国家技能型紧缺人才培养培训工程
高职高专软件技术专业规划教材

软件测试技术

徐 芳 主编

机械工业出版社
CHINA MACHINE PRESS

国家技能型紧缺人才培养培训工程
高职高专软件技术专业规划教材

软件测试技术

主编　徐　芳
参编　李　恒
主审　王　伟

机械工业出版社

本书根据软件测试教学的需要，结合读者对象未来的职业要求和定位，除了尽力全面阐述软件测试技术基本概念外，采取了计划、设计与开发、执行这样的工程步骤来描述软件测试的相关知识，使学生在学习软件测试的技术知识时，能够同时获得工程化思维方式的训练。

本书共7章。第1章介绍软件测试的基本知识；第2章介绍如何制定软件测试计划；第3章介绍测试用例的设计和相关技术；第4章介绍执行测试中相关技术和方法；第5章介绍实际工作中各种测试方法；第6章介绍MI公司的一套测试工具的使用，包括功能、性能和测试管理工具；第7章通过一个实例，给出了完整的与软件测试相关的文档。

本书内容充实、实用性强，可作为高职高专院校计算机软件专业软件测试技术课程的教材，也可作为有关软件测试的培训教材，对从事软件测试实际工作的相关技术人员也具有一定的参考价值。可免费提供电子教案，请发电子邮件至 wangyx@mail.machineinfo.gov.cn。

图书在版编目（CIP）数据

软件测试技术/徐芳主编 .—北京：机械工业出版社，2006.1
高职高专软件技术专业规划教材
ISBN 7-111-18049-6

Ⅰ.软 … Ⅱ.徐 … Ⅲ.软件－测试－高等学校：技术学校－教材
Ⅳ.TP311.5

中国版本图书馆CIP数据核字（2005）第145998号

机械工业出版社（北京市百万庄大街22号　邮政编码100037）
责任编辑：王玉鑫　版式设计：霍永明　责任校对：李秋荣
封面设计：鞠　杨　责任印制：杨　曦
北京机工印刷厂印刷
2006年1月第1版第1次印刷
787mm×1092mm 1/16·19.25印张·477千字
0 001—4 000册
定价：27.00元

前　言

本教材是为适应高职高专院校计算机软件专业软件测试技术课程教学需要而组织编写的。

软件测试在当前的软件开发中作为保证软件质量的一个重要手段越来越受到人们的重视。软件测试人员并不是那些反复使用软件，试图在频繁操作中寻找到错误发生的低层次的人员。软件测试是一门需要独特技术的专门学科。软件测试并非只凭直觉，测试人员必须学会如何进行测试。许多人错误地认为任何一个程序员都可以测试软件，如果你会编程，你就会测试，其结果造成测试工作效率低下。

随着我国信息产业的发展，产品的质量控制和质量管理将成为企业生存与发展的核心。然而，目前一方面企业对高质量的测试人才需求量逐步增加，另一方面国内原来对测试工作不够重视，许多人不了解测试工作，同时学校中相关课程的学习相对比较欠缺，计算机专业的毕业生在大学中没能学到足够的软件测试知识。

高职高专院校的在校学生可能在头脑中从未形成对软件测试的认识，希望本书能对他们在软件测试方面提供帮助。书中系统全面地阐述了软件测试技术所涉及的基本概念。除此之外，增加了测试工作中实际问题的分析和应对，力求使教材面向应用，使学生通过学习能够理解软件测试工作并具备相关的基本技能。

如何确定本书描述的主线，是一个需要深思的问题。在编写本书时，考虑到本书的读者对象，期望读者在学习完本书后能够掌握软件测试（特别是系统测试）的基本知识。另外，也期望读者能够逐步建立起工程化的思维方式。因此，本书的主线是按照工程活动的一般步骤来确定的，期望读者能够在掌握软件测试知识的同时，也能够掌握在本书中描述的“计划”、“设计与开发”、“执行”这样三个基本的工程步骤。

事实上，测试技能的提高不可能依靠一本书来完成，对于读者来说，需要不断地学习新的内容，不断地积累实践经验，而工程化的思维方式，将能够帮助读者迅速地吸纳新的知识，不断地基于实践经验而总结和提高。

