思想，构建松耦合、易维护的系统结构。

第 7 章 数据流与文件操作

1、学习目的和要求

本章旨在让学生全面掌握 Java 中的数据流（输入/输出流）概念及其应用。你将理解字节流与字符流的区别及适用场景，并熟练运用常见的流类进行数据读写。此外，你还会掌握缓冲流、过滤流和串接流的使用，以提升 I/O 操作的效率和灵活性。通过学习 File 类和 RandomAccessFile 类，你将能够对文件进行基本操作，最终具备编写实现文件读取、写入、打开、保存等功能的文件系统编程能力。

2、考核知识点

本章重点围绕 Java 输入输出（I/O）流体系与文件系统操作 展开，旨在让学生掌握数据在程序与外存之间传输的各种机制。核心内容包括：

（1）流的分类与基础操作，理解字节流（InputStream/OutputStream）与字符流（Reader/Writer）的区别，掌握 FileInputStream、FileReader 等基本流的使用；

（2）流的高级应用与包装，熟练运用缓冲流（BufferedStream）提高效率，掌握过滤流、序列流（SequenceInputStream）的串接技巧，以及 RandomAccessFile 对文件的随机读写控制；

（3）文件管理与交互，掌握 File 类对文件路径、属性及目录的操作，结合文件对话框实现可视化文件选择；

（4） 综合实战，能够将上述知识点融会贯通，独立编写具备文件复制、内容追加、随机修改等功能的综合性文件操作程序。

3、考核目标要求

（一）I/O 流模型与基础文件操作

（1） 识记

记忆“输入流/输出流”、“字节流/字符流”的基本概念；记住 FileInputStream 和 FileOutputStream 的类名。

（2） 理解

理解流是用来传输数据的管道；理解使用缓冲流（Buffered

Stream)比单纯使用字节流读取速度更快的原因(减少硬盘读写次数)。

（3） 简单应用

能编写代码使用字节流读取一个文本文件的内容并打印到控制台；能使用缓冲流复制一个简单的图片或小文件。

（4） 综合应用

能够编写程序，将一个指定文件复制到另一个指定目录，并确保使用 try-catch 处理文件不存在等常见异常。

（二）文件属性与随机读写

（1） 识记

记忆 File 类用于判断文件是否存在的方法名（如 exists()）；记忆 RandomAccessFile 的读写模式（"r", "rw"）。

（2） 理解

理解 File 类只是用来描述文件路径和属性的，不能直接读写文件内容；理解 seek()方法是用来移动文件内部的“光标”(指针)的。

（3） 简单应用

能创建一个 File 对象，并判断该文件是否存在；能使用 RandomAccessFile 在文件末尾追加一行新文字。

（4） 综合应用

能够编写程序，显示指定文件的大小、最后修改时间等属性，并实现跳过文件前 N 个字节开始读取的功能。

（三）字符流与对话框交互

（1） 识记

记忆 FileReader 和 BufferedWriter 用于读写文本；记忆 JFileChooser 是用来弹出文件选择窗口的。

（2） 理解

理解字符流适合处理人类能看懂的文字（如 .txt, .java），字节流适合处理图片、视频等二进制文件；理解对话框只是获取路径，真正的读写还要靠流。

（3） 简单应用

能使用字符流读取一个纯文本文件，并按行输出；能使用 JFileChooser 弹出一个窗口让用户选择一个文件。

（4） 综合应用

能够编写程序，点击按钮弹出文件选择框，选中一个文本文件后，将其内容读取并显示在界面的文本框（TextArea）中。

4、本章重点

本章重点在于掌握 Java 文件读写的基础操作与工具使用。核心内容包括：

（1）流的基础应用，理解输入/输出流及字节/字符流的区别，能使用字节流（如 FileInputStream）完成文件的复制，并了解缓冲流（BufferedInputStream）对效率的提升作用；

（2）文件属性与随机访问，学会使用 File 类判断文件是否存在、获取文件大小，并了解如何通过 RandomAccessFile 对文件进行简单的追加或指定位置读取；

