# Java 面向对象程序设计与实践应用

范洪辉　常玉慧　刘天霁　编著



電子工業出版社
Publishing House of Electronics Industry
北京 · BEIJING

## 内 容 简 介

本书是一本以产教融合为核心、以企业真实项目为载体的 Java 编程实战教材。本书共 12 章，以“云医院”系统开发为主线，将 Java 语言的核心知识点融入实际应用场景，通过理论与实践相结合的方式，帮助读者系统掌握 Java 编程技能，并培养其解决复杂工程问题的能力。本书从 Java 基础语法入手，逐步深入讲解程序流程控制、数组与字符串、面向对象编程、继承与多态、异常处理、集合框架、文件操作、多线程与并发编程等核心内容，并结合 Java 新特性拓展技术视野。本书内容由浅入深，从基础语法到综合项目开发循序渐进，配合配套实验指导，确保读者能够真正学以致用。



**图书在版编目（CIP）数据**

Java 面向对象程序设计与实践应用 / 范洪辉，常玉慧，刘天霁编著. -- 北京 : 电子工业出版社，2025. 7.
ISBN 978-7-121-50448-8

Ⅰ. TP312.8

中国国家版本馆 CIP 数据核字第 2025ZX3251 号

责任编辑：石会敏　　文字编辑：曹　旭
印　　刷：
装　　订：
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编 100036
开　　本：787×1 092　1/16　印张：15.75　字数：403 千字
版　　次：2025 年 7 月第 1 版
印　　次：2025 年 7 月第 1 次印刷
定　　价：59.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888，88258888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

本书咨询联系方式：shhm@phei.com.cn。

# 前　言

在漫长的人类文明史中，知识与智慧的传承始终是推动社会进步的重要力量。从古老的象形文字到现代的电子编码，信息的记录与传播方式经历了翻天覆地的变化。而今，我们站在数字化时代的潮头，编程已成为连接现实与虚拟、创造无限可能的桥梁。Java，这门诞生于 20 世纪 90 年代的编程语言，凭借其跨平台、面向对象、安全性高等特性，在全球范围内赢得了广泛的认可与应用。本书就是一部引领读者深入 Java 编程世界的指南，而云医院则是我们精心挑选的实践舞台，用以展示 Java 技术的强大魅力与广泛应用。

## 一、Java：编程世界的璀璨明珠

Java 自其诞生之日起，就注定了它的不平凡。作为一门面向对象的编程语言，Java 不仅继承了 C++等的优点，更在跨平台性、安全性等方面实现了质的飞跃。它允许开发者编写的程序在任何支持 Java 的平台上运行，无须进行烦琐的移植工作，极大地降低了开发成本，提高了程序的复用性。同时，Java 的安全机制也为网络应用提供了坚实的保障，使得 Java 成为企业级应用开发的首选语言之一。

## 二、Java 与生活中的产业

在信息技术飞速发展的今天，软件开发已经成为推动社会进步的重要力量。互联网、大数据、人工智能等技术的迅猛发展，使得各行各业对高质量软件的需求愈发迫切。

在医疗行业，随着人们对健康服务需求的不断提升，传统医疗模式逐渐向智慧医疗转变。通过信息化手段，提高医疗服务效率，改善患者体验，已成为医疗行业的重要发展方向。通过互联网和云计算技术，在线医疗平台能够为患者提供便捷的医疗服务，减少看病排队时间，提高医疗资源利用率。

在这样的时代与市场背景下，“云医院”作为一个领先的在线医疗平台应运而生。“云医院”项目利用云计算和大数据技术，提供了一个集成多种医疗服务的云平台，旨在为患者提供高效、便捷的医疗服务。通过“云医院”，患者可以实现在线预约挂号、远程问诊、医药购买、健康档案管理、医疗费用在线支付等操作，不仅提高了患者的满意度，也为医疗机构带来了显著的效益，减少了人工成本，提高了工作效率。

## 三、Java 的产教融合教学

本书采用源自企业真实案例的“云医院”作为教学主线，旨在通过产教融合的思想，将理论与实践紧密结合，为读者呈现一个生动、实用的 Java 编程学习之旅。

本书通过精心设计的“云医院”案例，将 Java 的各大知识点融入其中，从基础知识到进阶技能，每一步都紧密围绕实际应用的需求展开。在学习过程中，读者将仿佛置身于一个真实的开发环境，跟随作者的引导，逐步掌握 Java 编程的精髓。

当读者完成本书的学习后，不仅可以熟练掌握 Java 的基础语法及其使用规范，更能够运用所学，独立解决“云医院”案例中所涉及的各种实际问题。同时，读者能够了解系统架构原理，提升软件设计及数据库应用能力。这种基于产教融合的教学模式，不仅让读者在学习过程中获得更加直观、深刻的理解，更为其未来的职业发展奠定了坚实的基础。

四、教材结构

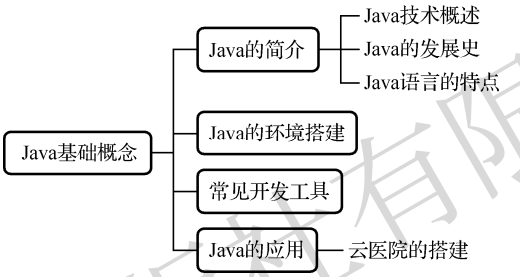
（一）章节概览与思维导图

本书分为 12 章，每章都围绕 Java 编程的一个核心领域展开，同时结合云医院系统的构建过程，将理论知识与实践应用紧密结合。接下来将通过思维导图的形式介绍每一章的知识点模块。

**1．第 1 章　初识 Java——走进编程的新世界**

在本章中，我们将带你认识 Java，了解它的历史背景、语言特点以及在企业级应用中的重要性。同时，我们还会探讨 Java 在医疗信息化领域的应用现状，特别是在云医院系统的构建中 Java 所扮演的角色。通过本章的学习，你将对 Java 有一个初步的认识，为后续的学习打下基础。

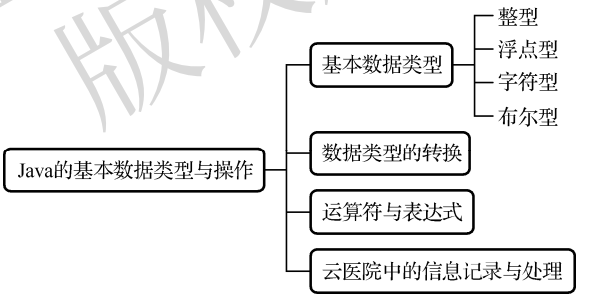
**思维导图：**




**2．第 2 章　Java 的基本数据类型与操作——构建云医院信息的基石**

在本章中，我们将学习 Java 的基本数据类型和操作，包括整型、浮点型、字符型等数据类型以及它们之间的转换和运算。这些基础知识将为你后续构建云医院系统中的患者信息记录、处理等功能提供坚实的支撑。

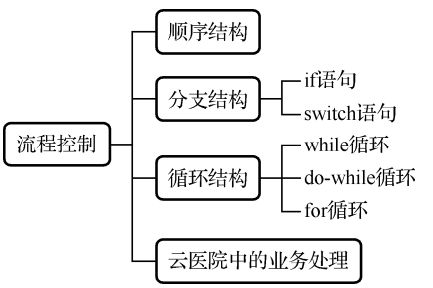
**思维导图：**




**3．第 3 章　流程控制——让云医院系统更加“聪明”**

在本章中，我们将学习 Java 的流程控制语句，包括顺序结构、分支结构和循环结构。通过掌握这些语句，你将能够编写出更加复杂、更加智能的程序，实现云医院系统中的业务逻辑处理，如患者挂号、科室分配等功能。

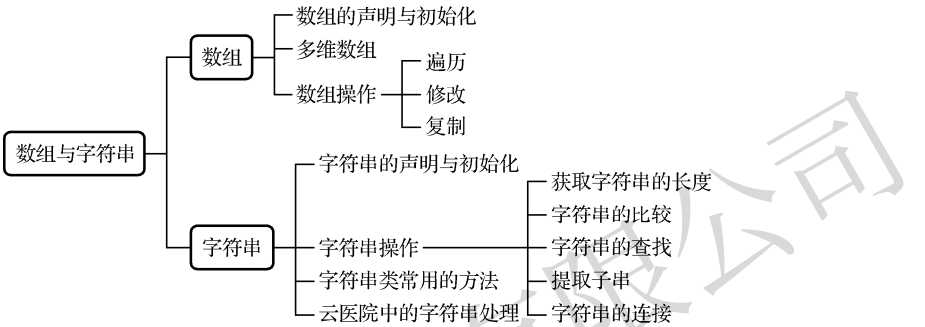
**思维导图：**



**4．第 4 章　数组与字符串——整理与阅读云医院信息的利器**

在本章中，我们将学习 Java 的数组和字符串处理。数组能够高效管理大量患者信息（如编号、年龄、科室分配等结构化数据），而字符串用于处理患者的体温记录、病历文本等非结构化信息，二者的结合使云医院系统更加完善。

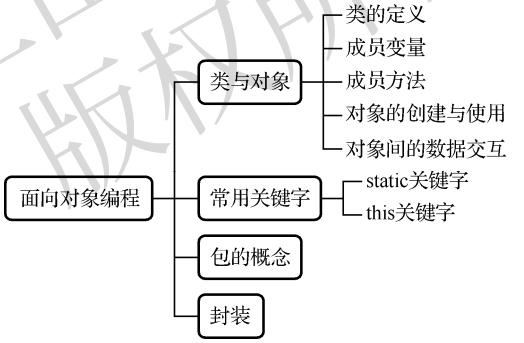
**思维导图：**




**5．第 5 章　面向对象编程——构建云医院系统的核心模块**

在本章中，我们将学习面向对象编程的基本概念，包括类、对象、常见关键字、包的概念、封装等。通过本章的学习，你将学会如何通过创建自己的类来定义云医院系统中的功能模块，如患者管理、医生管理等。你还将学会如何通过封装来对这些模块进行安全性处理，使你的系统更加可靠与稳定。