本书由徐芳编写第 1 章、第 3～4 章、第 6～7 章，李恒编写第 2 章、第 5 章，由王伟主审。在本书的编写过程中得到了王伟的大力支持和帮助，他提出了很多宝贵的意见，在此表示衷心的感谢，同时感谢王雷和葛永明两位老师对本书编写工作的支持。

限于作者的水平，加之时间仓促，书中难免有缺点和不妥之处，欢迎读者和同行批评指正，以利改进和提高。

作　者

目　录

第 1 章　软件测试基本知识

学习目标

阅读本章后，你应该具备如下能力：

- ✓ **对于软件开发过程和软件质量保证建立基本概念。**
- ✓ **理解软件测试工作的作用。**
- ✓ **理解软件测试的目标。**
- ✓ **了解软件测试工作的基本工作流程。**
- ✓ **了解软件测试的各种分类方式。**
- ✓ **理解软件测试人员的职业要求。**
- ✓ **了解软件测试人员的职业前景。**

本章要点

本章介绍软件测试的基本知识。作为初入门的软件测试人员，首先要理解软件测试工作的重要性和软件测试工作的基本内容。软件测试是整个软件开发工作中的一个重要环节，在学习软件测试时，首先要建立对软件开发过程的了解，同时要了解什么是软件质量和软件质量保证。在本章中，以一个小程序为例，介绍了软件测试的基本工作，以帮助读者建立对于软件的理解。本章还介绍了软件测试的分类和软件测试的基本流程。

作为初入门的软件测试人员，需要很好地了解软件测试人员的能力要求和职业前景。只有这样，才能通过不断地学习来提高自己，从而成长为优秀的软件测试人员。

1.1　软件开发过程

软件测试是软件开发过程中的一个重要组成部分。当一个项目组开发一个软件时，需要遵照一系列步骤来进行，这些步骤构成了软件开发过程。在软件开发过程中的每个步骤，都应该有明确的输入、输出和实施方法，有时候，一个步骤会被进行更加详细地分解，形成一系列子步骤。

软件开发过程应该是怎么样的呢？软件开发过程和一些常见的工程活动（如建造房屋、修建公路）是很相似的，概括地说，软件开发过程需要经历这样几个主要的阶段：

1）定义。明确软件开发的目标、软件的需求。

2）计划。制订软件开发所涉及的各种计划。

3）实现。进行设计、编码、文档编写工作，完成所要求开发的软件特性。

4）稳定化。以测试和缺陷修复工作为主，确保将提交的软件具有良好的质量。

5）部署。安装、提交开发完成的软件，建立可供用户使用的环境。

有时候，人们会把开发与实现阶段作为两个阶段（设计阶段、编码阶段）来看待，也可

能会使用不同的名词去描述这些阶段。但无论描述方式如何，其本质内容都是一致的。

另外，人们还经常使用“软件生命周期”这样一个词，那么什么是“软件生命周期”呢？前面我们介绍了软件开发过程的五个主要阶段，在软件开发过程结束后，软件还需要经历一个使用、维护，直至被停止使用的阶段。加上这个阶段之后，就形成了一个软件从诞生到最后被停止使用的完整周期，因此软件生命周期是包括了定义、计划、实现、稳定化、部署、运行与维护这样六个阶段的过程。

从20世纪70年代开始，人们就在不断地研究软件开发过程，到现在已经形成了一系列过程模型（也称为软件生命周期模型），当一个项目组在实施一个软件开发时，将会从这些生命周期模型中，选择最适合自己的一种来使用（当然，很多时候会根据项目实际情况，对这个模型加以剪裁）。

下面是一些常见的软件生命周期模型：

- 瀑布模型。
- 原型模型。
- 增量模型。
- 螺旋模型。

接下来，我们就对这几种软件生命周期模型进行概要介绍。那么，软件测试人员为什么要了解软件开发过程呢？软件测试工作是属于软件开发工作的一个部分，在软件开发过程中，包含着需求分析、设计、编码、文档和测试等过程，软件测试过程并不是和软件开发过程并列的两种工作过程，而是“隶属”于软件开发过程。因此，作为软件测试人员，需要了解软件开发过程的全貌，这样才能有利于对软件测试过程和方法有更好的理解，并能够清楚自己在软件开发过程每个阶段中所起的作用和所担负的职责。