（3）文本处理与界面交互，熟练使用字符流（如 FileReader）读写文本文件，并能结合文件对话框（JFileChooser）实现“选择文件—读取内容”的完整流程，为开发简单的文件管理功能打下基础。

第 8 章 GUI 图形界面设计

1、学习目的和要求

掌握 AWT 及 Swing 的特点，理解其在 Java GUI 开发中的作用，了解 AWT 和 Swing 中类的层次结构并掌握基本的框架窗口创建方法，能够正确创建简单的框架窗口并使用面板通过内容窗格添加组件，熟练使用标签和按钮组件并处理按钮事件，掌握 FlowLayout、BorderLayout、GridLayout、CardLayout 及 BoxLayout 等布局管理器的概念及使用方法，完成包括嵌套布局在内的界面设计，理解事件处理机制并掌握委托事件处理模型，熟练响应鼠标和键盘事件，掌握绘图基础，能够显示不同字体和颜色的文字并绘制各种基本几何形状图形并进行着色。

2、考核知识点

（1） WT 和 Swing 特点：理解 AWT 和 Swing 在 Java GUI 开发中的作用及特点。

（2） 类层次结构：掌握 AWT 和 Swing 中类的层次结构及基本框架窗口创建方法。

（3） 窗口与面板：能够正确创建简单的框架窗口，使用面板并通过内

容窗格添加组件。

（4） 组件使用：熟练使用标签和按钮组件，掌握按钮事件的处理。

（5） 布局管理器：掌握 FlowLayout、BorderLayout、GridLayout、CardLayout 及 BoxLayout 的概念和使用方法，完成包括嵌套布局在内的界面设计。

（6） 事件处理：理解事件处理机制，掌握委托事件处理模型，熟练响应鼠标和键盘事件。

（7） 绘图基础：掌握绘图基础，能够显示不同字体和颜色的文字，绘制各种基本几何形状图形并进行着色。

3、考核目标要求

（一）GUI 界面构建与布局管理

（1） 识记

记忆顶级容器（JFrame）与中间容器（JPanel）的区别；记忆常用布局管理器（FlowLayout, BorderLayout, GridLayout）的名称。

（2） 理解

理解布局管理器负责组件的位置与大小；理解通过“面板嵌套面板（Panel in Panel）”可以实现复杂的界面结构。

（3） 简单应用

能创建 JFrame 窗口并设置关闭操作；能在窗口中添加 JLabel 和 JButton；能为容器设置一个单一的布局管理器。

（4） 综合应用

能够设计并实现包含至少两种布局管理器的界面（例如：BorderLayout 管理整体，Center 区域嵌套 GridLayout 的按钮组），界面整齐美观。

（二）事件监听与交互响应

（1） 识记

记忆事件源、监听器（Listener）与事件对象的概念；记忆 ActionEvent、MouseEvent 等常用事件类。

（2） 理解

理解“委托事件模型”：组件产生事件，监听器负责处理；理解匿名内部类或 Lambda 表达式作为监听器的实现方式。

（3） 简单应用

能为按钮添加点击事件（ActionListener），点击后在控制台或标签上显示文字；能处理简单的键盘输入事件。

（4） 综合应用

能够编写代码处理鼠标点击坐标或键盘按键响应，并在界面上实时反馈（如移动组件位置），同时确保代码结构清晰、命名规范且有注释。

（三）图形绘制与程序演示

（1） 识记

记忆绘图方法（paintComponent）的存在；记忆 Graphics 类中绘制图形（drawLine, drawRect, fillOval）的方法名。

（2） 理解

理解 Graphics 对象相当于一支画笔；理解在 Swing 中绘图应重写 JPanel 的 paintComponent 方法，而不是直接在 JFrame 中绘制。