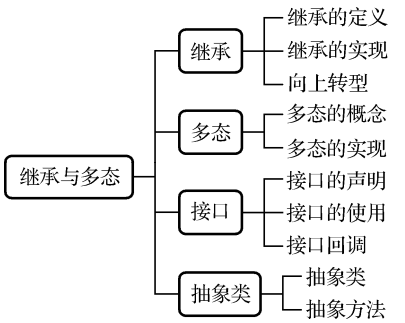
**思维导图：**




**6．第 6 章　继承与多态——云医院系统的扩展与维护**

在本章中，我们将继续深入学习类的相关特性，包括类的继承与多态、接口、抽象类等。通过本章的学习，你将学会如何使用这些特性来进一步扩展和优化你的云医院系统。例如，你可以通过接口来定义医生、护士等角色的共同行为，通过内部类来实现一些辅助功能等。

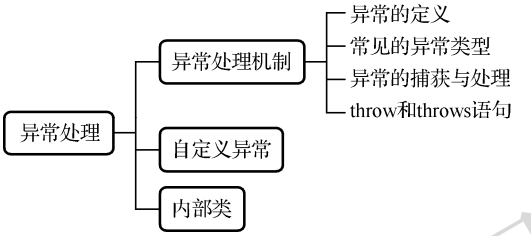
**思维导图：**



**7．第 7 章　异常处理——让云医院系统更加稳健**

在 Java 编程中，异常处理是非常重要的。通过捕获和处理异常，你可以避免程序因为一些小问题而崩溃。在本章中，你将学会如何捕获和处理异常，以及如何编写健壮的代码来应对潜在的错误情况。这些技能将使你的云医院系统更加稳定可靠。

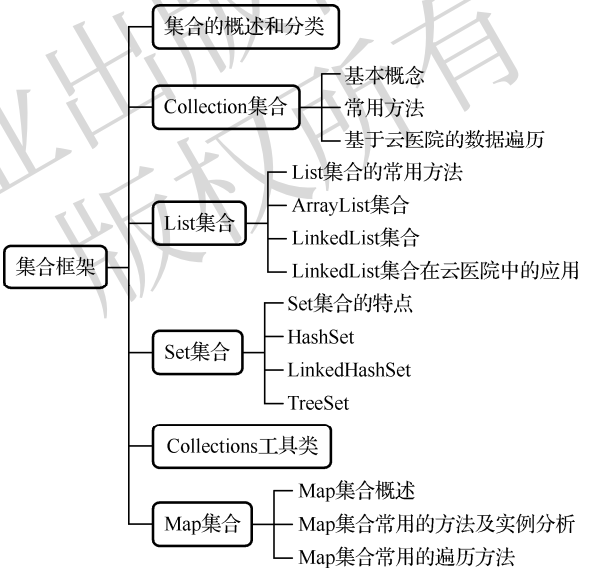
**思维导图：**




**8．第 8 章　集合框架——云医院数据的高效管理**

Java 的集合框架提供了一系列用于管理大量数据的接口和实现类。在本章中，你将学会如何使用这些集合来高效地管理云医院中的患者信息列表、医生排班表等数据。这些技能将使你的云医院系统更加高效和易于维护。

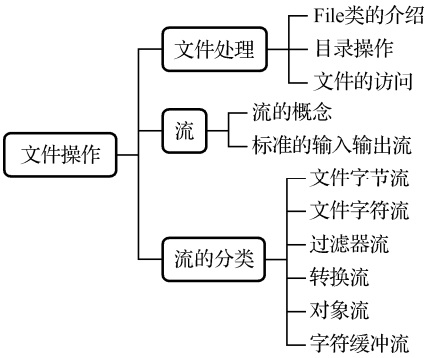
**思维导图：**




**9．第 9 章　文件操作——云医院数据的备份与恢复**

在 Java 中，你可以通过读取和写入文件来处理字节流和字符流。在本章中，你将学会如何使用 Java 的 API 手册来备份和恢复云医院系统中的数据。这些技能将使你的系统更加安全，并能够应对数据丢失或损坏等紧急情况。

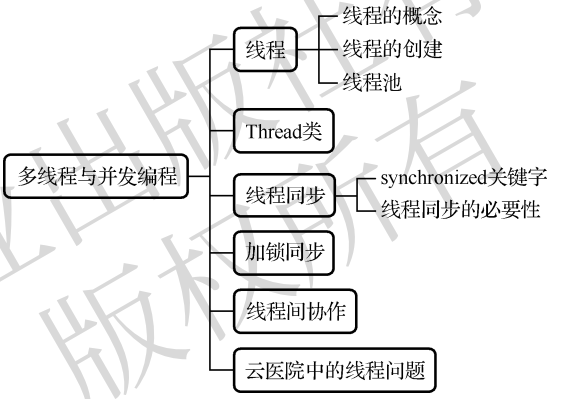
**思维导图：**




**10．第 10 章　多线程与并发编程——提升云医院系统的响应速度**

多线程与并发编程是 Java 编程中的重要组成部分。在本章中，你将学会如何创建和管理线程，如何保证线程安全，以及如何使用线程。

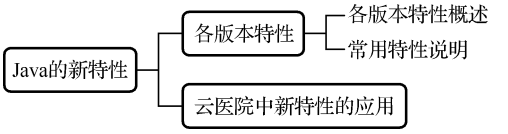
**思维导图：**




**11．第 11 章　Java 的新特性**

Java 的新特性是 Java 在发展的历程中，版本不断更迭所产生的新的技术与方法。在本章中，你将了解 Java 不同版本所提供的特色技术。

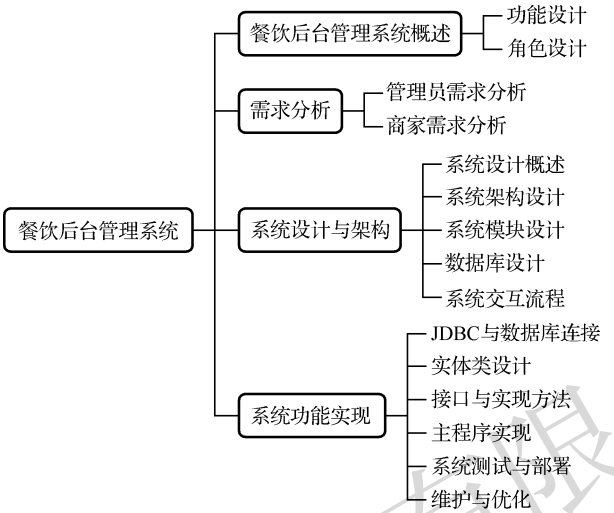
**思维导图：**




**12．第 12 章　综合项目——餐饮后台管理系统**

Java SE 凭借其面向对象、跨平台及多线程特性，为餐饮后台管理系统提供了稳定高效的核心能力，支持订单处理、商品管理及权限控制等关键业务逻辑的模块化开发。在本章中，你将学会如何设计并编码实现简易的餐饮后台管理系统。

**思维导图：**




## （二）任务场景表

| 章 | 知识点 | 学习目标和场景描述 | 云医院应用实例 |
|---|---|---|---|
| 第 1 章 | Java 的发展史、语言特点、在企业级应用中的重要性 | 了解 Java 的发展史、语言特点及其在企业级应用中的核心价值，同时探索 Java 在医疗信息化领域的广泛应用，特别是云医院系统的构建与运营 | （1）介绍 Java 在医疗信息化中的应用；<br>（2）介绍 Java 的环境搭建与工具的安装使用 |
| 第 2 章 | Java 的基本数据类型与操作 | 学习 Java 的基本数据类型与操作，掌握如何在 Java 中处理和转换患者信息，如姓名、年龄、病情等，为后续的医疗信息处理打下基础 | （1）云医院系统常见数据的声明定义；<br>（2）药品信息参数的常量定义与体温变化的变量声明；<br>（3）挂号费用的计算与运算符的使用 |
| 第 3 章 | 流程控制 | 掌握 Java 的流程控制语句，如顺序、分支和循环，以实现云医院中的复杂业务逻辑，如根据病情分配科室、管理医院任务等 | （1）云医院中挂号流程的处理；<br>（2）患者数据的自动化监测与信息采集 |
| 第 4 章 | 数组与字符串 | 学习数组和字符串的使用，以高效地管理患者信息，如体温记录、病历信息等，并处理医疗文本数据 | （1）患者信息的存储与管理；<br>（2）复杂信息——病历信息的处理 |
| 第 5 章 | 面向对象编程 | 掌握面向对象编程的概念，如类、对象、包、封装，设计和实现云医院中的各种功能模块，提高系统的可维护性和可扩展性 | （1）医院系统中监测报告单的具化；<br>（2）如何使用封装确保患者信息安全 |
| 第 6 章 | 继承与多态 | 通过继承与多态机制，实现代码复用，降低开发成本，同时提高云医院系统的灵活性和可扩展性，以应对不断变化的医疗业务需求 | （1）病因与病理库的继承关系；<br>（2）遗传中的多态性 |
| 第 7 章 | 异常处理 | 学习 Java 的异常处理机制，以捕获和处理云医院系统运行过程中可能出现的异常，确保系统的稳定可靠，改善用户体验 | 云医院中内存溢出与信息存储异常的处理 |

续表

| 章 | 知识点 | 学习目标和场景描述 | 云医院应用实例 |
| --- | --- | --- | --- |
| 第 8 章 | 集合框架 | 掌握 Java 集合框架的使用，以高效地管理云医院系统中的大量数据，如患者信息列表、医生排班表等，提高数据处理效率 | 患者就诊挂号排序与患者信息的遍历筛选 |
| 第 9 章 | 文件操作 | 学习 Java 的文件操作，以实现云医院系统中数据的备份和恢复功能，确保数据的安全性和可靠性 | 医护信息的读取与录入 |
| 第 10 章 | 多线程与并发编程 | 掌握 Java 的多线程与并发编程技术，以实现云医院系统同时处理多个请求的能力，如多个患者同时挂号、多个医生同时查看病历等，提升系统响应速度和用户体验 | 线程对云医院系统的优化方案 |
| 第 11 章 | Java 的新特性 | 学习 Java 的新特性，如 Lambda 表达式、Stream API 和模块化等，以更简洁、高效的方式编写代码，为云医院系统提供更丰富的功能支持 | Java 的新特性对云医院系统的支持 |
| 第 12 章 | 综合项目 | 综合应用实例，结合现实生活中餐饮平台的设计与实现，模拟完成餐饮后台管理系统的设计 | 无 |