1.1.1 瀑布模型

在瀑布模型中，包括了六个阶段：计划、需求分析、设计、编码、测试、运行维护。这六个阶段自上而下、相互衔接，以固定的次序来进行。瀑布模型的一个主要特征是强调阶段的顺序性和依赖性，即下一个阶段的开始必须以上一个阶段的完成为前提条件，例如在开始设计工作前，必须完成需求分析过程。此外，瀑布模型要求各个阶段必须有相应的文档作为审查的依据。因此瀑布模型是以文档驱动的，各个阶段有清晰的划分。图1-1描述了瀑布模型。

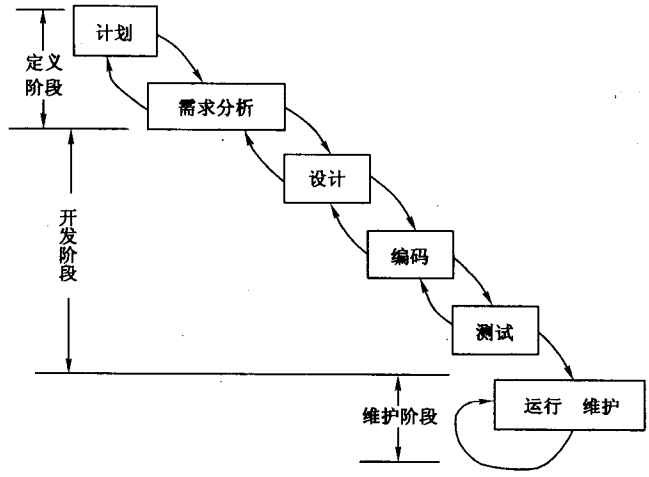

图1-1 瀑布模型

瀑布模型存在自身的一些缺点，一个典型的问题是许多系统要在系统开发的后期才能获得完整的需求，因此人们认为瀑布模型适合硬件系统开发，而软件开发是一个创造性过程而不是制造过程，几乎很难想象在软件开发的早期能够精确地定

义系统需求，其他模型，如螺旋模型、演化原型模型等更适合软件产品的开发。采用瀑布模型进行软件开发，测试人员可能在开发的后期发现大量错误，因此必须返回需求分析、设计或代码中定位问题，而瀑布模型中认为已完成的各阶段必须修改，其结果是昂贵的代价。

尽管瀑布模型存在这些缺点，但它包含了软件开发所必须的各个阶段，如软件开发需要从理解用户要求开始进行需求分析和设计，开发过程包括设计、编码、测试等活动，其他模型中也都包含这些活动，只是这些活动不像瀑布模型中以线性的方式组织起来。因此瀑布模型仍然值得学习研究。

在瀑布模型中，测试工作是在测试阶段比较集中地进行的。在瀑布模型中，设计阶段可以被更细地分解为概要设计和详细设计阶段，测试阶段也可以被更加细地分解为单元测试、集成测试、系统测试阶段，每个阶段都有不同的工作内容和工作目标。

1.1.2 原型模型

在很多时候，用户提出了软件需达到的一系列目标，但不能给出详细的输入、输出和处理过程；开发人员不能确定某种算法（解决方案）是否有效、所设计的人机交互方式和过程是否合适。在这种情况下，可以使用原型模型。图 1-2 表示了原型模型。

其主要思想是：先建立一个能反映用户需求的原型系统，使得用户和开发者可以对目标系统的概貌进行评价和判断，然后对原型进行反复的扩充、改进和求精，最终建立符合用户需求的目标系统。

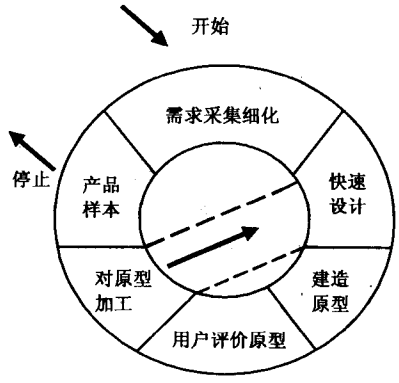

图 1-2 原型模型

原型模型从需求收集开始，这个时候所收集到的需求可能是局部的、不够详细的。然后开发者和用户一起定义软件的总体目标，识别出已知的需求，并规划出哪些内容需要进行进一步的定义。在这个工作的基础上，开发者对已知的部分进行设计和开发，从而构造出一个“原型”。接下去，用户对原型进行评估，在评估的基础上，开发者和用户得到更加详细的待开发软件的需求，对已经开发的原型进行调整，使之更加符合用户的需求。通过不断的迭代，最终开发出符合用户需求的软件。

