软件测试

第2版

RUANJIAN CESHI

策划编辑：高　锐
责任编辑：陈坤朋
封面设计：黄燕美

ISBN 978-7-5635-7450-6

定价：43.00元

软件测试 第2版 主编 吴迪

北京邮电大学出版社

福建省“十四五”职业教育规划教材

软件测试

第2版

主编　吴　迪
主审　王宁宁

北京邮电大学出版社
www. buptpress. com

福建省“十四五”职业教育规划教材

软件测试

第2版

主编　吴　迪

主审　王宁宁

北京邮电大学出版社
www. buptpress. com

内 容 简 介

本书是福建省“十四五”职业教育规划教材。全书包括9个模块，介绍了软件测试的基本原理与方法，内容包括软件测试基础知识，黑盒测试，白盒测试，性能测试，缺陷报告、分析及处理，API测试，自动化测试，软件产品测试与验收，测试实例——黎明资产管理系统。

本书既可用作高等职业院校软件测试课程的教材，也可用作各类软件工程技术人员的参考书。

图书在版编目（CIP）数据

软件测试 / 吴迪主编. -- 2版. -- 北京 : 北京邮电大学出版社，2024. -- ISBN 978-7-5635-7450-6

Ⅰ. TP311.5

中国国家版本馆CIP数据核字第2024U72R69号

策划编辑：高　锐　　**责任编辑**：陈坤朋　　**封面设计**：黄燕美

出版发行：北京邮电大学出版社

社　　址：北京市海淀区西土城路10号

邮政编码：100876

发 行 部：电话：010-62282185　传真：010-62283578

E-mail：publish@bupt.edu.cn

经　　销：各地新华书店

印　　刷：三河市骏杰印刷有限公司

开　　本：787 mm×1 092 mm　1/16

印　　张：13.25

字　　数：274千字

版　　次：2020年8月第1版　2024年12月第2版

印　　次：2024年12月第1次印刷

ISBN 978-7-5635-7450-6　　定　价：43.00元

Preface
第2版前言

人工智能(artificial intelligence,AI)技术的飞速发展极大地提升了软件开发的效率,但也给保障软件质量带来了挑战。AI模型的训练和推理过程可能存在误差或变化,但通过运用必要的软件测试技术和方法,结合大量的测试数据进行验证,并通过持续集成来优化,就可以保障在AI环境下开发的软件的质量。

本书第1版自2020年出版以来,受到了广大师生的好评,同时也收到了很多中肯的建议。为贯彻落实党的二十大精神进教材、进课堂、进头脑,加强数字化教学资源建设,以培养适应新时代软件测试领域需要的高素质技术技能人才,编者对教学内容和案例、软件版本、教学资源等方面做了修订,体现了如下新特色。

1. 更新软件

鉴于技术发展迅速,将原有的性能测试软件LoadRunner 12.55版本更新为LoadRunner 2022社区教育版。同时,为了便于测试团队分享和交流,将流程图画图软件从Visio改为了PlantUML。此外,鉴于Chrome浏览器的快速更新,教材也升级了Selenium WebDriver版本至123版本。其他软件如Postman、IntelliJ IEDA、PyCharm等也均改用了流行的免费授权教育版本。

2. 更新内容

模块1:重新梳理并更新了软件测试的基础知识,对知识的分类和内容进行更加系统、全面的阐述,旨在帮助学生对软件测试技术形成更加清晰和深入的认识。此外,书中还以泉州银行信息技术部的需求与测试中心工作现场为例新增了企业实际测试场景,以强化学生的实践应用能力。

模块2:替换了等价类划分法的第一个案例,使学生更易于理解案例设计的意图。

模块3:对部分插图进行了替换,将在基本路径测试法举例中使用的JavaScript语言改为Java语言,但保持了原有的思路和方法,以减少代码语法差异给学生带来的困扰。同时,对白盒测试的过程进行更加完整的阐述,并提供了操作视频。此外,还新增了安装和使用PlantUML插件绘制流程图的内容,并对习题部分进行了相应优化。

模块 4：将 LoodRunner 版本更新为 LoadRunner 2022 社区教育版，并更换了与新版本相对应的插图。

模块 5：补充了表格形式的报告，并对报告中的字段名称进行了详细的补充说明。

模块 6：本模块为新增内容，专注于讲解 API 测试方法。这些方法在软件测试领域中应用广泛，具有重要的实践意义。

模块 7：升级 Selenium WebDriver 版本，并提供相关的配套资源。在此基础上新增了 Page Object 的设计思想和实现过程，以及如何在 PyCharm 中编写符合 Page Object 模式的自动化测试脚本的内容。同时，还提供了相关的资源和代码。

模块 8：为了提升桩模块集成测试的实用性，新增了基于 Java 和 Mockito 的桩模块集成测试实例。学生对桩模块集成测试知识的学习不能只停留在理论层面，而应能够真正地进行实践应用。

模块 9：对部分内容进行了重新设计和改写，旨在使测试案例更加标准化和系统化。

3. 更新资源

（1）将本书的在线测试平台升级并迁移到新服务器，增强容错率，提高使用时的稳定性。

（2）全书认真贯彻落实课程思政的相关要求，使软件测试与生活和工作联系更加紧密，从而激发学生学习知识的热情，使其树立科技报国的信念。

（3）新增了操作视频、随堂测试题和知识拓展等栏目，扫描书中二维码即可查看。

（4）全面更新了配套的教学资料包，提供了包括课程标准、教案、试卷和案例源码等在内的教学资源。

本书由黎明职业大学软件专业教授吴迪和泉州银行信息技术部王宁宁、泉州嘉云科技有限公司技术团队组成的校企“双元”团队共同完成，体现了产教融合、校企合作的特色。在此谨向所有参与人员给予的大力支持表示诚挚的谢意。

由于编写水平有限，书中难免存在不足之处，恳请广大读者批评指正。

编　者

Preface
第1版前言

随着高等职业教育的快速发展和社会信息化的普及，就业市场对高校软件测试人才的需求增长很快。由于软件项目已呈现大型化、复杂化的发展趋势，大规模软件有成千上万个模块与接口，分布式、跨地域、跨系统的组织架构，开发周期较长且后续要不断地迭代和维护。这些都需要依赖软件测试提供对软件产品的质量保障。国内许多中小型软件公司对软件测试的重视程度不够，也没有规范化的做法，不过随着软件测试在整个软件开发过程中的占比越来越大，社会对软件测试人才的需求会大为增加。在这样的背景下，我们编写了本书。

本书具有以下特点。

（1）坚持正确导向，面向市场需求 。教材编写以《国家职业教育改革实施方案》、《职业院校教材管理办法》等文件精神为指导，落实立德树人根本任务，以企业岗位需求导向，引入真实案例，确保优质内容进课堂进教材，提高软件测试人才培养质量。

（2）坚持课程思政，突出职教特色。教材编写注重计算机伦理教育，引导学生树立正确的软件知识产权理念，坚定学生责任主体意识，遵守社会规范，形成正确的社会价值和伦理价值。通过课程思政，激发学生的职业认同和职业自信，提高学生的沟通表达、团结协作、精益求精的能力，培养学生规范的编码习惯，弘扬精益求精的专业精神、职业精神、工匠精神。