## 五、结束语

通过对本书的学习，你将从基础到进阶，从理论到实践，逐步掌握 Java 编程的精髓。每一章节的学习都将为你打开一扇新的大门，让你在 Java 的世界里畅游得更加自如。最终，当你完成本书的学习时，你将能够熟练地运用 Java 技术模拟构建云医院系统的部分模块，对医疗信息的数字化管理有深刻的认识。同时，本书将外卖点餐系统脱敏后进行教学案例转化，分解为课后实验供你进行练习，从而进一步提升实操能力。这不仅能为你的职业发展增添竞争力，更能让你在数字化时代成为一位真正的技术先锋。

让我们一起用 Java 编程创造更加美好的未来！

# 目　录


第 1 章　初识 Java——走进编程的新世界 …… 1
1.1　Java 简介与云医院系统开发基础 …… 1
1.1.1　Java 技术概述 …… 1
1.1.2　Java 的发展史 …… 3
1.1.3　Java 语言的特点 …… 3
1.1.4　云医院系统对 Java 的需求 …… 5
1.2　Java 的开发和执行环境 …… 6
1.2.1　JDK 的下载与安装 …… 6
1.2.2　Java 的环境搭建 …… 7
1.2.3　Java 项目结构初识 …… 9
1.3　Java 开发工具 IntelliJ IDEA …… 10
1.3.1　IntelliJ IDEA 简介 …… 10
1.3.2　IntelliJ IDEA 开发 Java 程序 …… 13
1.4　云医院项目的初步搭建 …… 14
1.4.1　项目技术选型 …… 14
1.4.2　项目结构规划 …… 16
小结 …… 17
习题 …… 17
第 2 章　Java 的基本数据类型与操作——构建云医院信息的基石 …… 18
2.1　数据的基石：Java 的基本数据类型 …… 18
2.1.1　数据的声明与定义 …… 18
2.1.2　标识符与关键字：数据声明的规范 …… 19
2.1.3　数据类型的分类 …… 20
2.1.4　数据类型的转换 …… 23
2.1.5　数据类型在云医院中与患者信息的结合 …… 25
2.1.6　Java 的代码结构 …… 26
2.2　数据值的设定：云医院中的常量与变量 …… 27
2.2.1　常量与药品信息参数设定 …… 28
2.2.2　变量与患者体温特征的变化 …… 28
2.3　运算符与表达式：云医院中的逻辑处理 …… 29
2.3.1　运算符：挂号费用计算 …… 29
2.3.2　表达式：诊断信息确认 …… 34
2.4　注释 …… 35
小结 …… 37
习题 …… 38

实验一　餐饮后台管理系统——基本数据类型和运算符 ……39
实验目标 ……39
实验任务 ……39
**第 3 章　流程控制——让云医院系统更加“聪明”** ……41
3.1　顺序结构：处理挂号流程 ……41
3.2　分支结构：决策在云医院中的应用 ……42
3.2.1　If 应对预约挂号问题 ……42
3.2.2　switch 与患者挂号科室的选择 ……44
3.3　循环结构：自动化处理云医院任务 ……46
3.3.1　while 与定时检查患者体温 ……46
3.3.2　do-while 与 while 的差异 ……47
3.3.3　for 循环与多位患者信息采集 ……48
3.3.4　患者监测中的 break 与 continue ……49
小结 ……51
习题 ……51
实验二　餐饮后台管理系统——控制流语句 ……51
实验目标 ……51
实验任务 ……52
**第 4 章　数组与字符串——整理与阅读云医院信息的利器** ……53
4.1　数组在云医院中的应用：患者信息管理 ……53
4.1.1　一维数组：体温记录 ……54
4.1.2　二维数组：记录时刻与信息的绑定 ……55
4.1.3　Arrays 类的优化 ……57
4.2　字符串处理：优化云医院中的文本信息 ……57
4.2.1　字符串的创建 ……58
4.2.2　字符串 String 类的操作 ……58
4.2.3　字符串 StringBuffer 类的操作 ……63
4.2.4　格式化字符串 ……66
4.2.5　格式化日期和时间 ……67
4.2.6　病历信息采集与处理 ……70
小结 ……71
习题 ……72
实验三　餐饮后台管理系统——数组和字符串 ……72
实验目标 ……72
实验任务 ……72
**第 5 章　面向对象编程——构建云医院系统的核心模块** ……74
5.1　类的设计：云医院中的实体与行为 ……74
5.1.1　患者类定义 ……75
5.1.2　类中的成员变量 ……76

5.1.3　类中的成员方法 ……77
5.2　对象创建与使用：云医院系统的具化 ……78
5.2.1　患者类对象创建 ……79
5.2.2　对象的使用 ……81
5.2.3　对象间的数据交互 ……82
5.3　static 关键字 ……84
5.3.1　类变量 ……84
5.3.2　类方法 ……85
5.4　this 关键字 ……87
5.5　包的概念 ……88
5.5.1　包语句 ……88
5.5.2　包引用 ……89
5.6　封装与访问权限：保护云医院数据安全 ……90
5.6.1　访问权限修饰符 ……90
5.6.2　封装对患者信息的保护 ……91
小结 ……93
习题 ……94
实验四　餐饮后台管理系统——面向对象编程基础 ……94
实验目标 ……94
实验任务 ……95
**第 6 章　继承与多态——云医院系统的扩展与维护** ……97
6.1　继承机制：构建云医院系统的层级结构 ……97
6.1.1　子类、父类与患者信息 ……97
6.1.2　继承的实现 ……100
6.1.3　成员变量的隐藏与方法的重写 ……101
6.1.4　向上转型 ……103
6.2　多态的应用：云医院中的多元化组合 ……105
6.2.1　多态的概念 ……105
6.2.2　多态的实现 ……105
6.3　抽象类与接口 ……107
6.3.1　抽象类与抽象方法 ……107
6.3.2　接口的声明与使用 ……109
6.3.3　接口回调 ……112
6.3.4　接口做参数 ……113
小结 ……115
习题 ……116
实验五　餐饮后台管理系统——继承与多态 ……116
实验目标 ……116
实验任务 ……116

第 7 章　异常处理——让云医院系统更加稳健 …… 118
7.1　异常处理机制：确保云医院系统稳定运行 …… 118
7.1.1　Java 中常见的系统异常类型 …… 118
7.1.2　异常捕获与处理 …… 119
7.1.3　throw 和 throws 语句 …… 122
7.2　自定义异常：增强云医院系统的异常处理能力 …… 123
7.2.1　数据内存溢出处理 …… 123
7.2.2　信息存储与读取异常分析 …… 125
7.3　内部类 …… 128
7.3.1　内部类的概念 …… 128
7.3.2　匿名内部类 …… 129
小结 …… 130
习题 …… 130
实验六　餐饮后台管理系统——包和异常处理 …… 131
实验目标 …… 131
实验任务 …… 131
第 8 章　集合框架——云医院数据的高效管理 …… 133
8.1　集合的概述和分类 …… 133
8.2　Collection 集合 …… 134
8.2.1　Collection 介绍 …… 134
8.2.2　Collection 集合的常用方法 …… 135
8.2.3　Collection 遍历方法及其在云医院数据处理中的应用 …… 136
8.3　List 集合 …… 140
8.3.1　List 集合的常用方法 …… 141
8.3.2　List 集合的遍历方式 …… 141
8.3.3　ArrayList 集合底层原理 …… 142
8.3.4　LinkedList 集合底层原理 …… 143
8.3.5　LinkedList 集合的特性及其在云医院中的应用场景 …… 144
8.4　Set 集合 …… 147
8.4.1　认识 Set 集合的特点 …… 147
8.4.2　HashSet 集合底层原理 …… 148
8.4.3　HashSet 集合去重原理 …… 149
8.4.4　LinkedHashSet 集合去重原理 …… 150
8.4.5　TreeSet 集合 …… 151
8.5　Collections 工具类及其在云医院数据处理中的应用 …… 154
8.6　Map 集合 …… 156
8.6.1　Map 集合概述 …… 156
8.6.2　Map 集合常用方法及实例分析 …… 157
8.6.3　Map 集合常用遍历方法一 …… 157

8.6.4　Map 集合常用遍历方法二 …… 158
8.6.5　Map 集合常用遍历方法三 …… 159
小结 …… 160
习题 …… 160
实验七　餐饮后台管理系统——集合框架 …… 161
实验目标 …… 161
实验任务 …… 161
第 9 章　文件操作——云医院数据的备份与恢复 …… 163
9.1　文件处理 …… 163
9.1.1　File 类简介 …… 163
9.1.2　使用 File 类 …… 165
9.2　流的基本概念 …… 166
9.2.1　输入/输出流 …… 166
9.2.2　Java 的标准输入/输出流 …… 169
9.3　流的分类 …… 170
9.3.1　文件字节流 …… 170
9.3.2　文件字符流 …… 172
9.3.3　过滤器流 …… 174
9.3.4　字符缓冲流 …… 175
9.3.5　转换流 …… 178
9.3.6　对象流 …… 179
9.4　文件的随机访问 …… 181
9.5　云医院的医护信息读取 …… 183
小结 …… 185
习题 …… 185
实验八　餐饮后台管理系统——输入/输出流与文件操作 …… 186
实验目标 …… 186
实验任务 …… 186
第 10 章　多线程与并发编程——提升云医院系统的响应速度 …… 188
10.1　线程的概念 …… 188
10.2　创建任务和线程 …… 189
10.3　Thread 类的深入 …… 191
10.4　线程池 …… 194
10.5　线程同步 …… 195
10.5.1　synchronized 关键字 …… 197
10.5.2　线程同步的必要性与实现方式 …… 197
10.6　利用加锁同步 …… 198
10.7　线程间协作 …… 199
10.8　线程的状态 …… 203