（3） 简单应用

能在 JPanel 上画出直线、矩形和圆形;能设置画笔的颜色（setColor）和字体（setFont）。

（4） 综合应用

能够开发一款简单的绘图板程序，展示多种颜色和形状的组合，并提交完整的运行截图或视频，展示程序在正常操作下的表现及简单的异常处理（如空指针防护）。

4、本章重点

本章重点在于掌握 Java 图形用户界面（GUI）的基础开发能力。核心内容包括：

（1）界面搭建与布局，学会使用 JFrame、JPanel 等容器构建窗口，通过布局管理器（如 FlowLayout、BorderLayout）及嵌套布局技术，实现按钮、标签等组件的有序排列；

（2）交互与绘图，理解事件监听机制，能为按钮点击、鼠标操作添加响应代码，并能利用 Graphics 画笔在窗体上绘制线条、矩形、圆形等几何图形，设置不同的颜色与字体；

（3）工程规范，能够编写结构清晰、命名规范的完整代码，并通过截图或运行结果展示程序功能，具备基础的桌面应用开发体验。

第 9 章 熟悉 Java 程序开发环境

1、学习目的和要求

本实验旨在通过设计和实现一个“用户信息注册”窗口，使学生掌握 Java Swing 图形用户界面（GUI）开发的基本方法，熟悉常用组件（如窗口、菜单、文本框、下拉框、列表、文本区等）的使用与布局管理，理解事件驱动编程模型，能够实现基本的人机交互功能，提升图形界面程序的设计能力和编程规范意识，增强学生综合运用所学知识解决实际问题的能力。

2、考核知识点

（1） 掌握 GUI 窗口创建与组件添加方法：能够使用 JFrame 搭建基本窗口，设置窗口属性，并向界面中添加各类控件。

（2） 熟悉常用 Swing 组件的使用：能够正确使用 JTextField（文本框）、JComboBox（下拉框）、JList（列表）和 JTextArea（文本区）等进行信息采集，并实现多选、滚动等交互功能。

（3） 掌握菜单栏构建与事件处理机制：能够使用 JMenuBar 构建完整的菜单系统，熟练为按钮和菜单项添加 ActionListener 以实现交互操作。

（4） 理解布局管理器的应用场景与组合方式：能根据组件特点合理选择并组合 BorderLayout、GridLayout、FlowLayout 等布局方式，使界面美观、结构清晰。

（5） 具备信息处理与交互反馈能力：能够实现“注册”信息的收集与展示、“清空”操作的内容重置、“退出”与“关于”等功能的弹窗提示。

（6） 具备良好的程序规范性与可维护性：代码结构合理、命名规范、注释清晰，界面无报错，交互逻辑流畅，具备良好的可读性与稳定性。

3、考核目标要求

（一）Swing 容器与基础组件构建

（1） 识记

记忆顶级容器 JFrame 的常用方法（如 setTitle, setSize, setVisible）；记忆菜单栏（JMenuBar）、菜单（JMenu）和菜单项（JMenuItem）的类名。

（2） 理解

理解 Swing 的容器层次结构（顶层容器包含内容面板）；理解菜单栏必须依附于 JFrame 才能显示；理解不同组件（如 JTextField 用于单行，JTextArea 用于多行）的适用场景。

（3） 简单应用

能创建 JFrame 主窗口并设置基本属性；能创建“文件”和“帮助”菜单，并添加“退出”、“关于”菜单项；能在界面上添加文本框和下拉框。

（4） 综合应用

能够构建一个包含完整菜单系统和多种输入组件（文本框、密码框、下拉框）的注册信息录入界面，界面结构清晰，组件摆放合理。

（二）复杂组件交互与布局美化

（1） 识记

记忆 JScrollPane（滚动面板）的构造方法；记忆 JList 的设置方法；记忆常用布局管理器（FlowLayout, BorderLayout, GridLayout）的方位常量。

（2） 理解

理解 JScrollPane 作为“视口”包裹 JTextArea 的机制；理解 JList 的选择模式（单选/多选）对数据获取的影响；理解布局管理器如何控制组件的排列与拉伸。