（3）内容科学先进，职业面向针对性强。教材编写以企业完整的工作过程为典型进行教学设计。在软件测试的实践教学中，融入软件工程整体思想，改革课程内容，创建典型项目案例，辅助自主开发的软件测试程序开展线上线下混合式教学。为了适应新时代软件测试人才培养要求，教材编写以项目贯通实践教学新模式，建设新时期课程教学标准和教学设计。

（4）突出“理实结合”“校企双元”，符合学生成长规律。教材紧跟信息技术发展和行业人才需求，充分反映行业企业发展最新进展，及时吸收比较成熟的新技术、新工艺、新规范等。编写时注重理论和实践相统一，强调实践性。让学生通过项目学习、案例学习、模块化学习系统掌握理论知识和实践操作技能。教材编写过程中得到润和软件泉州分公司郑春明先生的大力支持，许多案例都来源于企业真实业务。教材编排科学合理、层次明晰，图文并茂，生动活泼，形式新颖。

本书包括 8 个模块，具体内容介绍如下。

模块 1 主要介绍软件测试的基本知识，包括软件测试的发展过程、软件测试的目的和必要性、软件测试分类、常见的软件测试模型及软件测试原则，以及软件测试的一般流程。读者在学习完本模块的知识之后，能对软件测试产生比较直观的印象，为后续学习打下基础。

模块 2 介绍了黑盒测试常用的技术方法，包括等价类划分法、边界值分析法、因果图与决策法、正交试验设计法等，以及如何设计测试用例。

模块 3 介绍了逻辑覆盖法和基本路径测试法等白盒测试方法，以及如何使用 Visio 2013 绘制程序流程图和控制流图。

模块 4 介绍了性能测试的相关知识，包括性能测试的概念、目的、类型、指标和流程；以实例讲解的方式详细地介绍了 LoadRunner 性能测试工具的使用。通过本模块的学习，读者可以对性能测试有清晰的认识与了解，并熟悉性能测试工具 LoadRunner 的使用方法。

模块 5 介绍了什么是软件缺陷、如何发现缺陷、如何填写缺陷报告以及缺陷的状态变化和处理流程，并给出了缺陷报告及处理报告、返测报告的样例。通过本模块的学习，读者能意识到及早发现、报告和处理缺陷的必要性，避免隐藏缺陷导致的严重后果。

模块 6 介绍了自动化测试的相关知识，以实例讲解的方式介绍了 Selenium 的用法，最后编辑、执行脚本，完成自动化测试的过程。通过本模块的学习，读者能对自动化测试有比较全面的了解，并掌握自动化测试工具的使用方法，按流程进行测试。

模块 7 介绍了集成测试、系统测试、验收测试三个阶段的测试知识，给出了集成测试的策略及评价标准。同时介绍了验收测试的方法及测试可能出现的结果及分级标准。通过本模块的学习，读者能对测试模型有更深的认识，掌握产品测试的策略与方法。

模块 8 通过一个资产管理系统测试实例介绍了项目测试流程及各种文档的编写格式和方法。

本书编写时注重理论和实践相结合，项目导向、任务驱动，让读者在学习过程中能逐渐提升实操能力。另外，本书语言表达简单通俗，知识覆盖全面，可以让读者由浅入深、循序渐进地掌握软件测试的规范流程和操作技能。

本书由黎明职业大学副教授吴迪主编。由于编写水平所限，书中出现的不足之处恳请广大读者批评指正。

编　者

Contents

目 录

模块 1
软件测试基础知识

软件开发的基本要求是按时并高质量地发布软件产品，而软件测试是保证软件质量的重要手段之一。不论采用什么技术和方法来进行软件开发，软件产品中仍然或多或少地存在错误。采用先进的开发方式和较完善的开发流程可以减少错误的出现，但是不会根除软件中的错误，这些引入的错误需要通过测试来发现。软件测试伴随着软件开发，以检验每一个阶段性的成果是否符合质量要求和达到预先定义的目标，以尽早地发现错误并及时修正。

1.1 软件测试的起源与发展

在信息化时代，人们的手机、计算机上都安装有各种各样的软件。软件是人们工作和学习的好助手，人们平时常用的软件包括 Windows 操作系统、Office 办公软件、QQ 和微信等。软件是通过软件开发流程开发出来的产品，它是计算机系统运行的指令、数据和相关文档的集合，即软件等于程序、数据加上文档。程序是事先按照预定功能、性能等要求设计和编写的指令序列；数据是使程序正常处理信息的数据结构及信息表示；文档是与程序开发、维护和使用有关的技术数据和图文资料。

软件测试是伴随着软件的产生而产生的。早期的软件规模都很小、复杂程度较低，软件开发的过程混乱无序、相当随意，测试的含义比较狭窄，开发人员将测试等同于“调试”，目的是纠正软件中已知的错误经常常由开发人员自己完成这部分工作。对测试的投入极少，测试介入也较晚经常常是等到形成代码，产品已经基本完成时才进行测试。

到了 20 世纪 80 年代初期，IT 行业进入了大发展时代，软件趋向大型化、高复杂度，软件的质量越来越重要。这时，一些软件测试的基础理论和实用技术开始形成，并且人们开始为软件开发设计了各种流程和管理方法，软件开发的方式也逐渐由混乱无序的开发过程过渡到结构化的开发过程，以结构化分析与设计、结构化评审、结构化程序设计及结构化测试为特征。人们还将“质量”的概念融入其中，软件测试的定义发生了改变，测试不是一个单单发现错误的过程，而是将测试作为软件质量保证（SQA）的主要职能，包含软件质量评价的内容，Bill Hetzel（比尔 · 赫策尔）在《软件测试完全指南》（*Complete Guide of Software Testing*）一书中指出：“测试是以评价一个程序或者系统属性为目标的任何一种活动。测试是对软件质量的度量。”这个定义至今仍被引用。

1983 年，电气电子工程师学会（Institude of Electrical and Electronics Engineers，IEEE）提出的软件工程术语中给软件测试下的定义是：“使用人工或自动手段来运行或测试某个系统的过程，其目的在于检验它是否满足规定的需求或弄清预期结果与实际结果之间的差别。它是帮助识别开发完成（中间或最终版本）的计算机软件（整体或部分）的正确度、安全度和质量的软件过程。”这个定义明确指出：软件测试的目的是检验软件系统是否满足需求。它再也不是一个一次性的、开发后期的活动，而是与整个开发流程融合成一体。软件测试已成为一个专业，需要运用专门的方法和手段，由专门人才来操作。

1.2 软件测试的定义和目标

1.2.1 软件测试的定义

软件测试是一种系统性的活动，它涉及运行软件或系统的过程，以此来评估其功能、性能和质量，目的是发现潜在的问题或错误。作为软件开发生命周期中的一个重要环节，软件测试旨在确保软件能够按照预期的方式正常运行。

1.2.2 软件测试的目标

软件测试的主要目标包括以下几点。

(1)发现缺陷。软件测试的主要目的是识别软件中存在的潜在缺陷或错误，这些缺陷可能包括功能错误、设计问题、性能瓶颈和安全漏洞等。通过测试，可以尽早发现并修复这些问题，从而提升软件的质量和稳定性。