10.9　云医院与线程密不可分 …… 203
10.9.1　线程在云医院中的作用 …… 204
10.9.2　线程在云医院中的实现 …… 204
小结 …… 204
习题 …… 205
实验九　餐饮后台管理系统——多线程和并发 …… 205
实验目标 …… 205
实验任务 …… 205
**第 11 章　Java 的新特性** …… 207
11.1　Java 各版本特性 …… 207
11.1.1　各版本新特性概述 …… 207
11.1.2　常用特性说明 …… 209
11.2　云医院对 Java 新特性的应用 …… 212
小结 …… 213
习题 …… 214
**第 12 章　综合项目——餐饮后台管理系统** …… 215
12.1　餐饮后台管理系统概述 …… 215
12.1.1　功能设计 …… 215
12.1.2　角色设计 …… 216
12.2　需求分析 …… 216
12.2.1　管理员需求分析 …… 216
12.2.2　商家需求分析 …… 216
12.3　系统设计与架构 …… 217
12.3.1　系统设计概述 …… 217
12.3.2　系统架构设计 …… 217
12.3.3　系统模块设计 …… 217
12.3.4　数据库设计 …… 218
12.3.5　系统交互流程 …… 219
12.4　餐饮后台管理系统的具体功能 …… 219
12.4.1　JDBC 与数据库连接工具类实现 …… 219
12.4.2　实体类设计 …… 221
12.4.3　接口与实现方法 …… 223
12.4.4　主程序实现 …… 227
12.4.5　系统测试与部署 …… 231
12.4.6　维护与优化 …… 232
小结 …… 233
习题 …… 234

# 第 1 章　初识 Java——走进编程的新世界

【知识要点】

- Java 发展史和语言特点
- 什么是 JDK
- Java 开发环境的搭建（安装 JDK 和配置环境变量）
- 第一个 Java 程序（Hello World）
- Java 程序的基本结构（类、方法、语句）

【简介】

本章介绍 Java 的发展史和语言特点，指导读者搭建开发环境，并编写第一个简单的 Java 程序。通过对 Java 程序基本结构和注释的讲解，帮助读者养成良好的编程习惯。

【场景】

Java 编程语言在医院管理系统中扮演了重要的角色。Java 作为一种广泛应用于企业级应用开发的编程语言，特别适合构建复杂的医院管理系统。本章将探讨互联网技术如何在医疗行业中推动医院管理系统的现代化，并且详细介绍 Java 编程语言作为首选技术的原因。

## 1.1　Java 简介与云医院系统开发基础

### 1.1.1　Java 技术概述

Java 是一种广泛使用的编程语言，具有面向对象、跨平台、多线程和安全性高等特点。它由詹姆斯 • 高斯林（James Gosling）领导的团队于 1991 年开发，最初用于为消费电子产品开发一种能够在不同设备上运行的软件平台。1995 年，Java 作为 Sun Microsystems 的产品正式推出，随即在互联网浪潮中迅速流行。

Java 的开发始于一个名为“Green Project”的项目，目标是创建一种能够嵌入消费电子设备中的平台无关软件。最初的版本被称为“Oak”，后来因商标问题更名为 Java。这个名字来源于咖啡，因为团队成员经常在附近的一家咖啡馆讨论项目进展。

Java 的设计初衷是解决 C++语言的复杂性问题，提供一种简洁、安全且便于学习和使用的编程语言。“编写一次，到处运行”的跨平台特性，使得 Java 应用程序可以在不同的操作系统上运行，而无须修改代码。这一特性大大提高了开发效率和软件的可移植性。

随着互联网的普及，Java 凭借其强大的网络编程能力和跨平台特性，迅速成为开发网络应用的首选语言。Sun Microsystems 于 1995 年发布了 Java 的第一个公开版本——Java 1.0，标

志着 Java 正式进入市场。此后，Java 不断发展和完善，推出了多个版本，如 Java 2（也称为 J2SE 1.2）、Java 5、Java 8 等，每个版本都引入了新的特性和优化。

在 2009 年，Oracle 公司收购了 Sun Microsystems，成为 Java 的新的管理者和维护者。Oracle 对 Java 进行了大量的投资和开发，继续推动 Java 在企业级应用、移动开发和大数据等领域的发展。2017 年，Oracle 宣布 Java EE（企业版）移交给 Eclipse 基金会，重新命名为 Jakarta EE，以推动其开源社区的发展。

如今，Java 已经成为全球最流行的编程语言之一，被广泛应用于 Web 开发、企业级应用、大数据处理、移动应用开发等领域。Java 生态系统丰富，包括大量的开源框架和库，如 Spring、Hibernate、Apache Struts 等，进一步提升了开发效率和代码质量。

Java 平台由三大版本组成，分别适用于不同的开发需求和应用场景。

1．Java SE（Java Standard Edition）

Java SE 是 Java 平台的核心，提供了开发和运行 Java 应用程序的基本功能和 API。Java SE 包含了 Java 编程语言的基础类库，如集合框架、输入/输出、网络编程、多线程等。开发者可以使用 Java SE 开发桌面应用、工具、库以及各种通用应用。

Java SE 还包括 JDK（Java Development Kit），提供了编译器（javac）、运行时环境（JRE）、调试工具和文档等开发工具。常见的 Java SE 版本有 Java 6、Java 7、Java 8、Java 11 和 Java 17 等，每个版本都引入了新的特性。

2．Java ME（Java Micro Edition）

Java ME 是为嵌入式设备和移动设备设计的 Java 平台，提供了一组适用于资源受限设备的 API 和运行环境。Java ME 包含 Java SE 的一个子集，并且添加了一些特定于移动和嵌入式开发的功能，如 MIDP（Mobile Information Device Profile）和 CLDC（Connected Limited Device Configuration）。

Java ME 广泛应用于移动电话、嵌入式系统、智能卡和物联网设备的开发。

3．Java EE（Java Enterprise Edition）

Java EE 是针对企业级应用开发的 Java 平台，提供了一整套用于构建大规模、分布式、多层次应用的 API 和运行环境。Java EE 扩展了 Java SE 的功能，增加了许多企业级特性，如 Servlet、JSP、EJB、JMS、JPA 等。

Java EE 的主要特点如下。

组件模型：支持可重用的组件开发，如 EJB（Enterprise JavaBeans）和 Web 组件（Servlet、JSP）。

服务支持：提供事务管理、安全性、并发处理等企业级服务。

分布式计算：支持远程方法调用（RMI）、Web 服务和消息驱动的架构。

持久化：通过 JPA（Java Persistence API）提供对象关系映射（ORM）功能，简化数据库访问。

Java EE 适用于开发大型企业级应用，如企业资源规划（ERP）、客户关系管理（CRM）、供应链管理（SCM）系统等。

通过以上对 Java SE、Java EE 和 Java ME 的介绍，可以看到 Java 平台的全面性和灵活性，它能够满足不同开发需求和应用场景。从桌面应用到企业级应用，再到移动和嵌入式设备应用，Java 平台为开发者提供了强大的工具和技术支持。

### 1.1.2　Java 的发展史

Java 于 1995 年诞生。

1996 年 1 月，第一个 JDK——JDK 1.0 诞生。

1998 年 12 月 8 日，Java 2 企业版 J2EE 发布。

1999 年 6 月，Sun Microsystems 发布 Java 的三个版本：标准版（J2SE）、企业版（J2EE）和微型版（J2ME）。

2000 年 5 月 8 日，JDK 1.3 发布。

2001 年 9 月 24 日，J2EE 1.3 发布。

2002 年 2 月 26 日，J2SE 1.4 发布，自此 Java 的计算能力有了大幅提升。

2004 年 9 月 30 日，J2SE 1.5 发布，这成为 Java 语言发展史上的又一里程碑。为了表示该版本的重要性，J2SE 1.5 被更名为 Java SE 5.0。

2005 年 6 月，JavaOne 大会召开，Sun Microsystems 发布 Java SE 6。此时，Java 的各种版本已经更名，取消其中的数字“2”：J2EE 更名为 Java EE，J2SE 更名为 Java SE，J2ME 更名为 Java ME。

2006 年 12 月，Sun Microsystems 发布 JRE 6.0。

2011 年 7 月，JDK 7.0 发布，增加了简单闭包功能。

2014 年 3 月，Oracle 发布了 Java SE 8，这是一个重要的版本更新，它引入了 Lambda 表达式、Stream API 等现代编程特性。

2017 年 9 月，Oracle 发布了 Java SE 9，其中包含了模块化系统（Project Jigsaw）的实现，使得 Java 的结构更加模块化和可扩展。

2018 年 9 月，Oracle 发布了 Java SE 11，这是自 Java SE 6 以来的第一个长期支持（LTS）版本，去除了 Java EE 和 Corba 模块，加强了安全性和性能优化。

2021 年 3 月，Oracle 发布了 Java SE 16，主要包括新的语言特性和增强的开发者工具，如 Records、Pattern Matching for instanceof 等。

2022 年 9 月，Oracle 发布了 Java SE 17，引入了多项重要功能和改进，如本地模式、垃圾回收器的现代化、嵌入式语言支持等。

### 1.1.3　Java 语言的特点

随着互联网技术的普及，医院管理系统不再局限于简单的数据存储和处理。它们需要能够支持大规模数据管理、实时信息更新、安全性保障以及多方面的集成，以提高医院内部的运行效率和患者服务质量。

那么 Java 到底是一种什么样的语言呢？为什么要学习 Java 呢？Java 是如何应对医院管理系统的复杂需求的呢？看完下面 Java 的几个特点，读者就有答案了。

**1．应用广泛**

Java 是目前使用最为广泛的网络编程语言之一。它具有简单、面向对象、稳定、与平台无关、解释型、多线程、动态等特点。

**2．简单**

Java 语言简单是指这门语言既易学又好用，不要误解为这门语言很干瘪。如果你学习过

C++语言，你就会感觉 Java 很眼熟，因为 Java 中许多基本语句的语法和 C++一样，像常用的循环语句、控制语句等和 C++几乎一样，但不要误解为 Java 是 C++的增强版，Java 和 C++是两种完全不同的语言，它们各有各的优势，将会长期并存下去。如果从语言的简单性方面看，Java 要比 C++简单，C++中许多容易混淆的概念，Java 或者弃之不用，或者以一种更清楚、更容易理解的方式实现，例如，Java 中不再有指针的概念。