这个时候，软件开发过程实际上被分解为一个一个的原型的开发，而测试人员则需要加入到对每一个原型的开发中，而不是等最后一个原型开发出来后再开始测试。当然，每个原型的目标和质量要求是不一样的，在很多时候，第一个原型版本是被抛弃的（这被称为“抛弃型原型”），测试人员在每个原型中投入的工作量、测试的目标也将有所区别。

1.1.3 增量模型

在进行商业软件开发时，往往会面对紧迫的市场期限，而在这个市场期限内，很难完成一个完善的软件产品。比如 Windows 的开发，相对于 Windows 3.x 而言，Windows 95 是一个革命性的产品，在 Windows 95 之后，微软公司又陆续发布了 Windows 98、Windows 2000、Windows XP，每一个版本都比上一个版本更加完善。假如微软公司试图在 Windows 3.x 之后发布类似于今天 Windows XP 这样的产品，会面临什么问题呢？当时没有足够的硬件支持是

其中的一个因素，但一个更加重要的因素将是：如果想一口气开发出 Windows XP，那么需要做一个十年的开发计划，而在这十年中，市场早就发生了变化，由于没有迅速推出下一代产品，也许微软公司会在这十年中不复存在。所以，要想在一开始就推出一个完善的软件并不现实。在这种情况下，就需要开发者渐进地开发逐步完善的软件版本。

增量模型结合了瀑布模型和原型模型的特性，如图 1-3 所示。

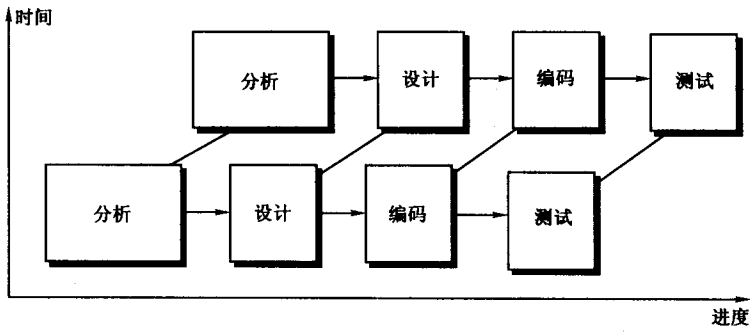

图 1-3　增量模型

增量模型中，在每个阶段都生成软件的一个可发布版本。这些阶段是交错进行的，这意味着前一个版本还没有发布时，下一个版本的部分工作就已经开始了。

增量模型和原型模型有一个很大的差别：在原型中，每个阶段是发布一个原型，而在增量模型中，是完成一个正式的版本。因此，对于测试人员而言，如果面对的是原型模型，那么对每个原型的测试内容、质量要求可能会有所区分，但如果面对的是增量模型，在不同的增量版本之间，质量要求几乎没有区别。

在增量模型中，软件版本是逐步完善的。但作为测试人员要注意到，这里所说的“逐步完善”，更多地是指在功能上的逐步完善，而不是质量上的逐步完善。

1.1.4　螺旋模型

对于复杂的大型软件，开发一个原型往往达不到要求。螺旋模型将瀑布模型和快速原型模型结合起来，并且加入了两种模型均忽略了的风险分析，弥补了两者的不足。螺旋模型沿着螺线旋转，如图 1-4 所示。

螺旋模型的每一周期都包括制定计划、风险分析、实施工程和评审四个阶段。开发过程每迭代一次，螺旋线就增加一周，软件开发又前进一个层次，系统又生成一个新版本，而软件开发的时间和成本又有了新的投入，最后得到一个客户满意的软件版本。

在螺旋模型中，软件也同样是以一系列版本的形式逐步发布的。螺旋模型和一些传统的过程模型不同，它不是当软件交付用户后就结束了，而是贯穿了软件的整个生命周期，在螺旋模型中，可能会包括了产品增强（升级）项目和产品维护项目。