(2)验证功能正确性。软件测试的另一个目标是验证软件功能的正确性。测试人员通过执行各种测试用例，以确保软件能够按照预期执行其指定功能，满足用户需求和规格要求。

(3)评估软件质量。软件测试还能够对软件的质量水平进行综合评估。通过全面测试，可以获得关于软件质量的指标，如可靠性、健壮性和易用性等。这些指标可以为开发团队提供克服软件弱点的依据。

(4)提高开发效率。软件测试还有助于提高开发效率。通过尽早发现和修复问题，可以减少后期的调试和维护成本，从而提高开发团队的整体效率。

(5)验证需求。软件测试确保软件能够符合用户需求和规格要求。测试人员通过执行各种测试用例，确保软件的功能和性能符合预期，并满足用户的期望。

(6)提高可靠性。软件测试可以帮助发现和修复潜在的问题，从而提高软件的可靠性。通过对不同情况下的边界条件、异常情况和负载情况进行测试，可以发现隐藏的错误和瓶颈，并对其进行修复。

(7)优化用户体验。软件测试有助于提升用户体验。通过测试不同的使用场景和用户情境，可以发现用户界面的问题和交互逻辑上的问题，提升软件的易用性和用户友好性。

(8)保障安全性。软件测试在确保软件安全性方面发挥着重要作用。通过执行安全测试，可以发现潜在的安全漏洞和风险，并采取相应的措施来防范潜在的攻击和数据泄露。

在实际的软件测试过程中，通常会运用多种测试方法和技术，如黑盒测试、白盒测试、灰盒测试和自动化测试等，以提高测试效率和覆盖率，同时也能够更好地满足软件测试的目标。

1.3 软件测试的方法和技术

软件测试发展至今，已经形成一个庞大的系统工程，有许多常用的测试方法，如黑盒测试、白盒测试和自动化测试等，其种类和名称十分庞杂。因此需要对软件测试的方法进行类型上的划分，便于明确测试过程、完成哪些操作步骤以及需要达到的测试目标，尽量保证测试的完整性。按照不同的分类标准可以将软件测试分为很多不同的种类，如图 1-1 所示。

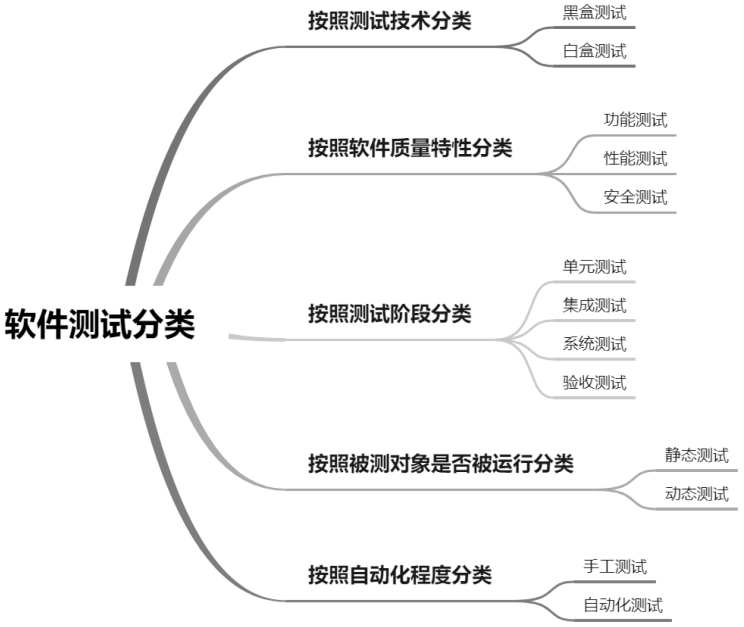

图 1-1 软件测试分类

1.3.1 黑盒测试和白盒测试

黑盒测试和白盒测试是软件测试中常用的两种方法，它们分别从不同的角度对软件进行测试。它们的主要区别在于测试人员在测试过程中需要对软件内部具体实现细节的了解程度不同，如图 1-2 所示。

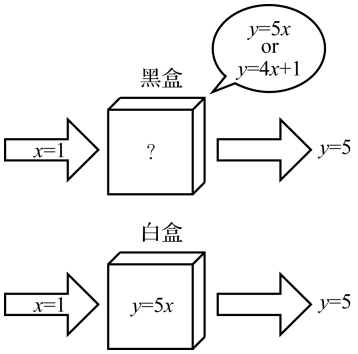

图 1-2 黑盒测试和白盒测试

1. 黑盒测试

黑盒测试是一种基于软件外部行为和规格要求的测试方法。在这种测试中，测试人员不需要了解软件内部的具体实现细节，而是仅根据软件规格或需求文档来设计测试用例，通过输入数据并观察输出结果来验证软件的正确性和功能完整性。

黑盒测试的特点如下。

(1)聚焦于软件的功能和规格要求，不关注其内部实现。

(2)不需要程序员的专业知识，测试人员可以独立进行测试。

(3)测试用例的设计主要基于等价类划分、边界值分析和错误推测等技术。

(4)旨在发现软件的功能问题，如输入验证、功能逻辑错误等。

(5)常用的技术和方法包括功能测试、界面测试、配置测试、性能测试和安全测试等。

2. 白盒测试

白盒测试是一种基于对软件内部结构、逻辑和代码的理解来设计测试用例并进行测试的方法。测试人员通过访问和分析源代码、控制流图及路径覆盖等信息，从内部的角度来评估软件的质量，从而揭示潜在的错误和缺陷。