**3．面向对象**

基于对象的编程更符合人的思维模式，使人们更容易编写程序。在实际生活中，我们每时每刻都在与对象打交道：日常用的钢笔、骑的自行车、乘坐的公共汽车等。而大家经常见到的卡车、公共汽车、轿车等都会涉及以下几个重要的物理量，如可乘载的人数、运行速度、发动机的功率、耗油量、自重、轮子数目等。另外，还有几个重要的功能，如加速、减速、刹车、转弯等，可以把这些功能称作其具有的方法，而物理量是它们的状态描述。在现实生活中，将这些共有的属性和功能给出一个概念：机动车类。一个具体的轿车就是机动车类的一个实例对象。Java 语言与其他面向对象语言一样，引入了类的概念，类是用来创建对象的模板，它包含被创建的对象的状态描述和方法的定义。

**4．与平台无关**

与平台无关是 Java 语言最大的优势。其他语言编写的程序面临的主要问题是：操作系统的变化、处理器升级以及核心系统资源的变化等都可能导致程序出现错误或无法运行。Java 的虚拟机成功地解决了这个问题，Java 编写的程序可以在任何安装了 Java 虚拟机（JVM）的计算机上正确运行，Sun Microsystems 实现了自己的目标“编写一次，到处运行”。

**5．解释型**

大家所熟知的 C、C++等语言，都只能对特定的 CPU 芯片进行编译，生成机器代码，代码的运行和特定的 CPU 有关。例如，在 C 语言中，会经常碰到类似下面的问题：int 型变量 x 的值是 10，那么代码 printf("%d，%d",x,x=x+1)的输出结果是什么呢？如果上述语句的计算顺序是从左到右，那么结果是“10,11”。但是，有些机器会从右到左计算，那么结果就是“11,11”。Java 不像 C++，它不针对特定的 CPU 芯片进行编译，而是把程序编译为称作字节码的“中间代码”。字节码是很接近机器码的文件，可以在提供 Java 虚拟机的任何系统上被解释执行。Java 被设计成为解释执行的程序，即翻译一句，执行一句，不产生整个的机器代码程序。翻译过程如果不出现错误，就一直进行到执行完毕，否则将在错误处停止执行。同一个程序，如果是解释执行的，那么它的运行速度通常比编译为可执行的机器代码的运行速度慢一些。但是，对 Java 来说，二者的差别不太大，Java 的字节码经过仔细设计，很容易能够使用 JIT 即时编译方式将字节码直接转化成高性能的本地机器码，Sun Microsystems 在 Java 2 发行版中提供了这样一个字节码编译器——JIT（Just In Time），它是 Java 虚拟机的一部分。Java 运行系统在提供 JIT 的同时仍具有平台独立性，因而“高效且跨平台”对 Java 来说并不矛盾。如果把 Java 程序比作“汉语”的话，字节码就相当于世界语，世界语不和具体的国家相关，只要这个国家提供“翻译”，就可以快速地把世界语翻译成本地语言。

**6．多线程**

Java 的特点之一就是内置对多线程的支持。多线程允许同时完成多个任务。实际上，多线程使人产生多个任务在同时执行的错觉，这是因为，目前的计算机处理器在同一时刻只能执行一个线程，但处理器可以在不同的线程之间快速切换，由于处理器的速度非常快，远远

超过了人接收信息的速度，所以给人的感觉好像多个任务在同时执行。C++没有内置的多线程机制，因此必须调用操作系统的多线程功能来进行多线程程序的设计。

**7．安全**

当你准备从网络上下载一个程序时，你最大的担心是程序中含有恶意的代码，试图读取或删除本地机上的一些重要文件，甚至该程序是一个病毒程序等。当你使用支持 Java 的浏览器时，你可以放心地运行 Java 小应用程序 Java Applet，而不必担心病毒程序的感染和恶意的企图，Java 小应用程序将被限制在 Java 环境中运行，不允许访问计算机的其他部分。

**8．动态**

Java 程序的基本组成单元是类，有些类是自己编写的，有些类是从类库中引入的，而类又是运行时动态装载的，这就使得 Java 可以在分布环境中动态地维护程序及类库，而不像 C++那样，每当其类库升级后，相应的程序都必须重新修改、编译。

所以说，Java 是一种简单的面向对象的、分布式的、解释的、健壮的、安全的、结构中立的、可移植的、性能优异的多线程的、动态的语言。

### 1.1.4　云医院系统对 Java 的需求

云医院系统，特别是基于 Java 开发的云 HIS（Hospital Information System）系统，在现代医疗行业信息化和数字化进程中扮演着重要角色。下面具体介绍云医院系统对 Java 的需求。

**1．Java 的适用性**

Java 作为一种广泛使用的编程语言，具有跨平台性、安全性、稳定性和可扩展性等特点，非常适用于开发云医院系统。Java 的这些特性使得云医院系统能够在不同的操作系统和硬件环境中稳定运行，同时保证数据的安全性和系统的可扩展性。

**2．Java 在云医院系统中的具体应用**

（1）系统架构。

云医院系统通常采用 B/S（Browser/Server）架构，这种架构将应用程序的业务逻辑和用户界面分别部署在不同的层次中。Java 语言在 B/S 架构中有着广泛的应用，因为 Java 具有良好的跨平台性，可以轻松地实现跨平台的访问和交互。

在 B/S 架构中，客户端只需安装一个浏览器即可访问系统，而后端则采用 Java 语言开发，负责处理业务逻辑和数据存储。

（2）功能模块开发。

**电子病历管理**：云医院系统需要实现患者的电子病历管理，包括患者信息、病史、检查报告、药物处方、医疗费用等内容的记录和管理。Java 语言可以用于开发这些功能模块，确保数据的准确性和安全性。

**医嘱管理**：医生可以通过系统对患者的医嘱进行管理和记录，医嘱内容包括药物、诊断、治疗方案等。Java 语言能够支持这些功能的实现，并提供良好的用户体验。

**预约和排班管理**：云医院系统需要实现预约和排班管理功能，以方便患者预约医生和医生安排工作时间。Java 语言可以用于开发这些功能模块，提高医疗服务的效率和质量。

**数据分析和决策支持**：云医院系统还需要对医疗数据进行分析和挖掘，以帮助医院进行决策支持和管理优化。Java 语言在数据处理和分析方面有着强大的能力，可以满足这些需求。

（3）系统集成与接口开发。

云医院系统需要与医保系统、其他医疗机构的信息系统等进行集成和对接。Java 语言在接口开发和系统集成方面有着丰富的经验和强大的功能，可以确保系统之间的顺畅交互和数据共享。

通过使用 Java 进行接口开发，云医院系统可以实现与医保系统的有效对接，实现医疗费用的自动结算和报销等功能，为患者和医疗机构提供更加便捷、高效的服务。

（4）安全性与稳定性保障。

Java 语言提供了强大的安全机制和异常处理机制，可以确保云医院系统的安全性和稳定性。通过使用 Java 进行开发，可以有效地防止数据泄露、系统崩溃等安全问题的发生。

同时，Java 还提供了丰富的调试和检测工具，可以帮助开发人员及时发现和修复系统中的问题，确保系统的正常运行。

综上所述，Java 在云医院系统中具有广泛的应用。通过使用 Java 进行开发，可以构建出高效、稳定、安全、可扩展的云医院系统，为医疗行业提供全面的信息化支持和服务。

## 1.2　Java 的开发和执行环境

JDK 是 Sun Microsystems 提供的基础 Java 语言开发工具软件包，其中包含 Java 语言的编译工具、运行工具以及类库。Sun Microsystems 是 Java 的开创者，它的开发工具和运行环境都是免费的。下面详细介绍 JDK 17 的下载、安装和配置过程。

### 1.2.1　JDK 的下载与安装

（1）在 Oracle 公司网站首页中单击“Resources”菜单，如图 1-1 所示，选择“Java Downloads”选项。

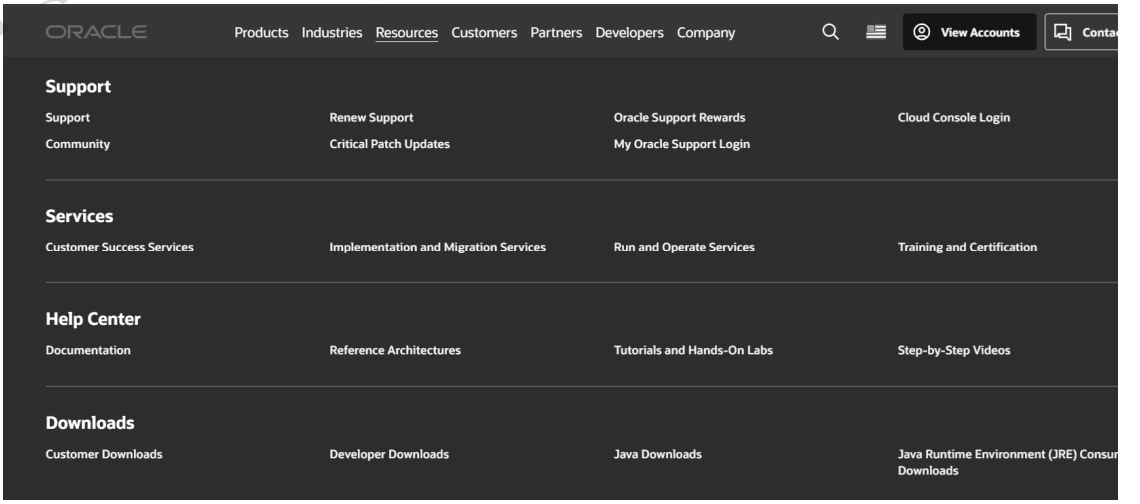



图 1-1　JDK 下载界面

（2）在弹出的界面上，找到 JDK Development Kit 17.0.12 downloads 对应的下载内容，并根据自己的设备选择 Windows 或者其他操作系统，找到压缩包版本 x64 Compressed Archive