1.1.5　小结

前面介绍了软件开发过程的主要阶段，也介绍了几种常用的软件生命周期模型。除了上面介绍的四种模型之外，还有一些其他的生命周期模型。在不同的模型中，软件测试人员的启动测试工作的时间、在不同时间的工作内容会有所差别。但其主要的工作内容、工作目标

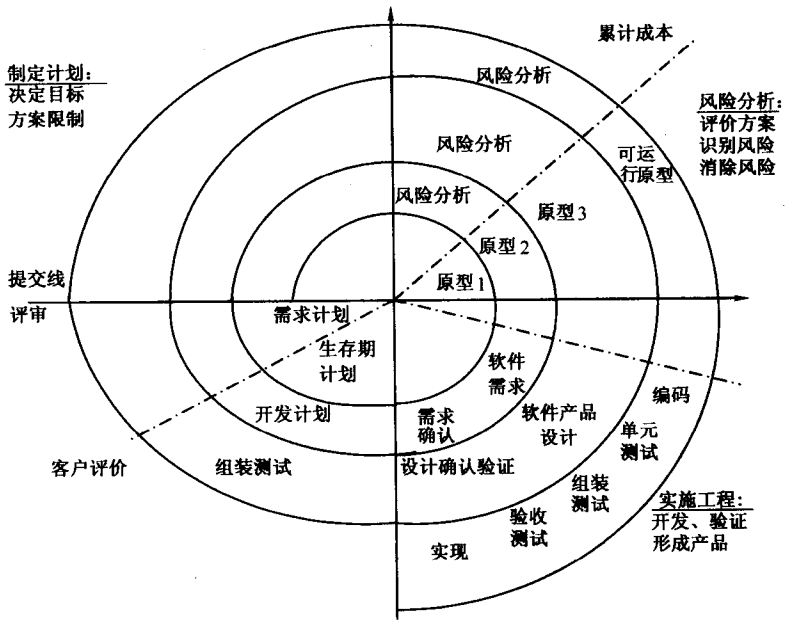

图 1-4　螺旋开发模型

是一致的。

从软件测试人员的角度来看软件开发过程，需要注意的是：测试贯穿在整个开发过程中，而不是在某个阶段集中地做一下测试，而其他阶段不用理会测试工作。

现代的软件测试不仅仅是在软件设计、编码完成以后来做测试工作，而是将测试渗入到软件开发的各个阶段。即使是在瀑布模型中，从表面上看，测试工作是集中在测试阶段来进行的，但在计划、需求和设计阶段，测试人员已经开始了测试方面的工作，如了解软件需求、编写测试计划、搭建测试环境。

前面介绍了几种典型的软件生命周期模型。在每种模型中，我们都可以发现，软件的开发过程实际上都是由一系列步骤来组成的，这些步骤都有不同的工作目标，有不同的输入和输出，也有不同的工作方式。从软件测试的观点来看，软件测试人员在不同的步骤中所做的工作也是不相同的，也将具有不同的工作目标。

1.2　软件质量保证

1.2.1　软件质量的定义

首先我们先来回答下面的问题：

如果公司一直为一个客户连续不断地开发产品，而且问题也连续不断的重复出现，你觉得公司还会拿到订单吗？就像一家饭馆的饭总是有沙子，你会一直订他们的饭吗？

从上面的问题可以看到，质量是客户的基本要求。

在软件开发中，如果客户提供了一份详细的需求规格说明，里面准确地描述了他的需

求，在这种情况下，质量意味着符合客户的规格说明。但大多数的软件开发者都不会遇到具备专业知识并且表达准确的客户。对于开发者而言，衡量产品质量和服务质量的标准，就是其客户的满意程度，而不是符合某个规格说明书。

什么是软件的质量？软件质量与传统意义上的质量概念并无本质差别，只是针对软件的某些特性进行了调整。从一般意义而言，质量通常会被定义为“无缺陷”。进一步讲，如果企业是以顾客为中心的，那么通常是根据顾客满意来定义质量：“如果顾客不喜欢，那该产品就是有缺陷。”

一个软件之所以被认定为质量优秀，并不是因为它获得了一个奖项，而是它的内在具备了这样一些特性：

- 满足用户的需求。
- 合理进度、成本、功能关系。
- 具备扩展性和灵活性，能够适应一定程度的需求变化。
- 能够有效地处理例外的情况。
- 保持成本和性能的平衡。