白盒测试的特点如下。

(1)需要测试人员具备一定的编程和代码分析能力，能够理解软件的内部实现。

(2)能够深入软件的内部逻辑，如发现隐藏的边界条件、错误路径和异常情况。

(3)旨在检测软件的结构性问题，如逻辑错误、代码覆盖不全等。

白盒测试常用的技术和方法包括语句覆盖、决策覆盖、条件覆盖、路径覆盖和静态分析等。

知识拓展

灰盒测试

1.3.2　功能测试、性能测试和安全测试

当进行软件测试时，常见的测试类型包括功能测试、性能测试和安全测试，下面分别对其进行阐述。

1. 功能测试

功能测试旨在验证软件的各项功能是否均按照规格要求正确运行。这种测试通常基于软件的需求规格和功能规格文档来设计测试用例。

功能测试的主要目标如下。

(1)验证软件的功能是否符合用户需求和规格要求。

(2)检查软件的输入和输出是否正确。

(3)确保软件的各项功能能够稳定、正常地工作。

(4)发现并报告功能缺陷、逻辑错误和算法问题等。

功能测试涵盖以下方面的测试：基本功能、边界条件、异常情况、用户界面、数据输入验证、功能组合和功能交互等。

2. 性能测试

性能测试旨在评估软件在各种负载情况下的性能表现和可靠性。该类型的测试有助于

确定软件在正常和峰值负载下的响应时间、吞吐量、稳定性和资源利用率等方面的表现。

性能测试的关注点如下。

(1)测试软件在不同负载情况下的响应时间和吞吐量。

(2)确认软件在长时间运行时的稳定性和可靠性。

(3)评估资源的利用率，如内存、CPU 和网络带宽等。

(4)进行负载均衡和容量规划等。

性能测试包括负载测试、压力测试、容量测试、稳定性测试和配置测试等。

3. 安全测试

安全测试旨在评估软件系统的安全性，以发现潜在的安全漏洞和脆弱性，并确保软件对潜在攻击有足够的抵御能力。

安全测试的重点如下。

(1)检查软件的认证和授权机制，以防止未经授权的访问。

(2)发现和修复潜在的漏洞，如输入验证、数据处理和访问控制等。

(3)评估软件的加密和数据保护机制，以确保数据的机密性和完整性。

(4)进行安全漏洞扫描和渗透测试等。

安全测试包括网络安全测试、应用程序安全测试、数据安全测试和身份验证/授权测试等。

除了上述的功能测试、性能测试和安全测试，还有其他类型的测试，如兼容性测试、可用性测试和易用性测试等，这些测试类型针对软件的不同方面进行测试，以确保软件的质量并满足用户需求。在具体的测试过程中，可以根据软件的特点和需求选择合适的测试类型和方法。

1.3.3 静态测试和动态测试

静态测试和动态测试是软件测试过程中常用的两种方法，它们从不同的角度对软件进行评估和验证。

1. 静态测试

静态测试是在不运行实际软件的情况下，对软件的设计、文档和代码进行检查和分析的方法。它主要关注软件的静态属性，如结构、语法、规范和设计等，以发现潜在的问题和缺陷。

静态测试的特点如下。

(1)没有实际去执行软件代码。

(2)静态测试主要基于软件文档、源代码和模型来进行验证和评估。

(3)目标是发现潜在的缺陷、错误和安全漏洞等。

(4)静态测试可以包括代码审查、文档审查、模型验证、静态分析等技术和方法。

静态测试的优势在于可以尽早发现问题，从而有效降低缺陷修复的成本。它不仅能帮助我们发现设计的缺陷、逻辑错误、命名不规范和未使用的变量等问题，还能确保代码遵循既定的标准和最佳实践。静态测试通常需要开发人员、质量保证团队或专门的审查小组来执行。

2. 动态测试

动态测试是一种在运行实际软件时，对软件的行为和功能进行验证和评估的方法。该方法通过模拟用户的实际操作和使用场景，来检查软件的实际执行结果和输出是否符合预期。

动态测试的特点如下。

(1)需要执行实际的软件代码。

(2)动态测试基于输入数据和操作流程来验证软件的功能和行为是否符合预期。

(3)目标是发现实际运行中的错误、异常情况和功能问题。

(4)动态测试可以包括单元测试、集成测试、系统测试和验收测试等。

动态测试涵盖了软件的不同层次和功能，从单个函数或模块的单元测试到整个系统的集成验证。这一测试过程通常需要测试人员来设计和执行测试用例，以确保软件在正确性、稳定性和性能等方面均符合预期要求。

综上所述，静态测试和动态测试是软件测试过程中常用的两种方法。静态测试在不执行实际软件代码的情况下，通过检查和分析设计、文档和代码来发现问题；而动态测试是在执行实际软件代码的情况下，通过验证和评估软件的行为和功能来发现问题。这两种测试方法对于提高软件质量、发现潜在错误和确保软件符合预期都是至关重要的。在实际测试中，通常需要同时使用静态测试和动态测试两种方法，以提供更全面和有效的测试覆盖。

1.3.4　单元测试、集成测试、系统测试和验收测试

按照测试阶段可以将软件测试分为单元测试、集成测试、系统测试和验收测试。这种分类方式与软件开发过程相对应，目的是验证软件开发各个阶段的成果是否符合预期要求。

(1)单元测试。单元测试是软件开发的第一步测试，目的是验证软件单元是否符合软件需求与设计要求。单元测试大多是开发人员进行的自测。

(2)集成测试。集成测试是将已经被测试过的软件单元组合在一起以测试它们之间的接口，用于验证软件是否满足设计要求。在持续集成(continuous integration，CI)的场景中，常见的测试方式包括冒烟测试和回归测试。

①冒烟测试。冒烟测试这一术语最初源自电路板测试领域。当电路板制作完成后，首先会进行加电测试，若电路板未出现冒烟等异常情况，则再进行其他测试；否则，需重新设计后再次测试。后来，这种测试理念被引入软件测试领域。在软件测试中，冒烟测试是指软件构建版本建立后，对系统的基本功能进行简单的测试。这种测试重点验证的是程序的主要功能是否按预期工作，而不对具体功能进行深入测试。若冒烟测试未通过，则需要返回给开发人员进行修正；若测试通过，则再进行其他测试。因此，冒烟测试是对新构建版本软件进行的最基本测试。

②回归测试。回归测试是指修改了旧代码后，重新进行测试以确认修改是否引入新的错误或导致其他代码产生问题。回归测试在整个软件测试过程中占有很大的工作量比重，特别是在渐进和快速迭代的开发环境中，新版本的连续发布使得回归测试进行得更加频繁。因此，选择合适的回归测试策略以提高回归测试的效率和有效性是很有意义的。根据回归测试的特点，回归测试策略包括：全部重复测试和选择性重复测试。

(3)系统测试。系统测试是在完成集成测试之后，把软件部分与系统的其他组成部分，

包括硬件、外部软件、操作人员等结合起来，在实际运行或模拟环境下对计算机系统进行的一系列严格有效的测试，以发现潜在的问题。系统测试包括恢复测试、安全测试、压力测试等类型。

(4)验收测试。验收测试主要是对软件产品说明进行验证，逐行逐字地按照说明书的描述对软件产品进行测试，确保其符合用户的各项要求。

1.3.5 手工测试与自动化测试

随堂测试题 1

根据自动化程度的不同，可以将软件测试分为手工测试与自动化测试。手工测试是指测试人员逐条执行代码来完成测试工作。然而，手工测试相对耗时费力，并且在测试人员处于疲惫状态时很难保证测试的效果和准确性。相比之下，自动化测试是借助脚本、自动化测试工具等手段完成相应的测试工作，它也需要人工的参与。

1.4 常见的软件测试模型

软件测试和软件开发一样，都遵循软件工程原理。测试专家通过实践总结出了很多很好的测试模型。这些模型对测试活动进行了抽象，明确了测试与开发之间的关系，是软件测试管理的重要参考依据。下面介绍几种比较有代表性的模型。

1. 瀑布模型

传统的瀑布模型包括项目计划—需求分析—软件设计—程序开发—软件测试—集成维护这几个阶段。由于这个模型难以适应频繁变化的需求，现在已经不常用，但是模型中这几个阶段都很有代表意义，对后面的模型有一定的参考价值。

(1)项目计划阶段：主要是制订项目总体研发计划，树立项目里程碑节点，输出项目计划书。

(2)需求分析阶段：明确用户的需求定义，并对这个定义进行清晰地描述，是充分理解用户需求，描述产品功能的重要阶段，这个阶段会输出产品的需求规格说明书。

(3)软件设计阶段：会根据需求的定义来确定产品实现的方案，包括软件和硬件的结构定义，组建模块的实现方法，接口、界面和数据的组织方式，这个阶段会输出包括概要设计、详细设计在内的多份说明书。