并单击下载，如图 1-2 所示。

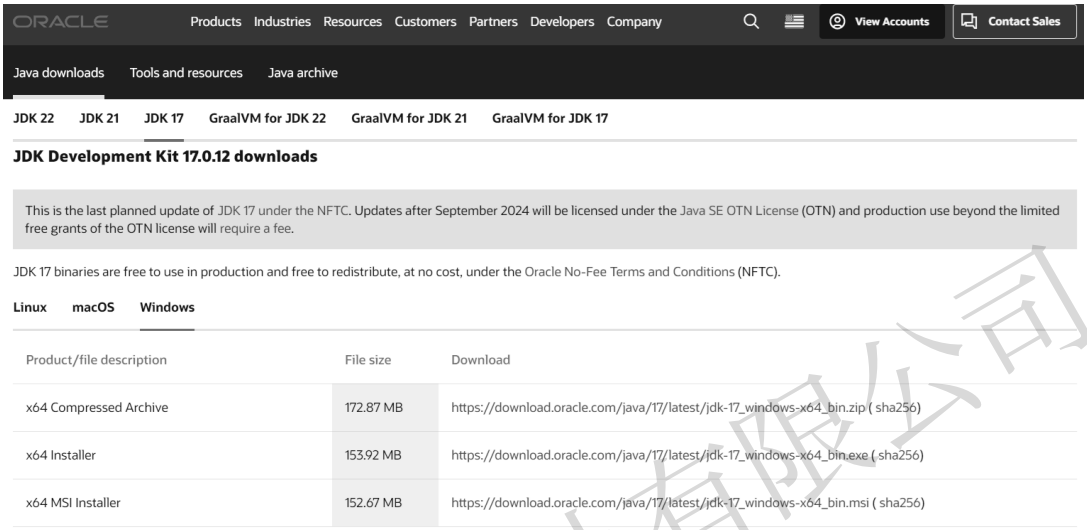



图 1-2　Java 运行平台选择下载界面

（3）下面介绍 JDK 17 的安装方法。

JDK 17 无须采用传统的安装方式，通过快捷安装工具 MSI 安装，直接将下载好的压缩包解压到本地即可，其解压后文件目录界面如图 1-3 所示。

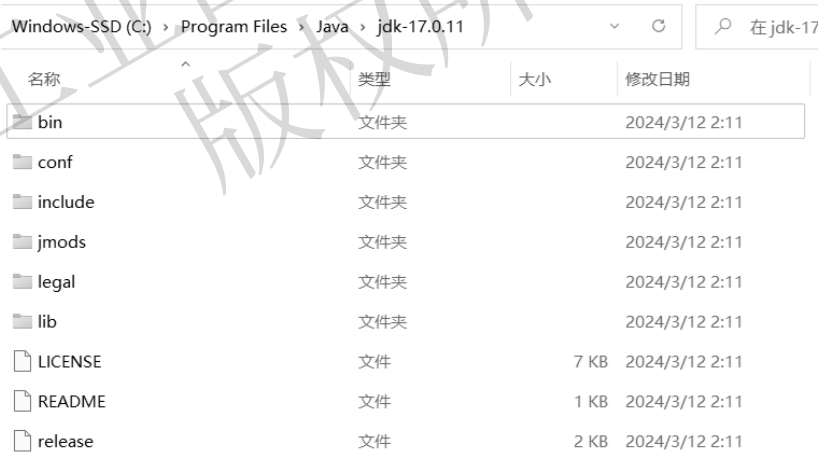



图 1-3　JDK 解压后文件目录界面

## 1.2.2　Java 的环境搭建

JDK 安装完成后，还需要进行 Java 运行环境的配置。配置的主要工作是设置操作系统的 Path 和 Classpath 两个环境变量，把 JDK 中的命令程序路径和 Java 的标准类库加入系统的环境变量中。下面介绍如何在 Windows 下设置 JDK 相关的环境变量。

（1）右击“我的电脑”，选择“属性”命令，弹出“系统属性”对话框，打开“高级”选项卡，如图 1-4 所示，单击“环境变量”按钮。


图 1-4 “高级”选项卡

（2）在打开的对话框中新建一项系统变量“JAVA_HOME”，值为 JDK 的安装路径，如图 1-5 所示，单击“确定”按钮。


图 1-5　设置系统变量“JAVA_HOME”

（3）配置系统变量，双击“path”，单击“新建”按钮，弹出“编辑系统变量”对话框，填入“%JAVA_HOME%\bin”，最后单击“确定”按钮，如图 1-6 所示。

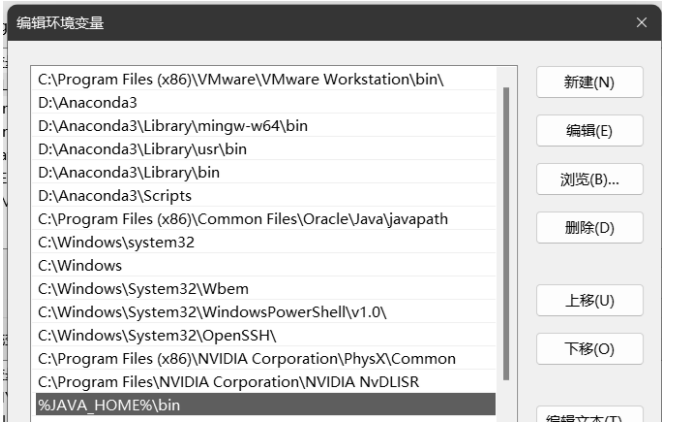



图 1-6 “编辑系统变量”对话框

完成所有的变量设置后，JDK 的运行环境即可生效。至此，我们便完成了 Java 运行环境的设置。

### 1.2.3　Java 项目结构初识

Java 如何做到让机器理解想要做的东西？下面用图 1-7 来描述这个过程。


图 1-7　Java 程序执行过程

首先编写一个后缀为.java 的源文件，然后通过 Java 编译器（javac.exe）把源文件编译成后缀为.class 的字节码文件，最后通过 Java 解释器（java.exe）运行这个程序。

**1．编写代码**

编写代码是指在 Java 开发环境中进行程序代码的输入，最终形成后缀为.java 的 Java 源文件。

**【例 1-1】**用记事本编写一个简单的医院登录入口的 HisMenu.java 程序。

```
/* HisMenu.java */
    public class HisMenu{              //一个 Java 应用程序
        public static void main(String args[ ]) {
            System.out.println("云医院后台员工登录入口");
            System.out.println(" 1.挂号收费员");
            System.out.println(" 2.门诊医生");
            System.out.println(" 3.退出系统");
        }
    }
```

**2．编译**

写完 Java 代码后，机器并不认识所写的 Java 代码，需要编译为字节码文件。使用命令行在控制台调用 JDK，输入如下命令：

```
javac HisMenu.java
```

将 HisMenu.java 文件编译成 HisMenu.class 文件。编译是指使用 Java 编译器对源文件进行错误排查的过程，编译后生成后缀为.class 的字节码文件。字节码文件是一种与任何具体机器环境及操作系统无关的中间代码，它是一种二进制文件。

**3．运行**

运行是指使用 Java 解释器将字节码文件翻译成机器代码，执行并显示结果。解释器是 Java 虚拟机的一部分，在运行 Java 程序时，在命令行输入如下命令：

```
java HisMenu
```

这时 JDK 会启动 JVM，然后由它来负责解释执行 HisMenu.class 字节码文件。

例 1-1 的 HisMenu.java 源文件的运行结果如图 1-8 所示。

```
云医院后台员工登录入口
1.挂号收费员
2.门诊医生
3.退出系统
```

图 1-8　例 1-1 运行结果

Java 程序可以分为 Java Application（Java 应用程序）和 Java Applet（Java 小应用程序）。其中，Java Application 必须通过 Java 解释器来解释执行其字节码文件，Java Applet 必须使用支持它的浏览器运行。两者的区别如表 1-1 所示。

表 1-1　Java Application 和 Java Applet 的区别

| 特性 | Java Application | Java Applet |
| --- | --- | --- |
| 执行环境 | 本地 JVM | 浏览器的 Java 插件（已不再支持） |
| 使用场景 | 桌面应用、服务器应用、命令行工具 | Web 浏览器内的交互式小程序（现已淘汰） |
| 独立性 | 独立运行，支持全面的 Java 功能 | 依赖于浏览器插件，功能有限且受限于安全性 |
| 典型应用 | 桌面软件、Web 服务器、游戏、数据库管理软件等 | 早期的 Web 小游戏、动画、交互式表单 |
| 安全性 | 完全访问系统资源 | 受限于“沙盒”环境，访问权限受限 |
| 发展现状 | 仍在广泛使用 | 已被淘汰，现代浏览器不再支持 |

Java Application 仍然是现代开发中非常重要的一部分，而 Java Applet 由于安全性问题和技术更新，已逐步退出历史舞台，本书将使用 Java Application 进行介绍。

## 1.3　Java 开发工具 IntelliJ IDEA

Java 的开发除了使用命令行方式外，也支持集成开发环境。这些开发工具集成了编译器和解释器，方便使用。最具代表性的开发工具是 IntelliJ IDEA 和 Eclipse，前者的功能强大，能胜任各种企业级 Java 程序的开发，也是近几年来 Java 开发的首选工具。本书后面所有的例题都是在此环境下运行调试的，下面简单介绍如何在 IntelliJ IDEA 环境下开发、运行相关的程序。

### 1.3.1　IntelliJ IDEA 简介

IntelliJ IDEA 是由 JetBrains 开发的一款强大的集成开发环境（IDE），主要用于 Java 开发，同时也支持 Kotlin、Scala、JavaScript 等多种语言。凭借其智能代码编辑功能、强大的调试工具和丰富的插件，IntelliJ IDEA 成为开发者中广受欢迎的工具。

**1．主要特点**

（1）智能代码编辑功能：提供智能代码补全、重构和错误检测，帮助快速编写和优化代码。

（2）强大的调试工具：内置调试器支持断点调试和变量监控，集成 JUnit、TestNG 等测试框架，方便测试和分析代码。

（3）支持版本控制集成：支持 Git、SVN 等版本控制系统，简化了代码管理和团队协作。

（4）多语言和框架支持：除了 Java，还支持 Kotlin、Scala、JavaScript 等语言，并且集成了 Spring、Hibernate 等常用框架。