其中满足用户的需求是最重要的一点，一个软件如果不能够满足用户的需要，设计的再好，采用的技术再先进，也没有任何的意义。所以这一点非常的朴实，但却是软件质量的第一个评判标准。

可靠性是质量的一个方面。作为测试人员，主要工作在于通过减少程序中的缺陷数量来提高客户满意度。但是如果一个项目在最后阶段修改程序使其具备了某个特别有用的特性，即使改动后的程序不太可靠，这样做也可能是在改进程序的质量。特性和缺陷都在决定着质量。

1.2.2 软件错误定义

前面我们介绍了软件质量的定义，而开发高质量的软件，并不是一件容易的事情。在现实中，人们已经遇到了太多的软件质量问题，这些问题，轻则给使用者带来不便，重则导致重要数据丢失、重大财产损失，甚至危及生命。这些质量问题被称为软件错误。

软件错误是指软件产品中存在的导致期望的运行结果和实际结果间出现差异的一系列问题，这些问题包括故障、失效、缺陷。软件错误又被称为“Bug”。Bug 的出现并不一定是代码问题，也可能是需求或设计等方面引起的。我们也可以认为软件错误是用户不喜欢的或者不能帮助用户使用应用程序达到目标的东西。这里有两种对于软件错误的定义：

- 当程序没有实现其最终用户合理预期的功能要求时，就表现为软件错误。
- 从来就没有对缺陷的绝对定义，也没有对其存在的绝对定义。程序存在缺陷的程度是由程序无法实现有用功能的程度来测量的。这是基本的人为测量。

1.2.3 软件质量保证

为了提高软件质量，人们进行了大量的研究和实践。最初的重点是着眼于技术革新，从各种软件工具（如编辑、编译、调试工具等等）到各种计算机辅助软件工程（CASE）环境；同时，注重对软件开发“模型”的研究。但是未达到预期的目标，人们逐渐认识到，如果能够同时对软件开发过程的质量加以控制，那么软件质量大幅度的提高才能成为可能。也就是

说，只有从一开始就在开发过程中实施严格的过程控制，软件产品的质量才可能有保证。因此，软件质量保证也从最初的技术、方法为重心，转移到以过程管理为重心。

软件质量保证的活动主要包括：

- 技术方法的应用。
- 正式技术评审的实施。
- 软件测试。
- 标准的执行。
- 修改的控制。
- 度量。
- 记录和记录保存。

软件质量保证（Software Quality Assurance——SQA）是为了确保软件开发过程和结果符合预期的要求，而建立的一系列规程，以及依照规程和计划采取的一系列活动及其结果评价。

软件质量保证并不等同于软件测试。软件质量保证评估过程质量，主要的目的是缺陷预防，而软件测试评估产品质量，主要目的是错误检测。软件质量保证通过评审测试结果和搜集软件质量度量监控测试的有效性，对软件测试文档的审核确定测试活动是否符合建立的标准和规范的要求。

1.3 测试一个小软件

在开始讲解软件测试的概念、方法之前，我们先一起来看一个小的软件（确切地说，应该是一个软件模块），并试图对这个软件进行测试工作。

1.3.1 软件功能

当刚开始踏入软件开发/测试行业时，往往会从一些简单的小程序或者软件中的功能模块开始。也许在刚开始工作的时候，第一项任务就是开发或者测试这样一个软件模块。千万不要小看这个登录界面。在 Microsoft Windows 2000 中，就曾经在登录界面中出现过安全问题。