(4)程序开发阶段：这个阶段是开发团队根据需求定义和设计要求具体实现产品，根据编程规范，构建组件模块，最后输出产品版本。

(5)软件测试阶段：这个阶段是通过独立的测试小组或 QA 团队评估产品是否满足需求定义，最后输出测试结果报告。

(6)集成维护阶段：产品经过测试后，交付给用户使用，开发人员根据用户的使用情况，对产品进行维护，并进行必要的修改和升级等操作。

2. V 模型

V 模型使用范围最广，它是从瀑布模型演化而来的，包括需求分析—概要设计—详细设

计—软件编码—单元测试—集成测试—系统测试—验收测试这几个阶段，如图 1-3 所示。

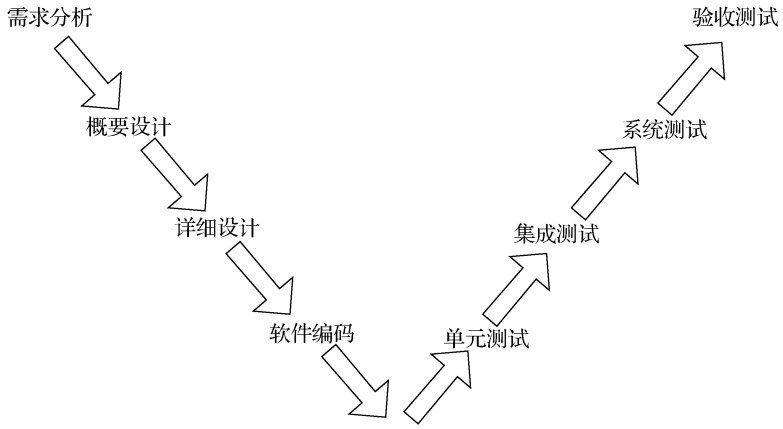

图 1-3　V 模型

V 模型按图 1-3 中箭头的方向指出了不同阶段的顺序和依赖性。每个阶段的分工和解决的问题不同。

(1)单元测试和集成测试：检测程序是否满足设计要求。

(2)系统测试：在功能、性能这些质量特性上是否满足系统要求的指标。

(3)验收测试：确定软件是否满足用户的需求及合同的规定。

3. W 模型

W 模型其实是双 V 模型，它的设计思想是并发执行软件测试与软件开发。开发与测试并行，有利于尽早发现问题，有利于及时了解项目的测试风险，及早地执行相应的应对方案，加快项目的开发进度。W 模型存在的缺点是整个开发阶段仍然是串行的，上一阶段未完全完成，无法进入下一阶段，不支持敏捷模式的开发。W 模型如图 1-4 所示。

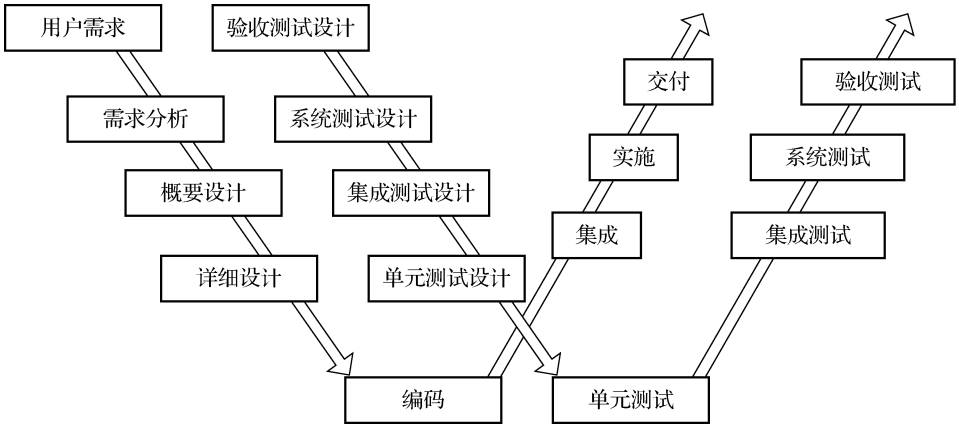

图 1-4　W 模型

4. X 模型

X 模型的基本思路是分离出多个程序片段，并对各片段独立地进行编码和测试，此后进行频繁的交接，再通过集成，最终合成可执行的程序。已经通过测试的程序可以进行封版提

交给用户，也可以作为更大集成的一部分。X 模型的右下方还定位了探索式测试，探索式测试是不进行事先计划的特殊类型的测试，能够帮助测试人员在测试计划之外发现更多的错误。X 模型如图 1-5 所示。

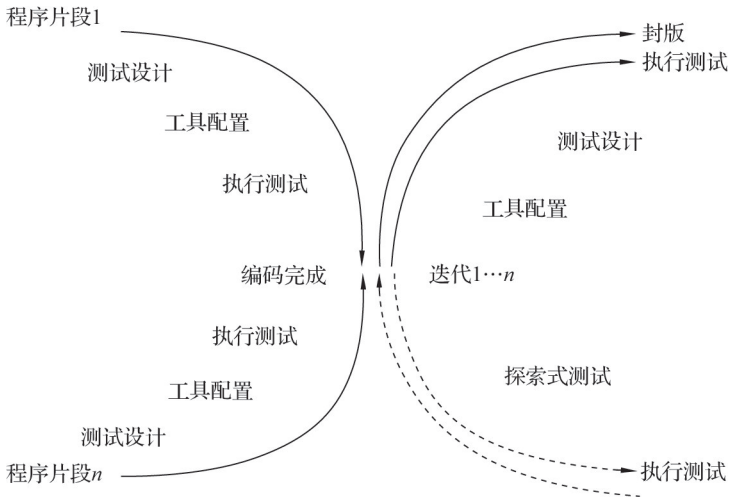

图 1-5　X 模型

5. H 模型

H 模型强调把测试分为测试准备和测试执行两个阶段，如果其他流程的进展使测试点准备就绪，测试执行活动就可以进行。在 H 模型中，测试是一个完全独立的模型，因此可以与其他流程交叉进行，也便于尽早地执行测试。H 模型如图 1-6 所示。

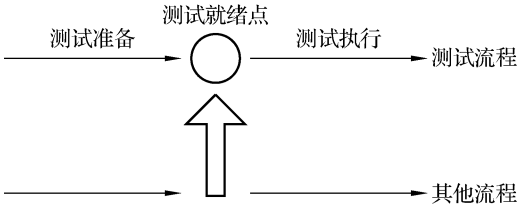

图 1-6　H 模型

随堂测试题 2

1.5 软件测试的原则

经过了多年的实践积累，人们为了尽可能地发现软件中的错误，提高软件产品的质量，总结出了一些基本原则用于指导软件测试工作。测试人员如果能把握好这些测试原则，就会以较少的时间和精力发现软件中存在的问题，以提高测试工作的效率和质量。