下面讲解如何安装与使用 IntelliJ IDEA。

### 2．下载 IntelliJ IDEA

访问 JetBrains 官方网站，在“IntelliJ IDEA”界面中，单击“Download”按钮。这里可以看到两个版本：Community Edition（社区版）和 Ultimate Edition（旗舰版）。如果只是进行一般的 Java 开发或学习，社区版已经足够。若需要高级功能，如对 Web、企业级开发的支持，则可以选择旗舰版，如图 1-9 所示。

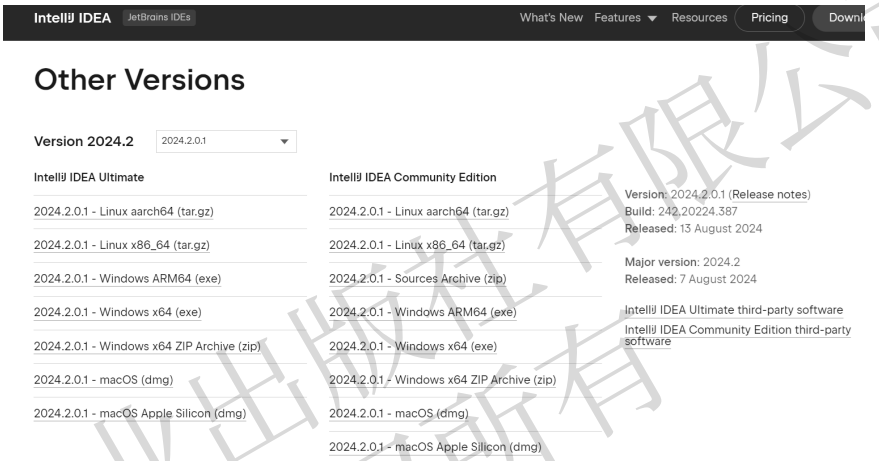



图 1-9　版本选择界面

根据需要选择适合的版本（Windows、macOS、Linux），然后单击“Download”按钮，下载对应的安装文件。

### 3．安装运行

运行下载的安装文件，开始进行安装。该软件的安装步骤简单，按照以下流程进行即可，如图 1-10～图 1-13 所示。

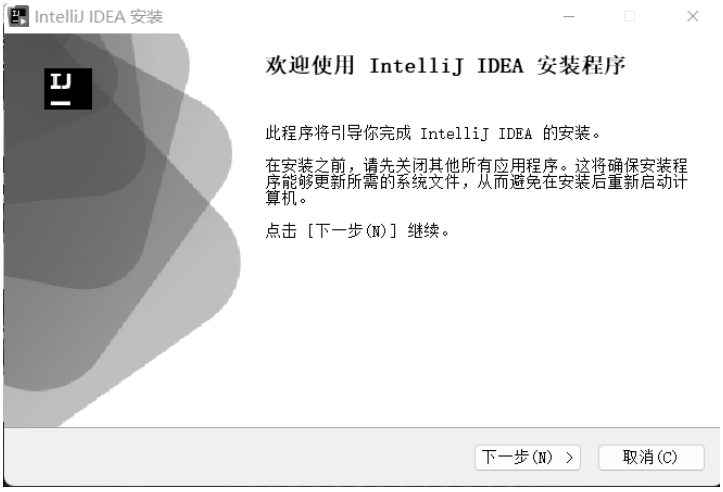



图 1-10　安装开始界面


图 1-11　选择安装目录


图 1-12　设置软件的基本配置

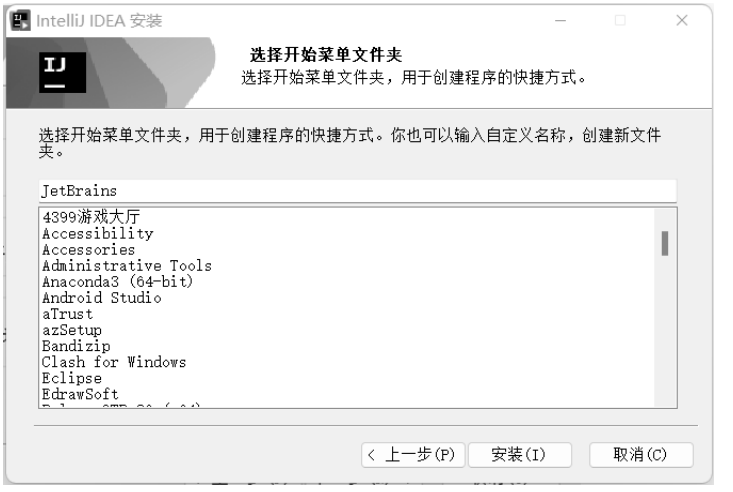



图 1-13　完成软件的安装

### 1.3.2　IntelliJ IDEA 开发 Java 程序

本小节通过例 1-1 的程序，介绍使用 IntelliJ IDEA 开发 Java 程序的过程及相关操作细节。

（1）单击“Projects”|“New Projects”，弹出相应的子菜单，输入自定义的项目名称“Demo”，并选择指定的开发语言“Java”和已经搭建好的 JDK 环境，如图 1-14 所示。

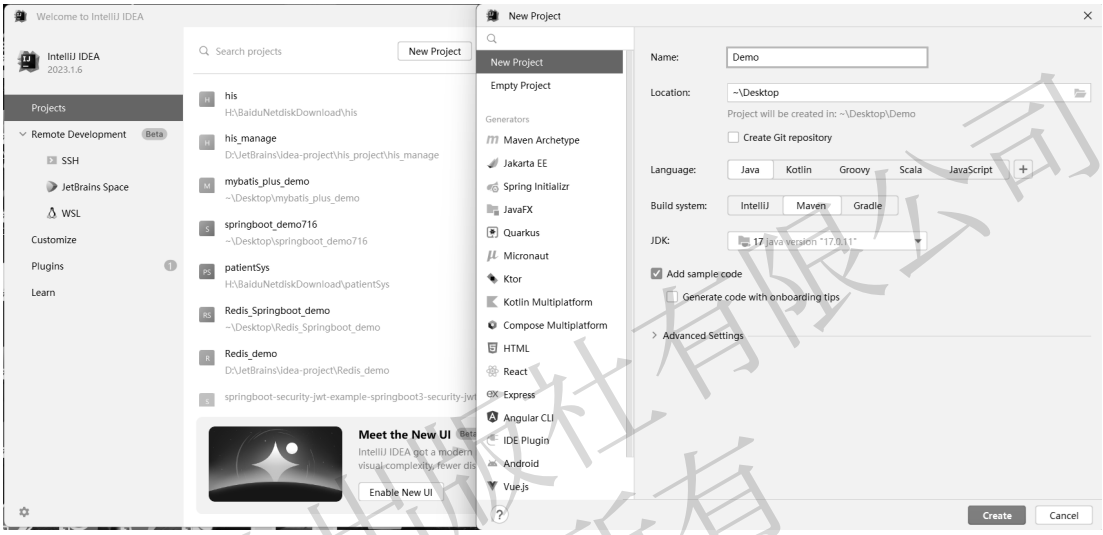



图 1-14　创建项目界面

（2）单击图 1-14 中的“Create”按钮即可完成项目创建，出现如图 1-15 所示的界面。

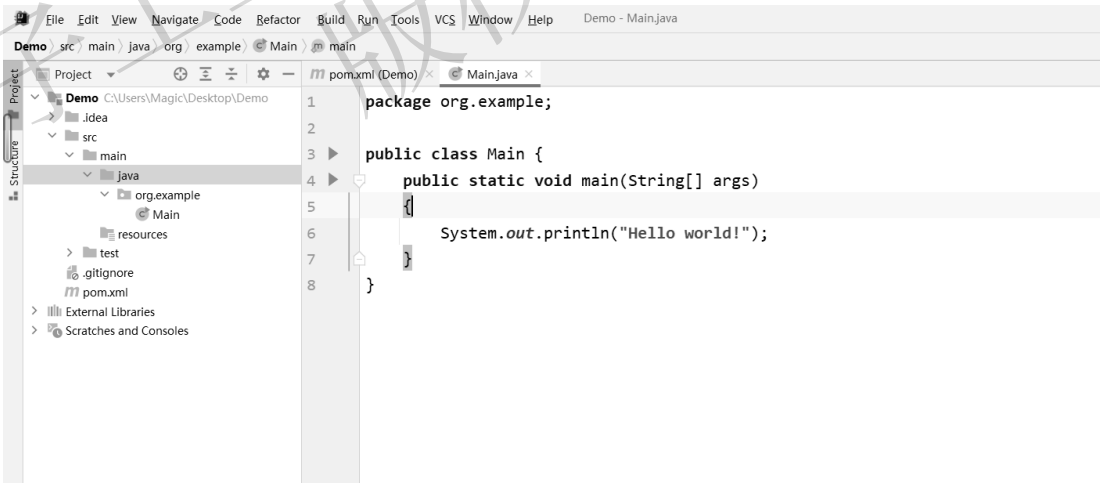



图 1-15　项目初始界面

（3）在该初始的“Main.java”文件中进行编辑并保存，或者创建一个新的文件，选择“org.example”目录，右击选择“New”|“Java Class”，创建一个新的文件“HisMenu.java”，效果如图 1-16 所示。