（3） 简单应用

能将 JTextArea 嵌入 JScrollPane 实现滚动功能；能将 JList 设置为多选模式；能使用一种或多种布局管理器排列界面组件。

（4） 综合应用

能够设计复杂的表单界面，结合使用网格布局和边框布局，实现包含“兴趣爱好”多选列表和用户简介多行输入的精美界面，并确保组件随窗口大小变化自适应。

（三）事件监听、数据校验与弹窗反馈

（1） 识记

记忆动作监听器（ActionListener）的接口定义；记忆 JOptionPane 显示消息框的静态方法（如 showMessageDialog）。

（2） 理解

理解事件源（按钮/菜单）与监听器之间的绑定关系；理解非空校验的逻辑流程（先获取文本，再判断是否为空）；理解弹窗作为信息反馈给用户的作用。

（3） 简单应用

能为“注册”和“清空”按钮添加点击事件；能使用 JOptionPane 弹出简单的提示信息；能获取下拉框或多选列表中的选中值。

（4） 综合应用

能够编写完整的业务逻辑：点击“注册”时检查姓名是否为空（为空则弹窗警告），无误则汇总所有表单数据并弹窗显示；点击“清空”时重置所有组件状态；点击“退出”菜单能关闭程序。

4、本章重点

本章重点在于掌握 Java Swing 图形用户界面（GUI）的综合构建与交互逻辑。核心内容包括：

（1）界面搭建与菜单设计，熟练使用 JFrame 创建主窗口，并构建包含 JMenuBar、菜单及菜单项的完整导航系统；

（2）组件应用与布局美化，灵活运用 JTextField、JComboBox、JList 及 JTextArea 等常用控件，掌握利用 JScrollPane 实现滚动区域及设置列表多选模式的技巧，并能结合 BorderLayout、GridLayout 等布局管理器设计出结构清晰、分布合理的界面；

（3）事件响应与数据校验，理解 ActionListener 监听机制，能为按钮和菜单添加点击响应，利用 JOptionPane 进行信息反馈，并实现“清空”重置功能及对输入内容的非空校验，完成具备基础业务逻辑的交互式桌面应用。

三、题型举例（考试时间为 150 分钟）（题型仅作参考，实际命题时不受此限）

（一）单项选择题（在备选答案中只有一个是正确的，将其选出并把它的题号写在题后括号内） 2 分×15 题=30 分

1. Java 程序开发需要安装的首要组件是？（ B ）。

A. Eclipse B. JDK C. JRE D. IntelliJ IDEA

（二）多项选择题（在备选答案中有二至五个是正确的，将其全部选出，

并把它们的题号写在题后括号内。错选或漏选均不给分）　2 分×5 题=10 分

1．方法重载的规则包括哪些？（　ABC　）。

A．方法名相同　B.参数数量不同　C.参数类型不同　D.返回类型不同

（三）判断题　2 分×10 题=20 分

1．StringBuffer 的 append() 方法返回新的 StringBuffer 对象。（ B ）

A.正确　B.错误

（四）填空题　2 分×5 题=10 分

1．"String s = ""apple,banana,orange"";

String[] fruits = s.split("","");

fruits[1] 的值是：__banana "。

（五）简答题　10 分×3 题=30 分

1．已提供的窗体代码基础上，补充以下功能：

在窗体中添加一个　文本框　（用于用户输入）。

添加一个　按钮（标题为显示内容）　。

```
import javax.swing.*;
import java.awt.*;
public class MainWindow extends JFrame {
    public MainWindow() {
        setTitle("补充题");
        setSize(300, 200);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new FlowLayout()); // 使用流式布局
        //  在此补充代码（添加文本框和按钮）
    }
    public static void main(String[] args) {
            new MainWindow().setVisible(true);
    }
}
```

答案:

```
JTextField textField = new JTextField(20);
JButton button = new JButton("显示内容");
add(textField);
add(button);
```