1. 测试应基于用户需求

所有的测试工作都应该建立在满足用户需求的基础上，从用户的角度来看，最严重的错误就是软件无法满足需求。应按照用户的需求配置环境，按照用户的使用习惯进行测试并评价结果。如果系统不能完成用户的需求和期望，那么这个系统的研发就是失败的。同时，

在系统中发现和修改缺陷也是没有任何意义的。在开发过程中用户的早期介入和接触原型系统就是为了避免这类问题发生的预防性措施。

2. 测试要尽早进行并不断迭代

由于软件的复杂性和抽象性，在软件生命周期各阶段都可能产生错误，所以不应把软件测试仅仅看作软件开发的一个独立阶段，而应当把它贯穿到软件开发的各个阶段中去。在需求分析和设计阶段就应开始进行测试工作，编写相应的测试计划及测试设计文档，同时坚持在开发各阶段进行技术评审和验证，这样才能尽早发现和预防错误，杜绝某些缺陷和错误，提高软件质量。尽早开展测试准备工作以使测试人员能够在早期了解到测试的难度，预测测试的风险，有利于开发人员制订出完善的计划和方案，提高软件测试及开发的效率，并规避测试中存在的风险。尽早开展测试工作有利于测试人员尽早发现软件中的缺陷，大大降低错误修复的成本。测试工作进行得越早，越有利于提高软件的质量，这是预防性测试的基本原则。

3. 穷举测试是不可能的

由于时间和资源的限制，穷举测试（各种输入和输出的全部组合）是不可能的，测试人员可以根据测试的风险和优先级控制测试的工作量，在测试成本、风险和收益之间取得平衡。此外，应避免冗余测试。

4. 遵循 Good Enough 原则

Good Enough 原则是一种权衡投入/产出比的原则：不充分的测试是不负责任的，而过分的测试则是一种资源的浪费，同样也是一种不负责任的表现。测试不充分无法保证软件产品的质量，但测试投入过多会造成资源的浪费。随着测试资源投入的增加，测试的产出也是增加的，但当投入达到一定的比例后，测试的效果就不会明显增强了。这个原则实施的困难之处在于如何界定什么样的测试是不充分的，什么样的测试是过分的。针对这种情况，测试人员最好制定最低测试通过标准和测试内容，然后具体问题具体分析。

5. 缺陷要符合“二八”定理

缺陷的“二八”定理就是在一般情况下，软件 80％的缺陷会集中在 20％的模块中，缺陷并不是平均分布的。存在这种现象的原因是：对一个程序模块进行测试，已发现的错误数越多，其中存在的错误概率也就越大。错误集中发生的现象可能与程序员的编程水平和习惯有很大的关系。因此，在测试时要抓住主要矛盾，如果发现某些模块比其他模块具有更多的缺陷，那么要投入更多的人力重点测试这些模块以提高测试效率。

6. 杀虫剂悖论

杀虫剂用得多了，害虫就会产生免疫力，杀虫剂就发挥不了效力。在测试中，同样的测试用例被反复使用时，发现缺陷的能力就会越来越差。这种现象的主要原因在于测试人员没有及时更新测试用例，同时对测试用例及测试对象过于熟悉，形成了思维定式。

要想避免这种情况，就要不断对测试用例进行修改和评审，不断增加新的测试用例，这样软件中未被测试过的部分或先前没有被使用过的输入组合就会被重新执行，从而发现更多的缺陷。同时，测试人员也要有探索性思维和逆向思维，不能只是做输出结果与期望结果的比较。

1.6 软件测试的一般流程

软件测试流程没有完全统一的标准，允许存在个性化差异。不同类型的软件产品测试的方式和重点不一样，测试流程也会不一样。即使是同类型的软件产品，不同的公司所制定的测试流程也会存在差异。尽管如此，软件测试所遵循的基本测试流程总体是一样的：评测测试需求—制订测试计划—设计测试用例—执行测试—编写测试报告。

1.6.1 评测测试需求

在软件测试前需编写一份软件需求规格说明书检查清单，目的是确保测试的全面性和准确性，以满足软件需求的验证和评估。注意，评测测试需求并不是需求分析，而是测试人员在制订测试计划之前先对软件需求规格说明书进行评测，按检查项逐项检查和判断，从而得出前期需求分析的过程和结果的评价，活动产出的结果是一张检查表，如表 1-1 所示。

表 1-1 软件需求规格说明书检查列表

序　号	检　查　项	检查结果	说　明
1	系统定义的目标是否与用户的要求一致	是【 】否【 】NA【 】	
2	系统需求分析阶段提供的文档资料是否齐全	是【 】否【 】NA【 】	
3	文档中的所有描述是否完整、清晰、准确地反映了用户要求	是【 】否【 】NA【 】	
4	与其他系统成分的重要接口是否都已经描述	是【 】否【 】NA【 】	
5	被开发项目的数据流与数据结构是否足够、确定	是【 】否【 】NA【 】	
6	所有图表是否清晰，在不补充说明时能否被理解	是【 】否【 】NA【 】	
7	主要功能是否已被包括在规定的软件范围之内，是否都已被充分说明	是【 】否【 】NA【 】	
8	软件的行为和它必须处理的信息、必须完成的功能是否一致	是【 】否【 】NA【 】	
9	设计的约束条件或限制条件是否符合实际	是【 】否【 】NA【 】	
10	是否考虑了开发的技术风险	是【 】否【 】NA【 】	
11	是否考虑过软件需求的其他方案	是【 】否【 】NA【 】	
12	是否考虑过将来可能提出的软件需求	是【 】否【 】NA【 】	
13	是否详细制定了校验标准，它们能否在系统定义成功时进行确认	是【 】否【 】NA【 】	
14	有没有遗漏、重复或不一致的地方	是【 】否【 】NA【 】	
15	用户是否审查了初步的用户手册或原型	是【 】否【 】NA【 】	
16	项目开发计划中的估算是否受到了影响	是【 】否【 】NA【 】	

检查表可以对需求说明的正确性、完整性和清晰性，以及其他指标进行评测，测试人员可从中发现不合理的地方并加以改进。对于有些需求不明确的地方，需要尽早与用户进行沟通，消除分歧。明确测试对象及测试工作的范围和测试重点，为后续制订测试计划做好准备。检查表的设计可以从以下几个方面考虑。

(1)确保需求理解的一致性。软件需求规格说明书是软件开发过程中的重要文档，描述了软件的功能、性能、界面和其他需求。编写一份检查清单可以确保测试团队对需求的理解与其他相关方的一致性，以避免因误解、遗漏或不一致而导致的测试问题。

(2)验证需求可测性。检查清单可以帮助测试团队验证需求是否具备可测性，即需求是否能被明确、准确地转化为可量化和可测量的测试条件和测试用例。这有助于确保测试团队能够有效地评估软件是否满足需求。

(3)检查需求的完整性和一致性。通过检查清单，测试团队可以逐个验证需求是否具备完整性和一致性。它可以帮助识别遗漏的需求、需求之间的冲突或矛盾，以及需求的合理性和可行性。这有助于确保测试团队完整地覆盖所有相关的需求。