（4）创建完毕后，即可在对应的文件中进行编辑并保存，然后右击选择“Run HisMenu.main()”运行文件，如图 1-17 所示。

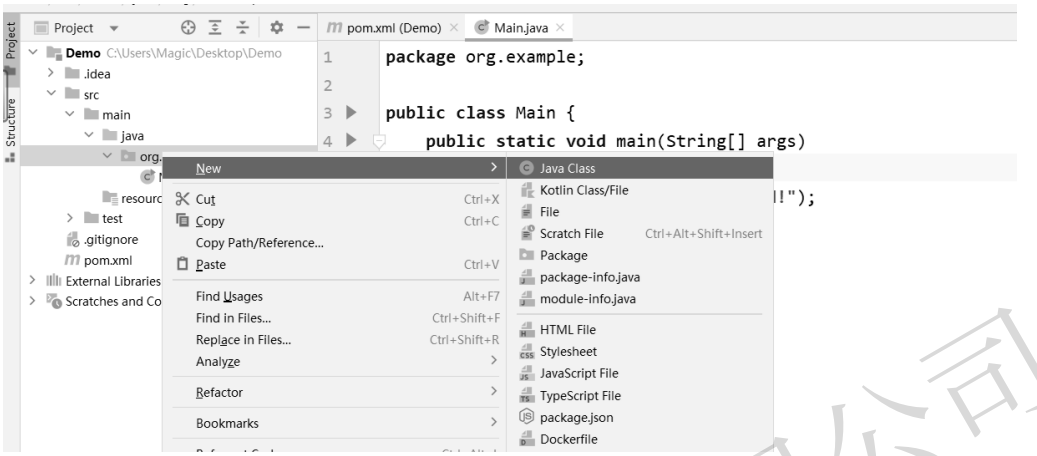



图 1-16　新建一个文件

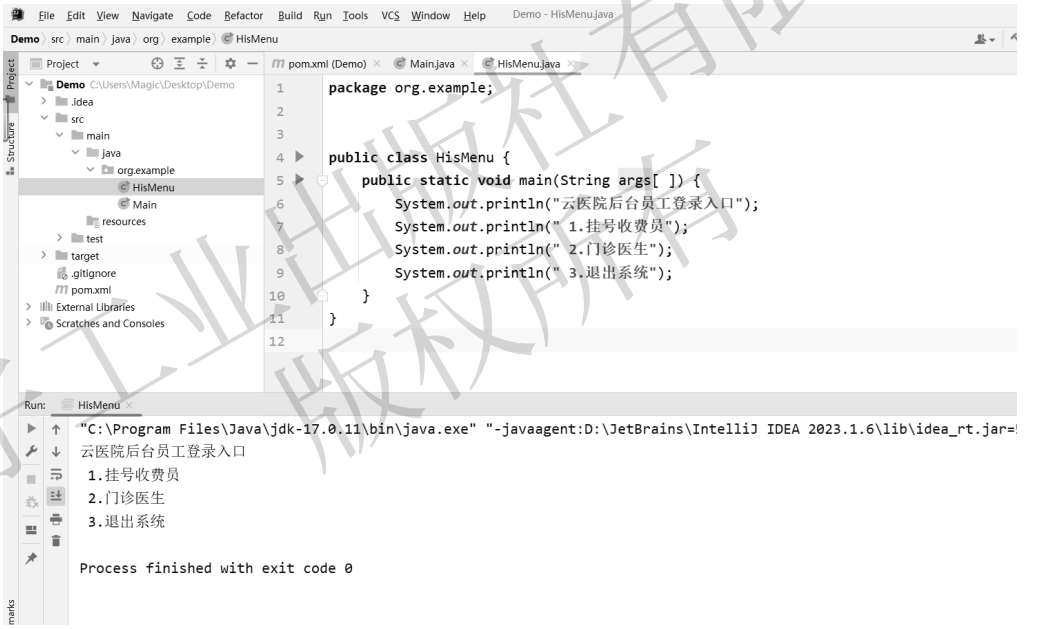



图 1-17　运行效果

通过上述操作步骤，就可以实现使用 IntelliJ IDEA 对一个 Java 文件从创建到编译运行的过程。由于篇幅有限，这里对该开发工具不再赘述，若读者有兴趣，可以参看其他相关的参考资料。

## 1.4　云医院项目的初步搭建

### 1.4.1　项目技术选型

在设计云医院系统时，技术选型是一个至关重要的环节，它直接关系到系统的性能、稳定性、安全性以及未来的可扩展性。以下是一些关键的技术选型考虑因素，以及针对云医院

系统的具体建议。

1．编程语言——Java

**优势**：Java 是一种广泛使用的编程语言，具有跨平台性、安全性、稳定性和可扩展性。它拥有强大的生态系统，包括丰富的开源库和框架，如 Spring Boot 等，这些都可以为云医院系统的开发提供有力支持。

**应用**：Java 可用于开发云医院系统的后端服务，如用户管理、预约挂号、在线问诊、健康档案管理等核心功能模块。

2．数据库——关系型数据库 MySQL

**优势**：MySQL 等关系型数据库具有数据一致性强、事务处理能力强等特点，适合存储和管理结构化数据。

**应用**：在云医院系统中，MySQL 可用于存储患者信息、病历资料、预约记录等关键数据。

3．前端技术——Vue.js

**优势**：React.js 和 Vue.js 等现代前端框架具有组件化开发、响应式设计、高效的数据绑定等特点，能够提升用户界面的交互体验和性能。

**应用**：这些前端框架可用于开发云医院系统的用户界面，如患者端 App、医生工作站等。

4．微服务架构——Spring Cloud

**优势**：微服务架构将系统拆分为多个独立的服务模块，每个模块通过 API 进行通信。这种架构具有高度的可扩展性、灵活性和可维护性。

**应用**：在云医院系统中，可采用微服务架构来实现系统的模块化设计，如将用户服务、问诊服务、预约服务等拆分为独立的服务模块。

5．实时通信技术——WebSocket

**优势**：WebSocket 是一种在单个 TCP 连接上进行全双工通信的技术，能够实现实时的数据传输和通信。

**应用**：在云医院系统中，WebSocket 可用于实现在线问诊、远程医疗咨询等实时通信功能。

6．云服务与容器化技术

（1）云服务

**优势**：云服务提供商如阿里云、腾讯云等提供了丰富的云计算资源和服务，如服务器、存储空间、数据库、CDN 等，能够降低系统的运维成本和复杂度。

**应用**：在云医院系统中，可利用云服务来部署和运维系统，提高系统的可用性和可扩展性。

（2）容器化技术——Docker

**优势**：Docker 等容器化技术能够实现应用程序的打包、分发和部署，提高系统的可移植性和部署效率。

**应用**：在云医院系统中，可采用 Docker 来打包和部署各个服务模块，实现系统的快速部署和扩展。

7．安全性与隐私保护措施——数据加密技术

**重要性**：云医院系统涉及大量敏感数据，如患者身份信息、病历资料等，必须高度重视数据安全和隐私保护。

**应用**：应采用先进的数据加密技术，如 AES、RSA 等，确保用户数据在传输和存储过程中的安全性。

综上所述，在设计云医院系统时，应根据系统的实际需求和特点进行技术选型。通过合理地选择编程语言、数据库、前端技术、微服务架构、实时通信技术、云服务与容器化技术以及安全性与隐私保护措施，可以构建出一个高效、稳定、安全、可扩展的云医院系统。

### 1.4.2　项目结构规划

使用 Java 开发云医院系统时，通常会采用一种或多种现代且高效的开发模式，以确保系统的稳定性、可扩展性和易用性，以下是一些常见的开发模式。

**1．微服务架构（Microservices Architecture）**

微服务架构是一种将大型应用拆分成一系列小的、自治的服务的方法，每个服务运行在其独立的进程中，并使用轻量级通信机制（如 HTTP REST API）进行通信。对云医院系统而言，微服务架构可以帮助其实现以下性能。

- 高可用性：通过负载均衡和容错机制，确保系统的高可用性。
- 可扩展性：根据业务需求，可以轻松地增加或减少服务的实例数量。
- 技术多样性：不同的服务可以使用最适合其需求的技术栈进行开发。

**2．前后端分离（Frontend-Backend Separation）**

前后端分离是一种将前端（用户界面）和后端（业务逻辑和数据存储）的开发和维护工作分离的开发模式。这种模式在云医院系统的开发中非常常见，它带来了以下好处。

- 提高开发效率：前端和后端开发人员可以并行工作，互不影响。
- 降低耦合度：前端和后端通过 API 进行通信，降低了系统各部分的耦合度。
- 更好的用户体验：前端可以专注于提供更好的用户体验，而后端则专注于提供稳定、高效的数据服务。

**3．敏捷开发（Agile Development）**

敏捷开发是一种以人为核心、迭代、循序渐进的开发方法。在云医院系统的开发中，敏捷开发可以帮助团队快速响应变化，确保系统始终满足用户需求。敏捷开发强调以下原则。

- 快速迭代：通过短周期的迭代，快速交付可用的软件产品。
- 持续反馈：在开发过程中不断收集用户反馈，及时调整开发计划。
- 团队协作：鼓励团队成员之间的紧密协作和有效沟通。

**4．DevOps（Development and Operations）**

DevOps 是一种强调开发（Dev）和运维（Ops）之间沟通与合作的文化、实践和工具集。在云医院系统的开发中，DevOps 可以帮助团队实现自动化部署、持续集成和持续监控，从而提高系统的稳定性和可靠性。

**5．云服务集成（Cloud Service Integration）**

由于云医院系统是基于云计算的，因此会大量使用云服务（如 AWS、Azure、阿里云等）。在开发过程中，需要集成这些云服务以提供数据存储、计算资源、安全保护等功能。云服务集成可以帮助团队快速构建稳定、可扩展的系统，并降低运维成本。

## 小　　结

本章介绍了 Java 编程语言的基本概念及其发展历程。Java 以其“编写一次，到处运行”的特性，在各平台上得到了广泛的应用，包括 Web 开发、移动应用和企业级应用等。

本章同时还讲解了如何搭建 Java 开发环境，包括安装 JDK（Java 开发工具包）和选择适合的 IDE（如 IntelliJ IDEA、Eclipse）。通过了解 Java 程序的基本结构及其主要组件的相关知识，能够为后续章节的学习奠定基础。

本章的内容为后续深入理解 Java 编程打下了坚实的基础，为接下来的学习提供了必要的背景支持。

## 习　　题

1-1　解释 Java 的“编写一次，到处运行”特性是如何实现的？

1-2　以下哪项不是 Java 的主要特点。

A．自动垃圾回收　　B．强类型语言　　C．编译器和解释器　　D．多线程支持

1-3　操作题：在 IDE（如 IntelliJ IDEA 或 Eclipse）中创建一个新的 Java 项目，编写一个简单的 Java 程序，输出“Hello, World!”。

1-4　选择题：下列哪项是 Java 的典型应用场景？（　　）

A．系统底层开发　　B．硬件驱动程序

C．Android 应用开发　　D．嵌入式系统开发

1-5　描述 Java 在企业级应用开发中的作用，并举例说明 Java 的企业级框架或技术。