（1）软件界面　如图 1-5 所示。

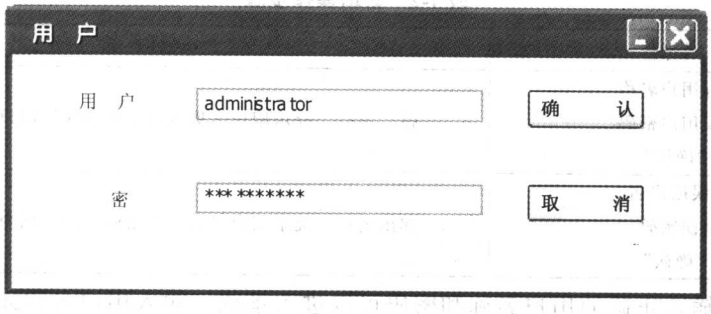

图 1-5　登录软件界面

(2) 功能描述　在登录界面中，包括两个文本输入框（用户名称和密码），然后是一个“确认”按钮和一个“取消”按钮。用户在“用户名称”文本输入框中输入用户名称，然后在“密码”文本输入框中输入密码（为了保密，密码以星号形式来显示）。用户按下“确认”按钮时，如果输入的用户名称和密码正确（所谓正确，是指存在这个名称的用户，并且该用户的密码与所输入的一致），则显示登录成功，然后用户可以进入系统；如果输入错误，则提示“用户名称或者密码错误”，然后用户可以重新输入。如果用户按下“取消”按钮，则清除在“用户名称”和“密码”中输入的内容，用户可以重新输入。另外，读者需要注意到，无论是用户名称输入错误还是密码输入错误，系统的提示都是一样的。这是基于这样的考虑：如果某个未经授权的用户，期望猜测用户名称和密码，当他输入错误后，如果分别提示用户名称错误和密码错误，那么这个攻击者就可以确定某个用户在系统中是存在的，然后再尝试找到这个用户的密码。而系统仅仅提示“用户名称或者密码错误”，那么对于这个攻击者来说，不能确定到底是输入了错误的密码，还是所输入的用户名称根本不存在，这样就增加了对系统攻击的难度。

假定程序员已经完成了代码编写工作，现在我们作为测试人员来查找里面可能存在的软件错误。在开始查找错误前，我们要牢记一点：无论是设计、编码还是测试工作，软件需求都是这些工作的最原始依据。

1.3.2　寻找错误

在开始进行测试之前，我们建立一个表格（见表 1-1），记录测试人员的操作步骤和软件的反应，然后我们看系统的实际反应和预期的反应是否一致。如果不一致，则说明在软件中存在错误。

表 1-1　测试人员的操作步骤和软件的反应

测试人员操作	软件反应
第一步：输入正确用户名称	弹出窗口，显示登录成功，用户进入系统
第二步：输入该用户密码	
第三步：单击“确认”按钮	

结果显示能够进行系统登录，测试结束了吗？上面的测试，证明了软件在输入正确的用户名称和密码的情况下，能正常工作。

系统能够登录并不表明登录不会存在问题，修改我们的测试，增加操作（见表 1-2）。

表 1-2　新增操作步骤

测试人员操作	软件反应
输入已有用户名称 输入错误用户密码 单击“确认”	弹出窗口，提示用户名称或者密码错误，用户无法进入系统
输入错误用户名 输入错误密码 单击“确认”	弹出窗口，提示用户名称或者密码错误，用户无法进入系统

以上测试了输入正确的用户名称和密码可以进入系统；输入用户名称或密码错误无法进入系统。测试不是盲目的，在一阵显而易见的测试之后，对还需要测试什么做一些思考。

这个时候，我们会意识到：测试是需要设计的！所谓“设计”，指的是测试什么内容、怎么进行测试。

测试的设计可以从系统的需求和设计着手。通过阅读系统需求和设计文档发现，对系统登录有如下描述：

- 用户名称长度为6～10位（含6位和10位）。
- 用户名称由字符（a～z、A～Z）和数字（0～9）组成。
- 不能为空、空格和特殊字符。
- 密码规则同用户名称规则。

根据系统要求，重新设计测试（见表1-3）。

表1-3 重新设计测试

测试人员操作	预期结果	软件反应
输入正确的用户名称和密码（均为6位） 单击“确认”按钮	进入系统	
输入正确的用户名称和密码（均为10位） 单击“确认”按钮	进入系统	
输入正确的用户名称和密码（均为6～8位之间） …	进入系统	
用户名称为空，……	提示用户名称或者密码错误，不能进入系统	
用户名称为空格，……	提示用户名称或者密码错误，不能进入系统	
用户名称小于6位，……	提示用户名称或者密码错误，不能进入系统	
……		……

另外，大家是否注意到在上面的表格中，与我们前面所用的表格相比，加了“预期结果”这样一列？这份表格是我们在开始进行操作前完成的，在那个时候，我们要明确，在某个操作之后，软件应该有什么样的反应，然后在操作的过程中，记录软件的反应，与我们预期的结果对比，如果不一致，则说明软件中存在错误。