(4)确保需求追踪和变更管理。检查清单还有助于追踪和管理需求的变更。它可以帮助测试团队识别已变更的需求、新增的需求、已删除的需求或任何其他需求变更。这有助于测试团队调整测试策略和计划以适应需求变更。

1.6.2　制订测试计划

测试工作是软件开发过程中不可缺少的组成部分，在软件开发工作的开始阶段就要制订好软件测试计划，作为软件开发的保障性工作，它涉及的内容很多，而且不是一成不变的，随着项目推进或需求变更，测试计划也会不断地发生改变，因此，测试计划的制订是一个随着项目发展不断调整、逐步完善的过程。

测试计划一般要包括以下内容。

(1)确定测试内容。列出包括哪些子系统及其下级各个模块，对这些模块的测试点及优先级都要进行准确说明。

(2)制定测试规则。设计测试活动的前置进入准则、暂停/退出准则，描述采取的测试流程(如黑盒或白盒)，采用的测试手段(如手工测试、自动测试或两者相结合)，测试要点、测试工具等。

(3)设定测试环境。描述测试的软件、硬件、网络通信、安全性要求及其他可能的环境需求。

(4)安排测试任务。根据软件开发计划、产品的整体计划来安排测试工作的进度，同时还要考虑各部分工作的变化。在安排工作进度时，最好在各项测试工作之间预留一个缓冲时间以应对计划变更。

(5)计划实施。列出系统各测试阶段所使用的资源及资源如何安排。

(6)风险管理。详细描述测试所面临的各类风险(包括人力资源、测试技术、测试资源和质量保障)及相应的建议及解决办法。

1.6.3 设计测试用例

1. 目的和作用

编写测试用例主要是为了进行软件的测试，发现潜在的问题和缺陷，并评估系统的性能、功能和质量。这是测试过程中重要的一环，对于确保系统的质量和稳定性至关重要。

2. 测试用例编写规范

1）常用字段

（1）用例编号。由名称和数字构成，具有唯一性。

（2）用例标题。为测试用例赋予一个有意义的标题，并要简洁明了。

（3）前置条件。指明执行测试用例所依赖的前置条件，在执行用例之前需要满足的环境或数据设置。

（4）输入。指明输入参数和约束条件。

（5）测试步骤。指明测试的具体步骤和操作，以及所需的输入和预期输出。

（6）预期结果。明确用例执行后预期得到的结果或期望的行为。

（7）执行结果。记录实际执行测试用例后的结果，包括实际输出和实际行为是否符合预期。

（8）是否通过。即测试是否通过的状态标记，如“通过”“失败”“跳过”等，以便于跟踪和管理用例状态。

一般将测试用例设计成表格形式，如表 1-2 所示。

表 1-2 测试用例示例

用例编号	用例标题	前置条件	输入	测试步骤	预期结果	执行结果	是否通过
TC-1001	登录功能	登录页面正常显示	输入登录信息，包括：账号、密码	1. 正常打开登录页面。 2. 输入正确的登录信息	登录成功，以该账号跳转到系统管理界面	登录成功，并在管理页面显示该账号信息	通过

以上测试用例字段并不是一成不变的，项目团队可根据项目的实际需求和团队的实践经验进行适当调整和补充。编写规范的测试用例有助于跟踪测试进度和结果。

2）自动化测试

如果使用自动化测试工具或平台，可将数据参数化，即针对需要多次重复执行的用例，尽量使用参数化的方式来提高测试的覆盖度。将测试数据和参数从测试用例中分离，减少用例的重复编写。

例如，要测试用户登录功能，可以在测试前准备好测试数据集，如表 1-3 所示。

表 1-3 测试数据集

序　号	账 号 名	密　码	邮　箱	角　色
1	zhangsan	zs123456#	zhangsan@163.com	资产管理员

续表

序　号	账 号 名	密　码	邮　箱	角　色
2	xiaowang	xw123456＃	xiaowang@126.com	资产管理员
3	liming	mm123123＃	liming@126.com	资产管理员
4	chengang	cc333666@	chengang@qq.com	系统管理员

然后在自动化测试脚本中读取测试数据，按顺序自动读取所需字段填入测试用例中对应的参数。目前如 LoadRunner、JMeter 和 Selenium 等测试平台均支持数据驱动自动化测试功能。

3）正确性和完整性

测试用例为了达到正确性和完整性，要尽可能覆盖系统的所有功能和边界条件，使测试用例兼具广度和深度。在测试过程中，应通过不断的反馈和重复执行来发现并弥补测试用例中可能存在的遗漏或不足。

（1）测试用例要满足正确性，参照以下做法。首先，应该基于清晰、详细的需求规格来编写。仔细分析和理解需求，确保所有功能需求和业务场景都有相应的测试用例覆盖。其次，按照真实的业务场景设计测试用例。通过了解用户的实际使用场景，编写涵盖常见和典型的业务场景的测试用例。

（2）测试用例要满足完整性，参照以下做法。首先，完整分析需求，对系统的每个功能点编写相应的测试用例。确保每个功能都有一个或多个测试用例，覆盖正常、异常和边界条件。其次，关注功能的边界条件，即输入的上下限或边界值。编写针对边界条件的测试用例，以验证系统在边界处的行为和处理能力。再次，推测用户错误，尝试推测用户可能犯的错误以及常见的错误情况。编写测试用例来模拟这些错误情况，以验证系统对错误的处理和恢复能力。

4）可读性

测试用例应具备良好的可读性和可维护性，使得其他人能够轻松理解和执行这些用例。采用清晰的语言描述和良好的结构化，避免冗余和重复。

5）持续更新

随着项目的进行和需求的演化，测试用例需要持续更新和维护，保持与实际系统的一致性。

3. 测试用例的管理

测试用例的管理是测试过程中非常重要的一环，它可以帮助团队有效组织、跟踪和执行测试工作。通常可以使用 Excel 电子表格管理测试用例，在前面已经举例说明。这类非专业的测试用例管理软件存在一些优点和缺点。其优点是简单易用，可批量操作数据。缺点为：基于文件形式存储，分布零散；无法有效统一用例规范；版本管理混乱，可追溯性差；缺少对团队协作支持。

可以使用专门的测试用例管理工具，如 TestRail、TestLink 和 Zephyr 等，方便进行在线操作和版本管理，一些项目管理工具也具有类似功能。测试用例管理工具一般具有以下功能。

（1）测试用例管理。提供了用例创建、编辑、执行和跟踪的功能，并可以生成测试报告和

统计指标。

(2)用例库和目录结构。建立用例库和目录结构，将测试用例按照模块、功能和场景进行组织和分类。确保每个用例都有唯一的标识符，方便查找、引用和管理。

(3)用例属性和标签。给测试用例定义一些属性和标签，如优先级、状态和负责人等，以便筛选和排序用例。标签可以基于不同的维度进行分类，如功能、优先级和风险等。

(4)版本控制。使用版本控制工具对测试用例进行版本管理。确保每个版本的用例都有对应的标签和注释，方便跟踪用例的变更历史和回溯。

(5)用例批量导入和导出。测试用例可以批量导入测试管理工具中，以便快速建立用例库。同时，也可以将用例导出为 Excel 或 CSV 等格式，方便备份、共享和跨系统使用。

(6)用例执行和跟踪。在测试执行过程中，记录每个用例的执行结果和状态，包括通过、失败和跳过等。及时更新用例的执行情况，方便跟踪和统计测试进度和结果。

(7)用例评审和审查。定期进行用例评审和审查，确保测试用例的准确性和完整性。通过团队的协作和讨论，对用例进行改进和优化。

(8)用例维护和更新。随着系统需求的变化和产品迭代，测试用例需要不断维护和更新。及时处理用例中的缺陷、改进需求和变更请求，保持用例的可用性和可靠性。

(9)用例复用和参数化。鼓励用例复用，避免重复编写相似的测试用例。将通用的测试步骤和数据进行提取和封装，以便于复用和维护。

(10)用例报告和统计。生成测试报告和统计指标，对测试用例执行情况进行可视化展示。报告可以包括用例通过率、失败率和执行时间等指标，这些指标有助于项目管理和决策。

因此，通过测试用例管理方法和工具，测试团队可以更好地组织和执行测试工作，提高测试效率和成果。读者可以参考上述的功能介绍，根据自身需求选择合适的管理工具。

1.6.4 执行测试

执行测试就是按照测试用例进行测试的过程，这是测试人员最主要的活动阶段。在执行测试时要根据测试用例的优先级进行。执行测试过程看似简单，测试人员只要按照测试用例完成测试工作即可，但实际并非如此。测试用例的数目非常多，测试人员需要完成所有测试用例的执行，每一个测试用例都可能会发现很多缺陷，测试人员要做好测试记录与跟踪，衡量缺陷的质量并编写缺陷报告。

当提交后的缺陷被开发人员修改后，测试人员需要对其进行回归测试。如果系统对测试用例产生了缺陷免疫，测试人员则需要编写新的测试用例。在单元测试、集成测试、系统测试和验收测试各个阶段都要进行功能测试、性能测试等，这个工作量无疑是巨大的。除此之外，测试人员还需要对文档资料(如用户手册、安装手册、使用说明等)进行测试。因此，不要简单地认为执行测试就是按部就班地完成任务，可以说这个阶段是测试人员最重要的工作阶段。

1.6.5 编写测试报告

测试报告是对一个测试活动的总结，对项目测试过程进行归纳，对测试数据进行统计，对项目的测试质量进行客观评价。不同公司的测试报告模板不同，但测试报告的编写要点都是一样的，一般都是先对软件进行简单介绍，然后说明这份报告是对该产品的测试过程进

行总结，对测试质量进行评价。

一份完整的测试报告必须包含以下几个要点。

(1)引言。描述测试报告编写的目的、报告中出现的专业术语解释及参考资料等。

(2)测试概要。介绍项目背景、测试时间、测试地点及测试人员等信息。

(3)测试内容及执行情况。描述本次测试模块的版本、测试类型、使用的测试用例设方法和测试通过覆盖率，依据测试的通过情况提供对测试执行过程的评估结论，并给出测试执行活动的改进建议，以供后续测试执行活动借鉴参考。

(4)缺陷统计与分析。统计本次测试所发现的缺陷数目、类型等，分析缺陷产生的原因，给出规避措施等建议，同时还要记录残留缺陷与未解决的问题。

(5)测试结论与建议。从需求符合度、功能正确性、性能指标等多个维度对版本质量进行总体评价，给出具体、明确的结论。

1.7 企业测试举例

泉州银行自成立至今已有二十多年历史，是本书编写过程中的合作企业。银行软件系统十分复杂，包含几十个乃至上百个子系统。每个业务流程通常需要跨越多个子系统来完成，因此，确保各模块功能的正确性以及系统整体的稳定性显得至关重要。随着业务的不断扩展和新功能的不断开发，泉州银行对软件测试领域的需求日益增长，同时对软件测试的严格性要求也不断提升。

为了高效推进软件测试工作，泉州银行信息技术部下设需求与测试中心，专职负责软件技术的测试工作。该中心下设系统集成测试组、性能测试组及自动化测试组等。各组的职责分别是：系统集成测试组负责测试需求的深入分析，通过等价类划分、边界值分析和判定表等技术测试手段，按照需求对软件进行全面测试；性能测试组负责系统的性能评估与测试；自动化测试组负责开发与维护自动化测试脚本，旨在提高测试效率和覆盖率。

通过合理的部门划分和科学规范的工作流程，泉州银行能够高效地进行系统集成测试、性能测试任务，确保系统稳定、安全地运行。

图 1-7 是在银行现场拍摄的照片，可见员工统一着装，十分整齐。

图 1-7　泉州银行信息技术部的需求与测试中心工作现场

小结

本模块主要介绍了软件测试的基础知识，首先介绍了软件测试的发展过程；其次介绍了软件测试的定义和目标；接着介绍了软件测试的方法和技术，如黑盒白盒测试、常见的软件测试模型及软件测试原则；并比较详细地讲解了软件测试的一般流程和编写测试用例的规范做法。读者在学习完本模块的知识之后，能对软件测试有比较直观的认识，也能为其后续学习打下基础。

思考与练习

一、单选题

1. 按照测试阶段分类，（　　）不属于该分类。

A. 单元测试　　B. 冒烟测试

C. 集成测试　　D. 验收测试

2. 下列说法错误的是（　　）。

A. 在软件测试中，冒烟测试是指软件构建版本建立后，对系统的基本功能进行简单的测试，这种测试重点验证的是程序的主要功能

B. 系统测试：将经过测试的软件在实际环境中运行，并与其他系统的成分组合在一起进行测试

C. 回归测试是指把已经测试过的软件单元进行组合，重新进行测试

D. 验收测试主要是对软件产品说明进行验证，确保其符合用户的各项要求

3. 下列说法错误的是（　　）。

A. 黑盒测试要了解程序内部是如何实现的

B. 白盒测试在测试时，按照程序的执行路径得出结果

C. 相较于黑盒测试，白盒测试对测试人员的要求会更高一点

D. 回归测试策略按覆盖程度包括全部重复和选择性重复

4. （　　）是常用的测试模型。

A. 瀑布模型　　B. V 模型

C. W 模型　　D. H 模型

5. 软件测试原则包括（　　）。

①测试应基于用户需求；②测试要尽早进行；③穷举测试是不可能的；④遵循GoodEnough原则；⑤缺陷要符合“二八”定理；⑥避免缺陷免疫。

A. ①②③④⑤⑥　　B. ⑤⑥

C. ①②④⑤⑥　　D. ①②③④

二、填空题

1. 在执行测试需求评测时，测试人员一般会根据软件开发需求文档制作_____。

2. 一份完整的测试报告必须包含以下几个要点：_____、_____、_____、_____、_____。

3. Bill Hetzel 在《软件测试完全指南》（*Complete Guide of Software Testing*）一书中指

出测试的定义：_____。

4.1983 年，IEEE 指出软件测试再也不是一个一次性的、开发后期的活动，而是_____。

5.软件测试的必要性包括_____、_____、_____、_____、_____。

三、简答题

1.简述软件测试的一般流程。

2.如何制订测试计划？测试计划的主要作用是什么？