1.4 理解软件测试

有一个古老的寓言：在古代的中国有个祖传名医家庭，其中有一位被认为是位了不起的内科医生，他被当地的君主所雇用。有人问这位内科医生他的家庭中是谁的医术最高。他回答说：“我以很极端的疗法去医治那些垂死的病人，有时的确救活了个别病人，于是我的名声就在君主中间传开了。我二哥在疾病刚刚开始出现时就治好病人，他的技艺和名声在本地农民和邻居中间传开了。而我的大哥能觉察疾病的预兆，在疾病发出来以前就根除它。他的名字在我们家外面的人是不知道的。”软件产品的测试工作就像这位医生的大哥做的事情。测试人员通过测试在软件产品中的“疾病”（缺陷）发病之前就找到它，并把“疾病”排除，使得软件到了用户手里的时候，用户看不到明显的“疾病”（缺陷）。

1.4.1 基本概念

即使有丰富经验的程序员，也难免在编码中发生错误，何况有些错误在设计甚至需求分析阶段早已埋下祸根，无论是早期潜伏下来的错误或编码中新引入的错误，若不及时排除，轻者降低软件的可靠性，重者导致整个系统的失败。

软件测试就是为了发现程序中的错误而分析或执行程序的过程。具体地说，软件测试是分析程序或根据软件开发各阶段的规格说明和程序的内部结构而精心设计出一批测试用例，并利用测试用例来运行程序，以发现程序错误的过程。

软件测试是软件质量保证的重要手段，据研究机构统计分析表明，国外软件开发机构40%的工作量花在软件测试上，软件测试费用占软件开发总费用的30%～50%。对于一些要求高可靠、高安全的软件，测试费用所占比例更高。由此可见，要成功开发出高质量的软件产品，必须重视并加强软件测试工作。

软件测试就是为了发现程序中的错误而分析或执行程序的过程。

根据软件测试的定义，测试包含了“分析”或“运行”软件。分析软件产品的过程称为静态测试（Static Testing）。静态测试不实际运行软件，包括走查、代码审查、代码评审、桌面检查。相反，在目标环境中实际运行软件的测试过程称为动态测试（Dynamic Testing）。静态测试和动态测试互为补充，各自用不同的方式检测软件错误。

软件测试有两个基本的功能：验证（Verification）和确认（Validation）。验证指保证软件正确地实现了特定功能的一系列活动。确认指保证最终的产品满足系统需求。验证更多地关注开发过程中的各项活动，比如，系统测试的目的之一是保证系统设计与需求的一致性。通俗地说，验证保证产品正确性，确认保证生产了正确的产品。

测试和改正活动可以在软件生命周期的任何阶段进行。然而，随着开发不断推进，找出并改正错误的代价会急剧地增加。在需求文档的初次评审期间对其进行修改，付出的代价会很小。如果代码已经编写完毕，再要进行需求变更，代价会大得多，因为必须重新编写代码。程序员自己发现错误并进行缺陷改正，代价会很小。此时没有交流成本，因为无须把错误向其他任何人解释，也无须将其登记到缺陷跟踪数据库中。测试人员和经理无须审核缺陷的状态。错误也不会阻碍或破坏其他任何人的工作。在发布程序前改正错误要比分发新磁盘甚至派遣技术人员到每个用户现场进行后期改正代价小得多。

与其他的技术学科一样，软件测试也是一个需要专门技术的领域。为了有效地实施测试，软件测试人员应该至少具备以下两个关键领域方面的知识：

- 软件测试技术。
- 被测应用程序及其相关应用领域。

对于每个新分配的测试任务，测试人员必须花费时间了解应用程序。没有经验的测试人员还必须学习测试技术，包括一般的测试概念以及定义测试用例。

关于软件测试，读者还需要了解这样一些描述：

- 测试能提高软件的质量，但是提高质量不能依赖测试。
- 测试只能证明错误存在，不能证明错误不存在。“彻底地测试”难以成为现实，要考虑时间、费用等限制，不允许无休止地测试。
- 测试的主要困难是不知道如何进行有效地测试，也不知道什么时候可以放心地结束