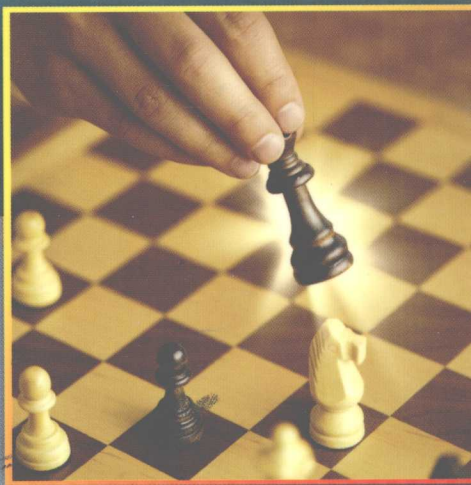


普通高等学校计算机科学与技术专业规划教材

软件测试方法与应用

METHOD AND PRACTICE OF SOFTWARE TESTING

孙海英 等编著



中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

普通高等学校计算机科学与技术专业规划教材

软件测试方法与应用

孙海英 等编著

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

内 容 简 介

软件测试是以发现隐藏在软件中的问题为主要目的的软件研发活动。本书系统地阐述了与软件测试相关的各项工作的基本内容,重点讲解了主流的测试用例设计方法和软件测试过程,并以一个缺陷管理工具的完整测试过程为案例说明了上述内容在实际项目中的具体运用。

本书适合作为软件学院本科学生软件测试课程的基础教材,也可作为计算机学院本科生相关专业课程的教材,还可作为软件测试工程师以及对软件测试领域感兴趣的IT人员学习和参考的资料。

图书在版编目(CIP)数据

软件测试方法与应用/孙海英等编著. —北京:中国铁道出版社, 2009. 2

普通高等学校计算机科学与技术专业规划教材

ISBN 978-7-113-09709-7

I. 软… II. 孙… III. 软件—测试—高等学校—教材
IV. TP311.5

中国版本图书馆CIP数据核字(2009)第022813号

书 名: 软件测试方法与应用
作 者: 孙海英 等编著

策划编辑: 秦绪好 杨 勇

责任编辑: 王占清

编辑助理: 张 丹

封面设计: 付 巍

编辑部电话: (010) 63583215

封面制作: 白 雪

责任印制: 李 佳

出版发行: 中国铁道出版社(北京市宣武区右安门西街8号 邮政编码: 100054)

印 刷: 三河兴达印务有限公司

版 次: 2009年5月第1版 2009年5月第1次印刷

开 本: 787mm×1092mm 1/16 印张: 16.75 字数: 380千

印 数: 5 000册

书 号: ISBN 978-7-113-09709-7/TP·3158

定 价: 28.00元

版权所有 侵权必究

本书封面贴有中国铁道出版社激光防伪标签,无标签者不得销售

凡购买铁道版的图书,如有缺页、倒页、脱页者,请与本社计算机图书批销部调换。

普通高等学校计算机科学与技术专业规划教材

主 任：蒋宗礼（北京工业大学）

副主任：王志英（国防科技大学）

杨 波（济南大学）

委 员：（按姓氏音序排列）

常会友（中山大学）

陈俊杰（太原理工大学）

陈 明（中国石油大学）

陈笑蓉（贵州大学）

陈志国（河南大学）

顾乃杰（中国科技大学）

胡 亮（吉林大学）

黄国兴（华东师范大学）

姜守旭（哈尔滨工业大学）

李仲麟（华南理工大学）

刘腾红（中南财经政法大学）

罗军舟（东南大学）

王国仁（东北大学）

王命延（南昌大学）

吴 跃（电子科技大学）

袁晓洁（南开大学）

岳丽华（中国科技大学）

张 莉（北京航空航天大学）

本书责任编辑：黄国兴（华东师范大学）

计算机学科虽然是一个年轻的学科,但它已经成为一门基础技术学科,在各个学科发展中扮演重要角色,使得社会产生了对计算机科学与技术专业人才的巨大需求,促使计算机科学与技术专业成为我国理工专业中规模最大的专业,在高等教育发展中做出了巨大贡献。近些年来,随着国家信息化建设的推进,作为核心技术的计算机技术更是占有重要的地位。信息化建设不仅需要更先进、更便于使用的先进计算技术,同时也需要大批的建设人才。根据社会需求准确定位,培养计算机人才,是计算机科学与技术专业及其相关专业的历史使命,也是实现专业教育从劳动就业供给导向型向劳动就业需求导向型转变的关键,从而也就成为提高高等教育质量的关键。

教材在人才培养中占有重要地位,承担着“重要的责任”,确定了其高质量的基本要求。社会对计算机专业人才需求的多样性和特色,决定了教材建设的针对性,从而也造就了百花齐放、百家争鸣的局面。

关于建设高质量的教材,教育部在提高本科教育质量的文件中都提出了明确要求。教高〔2005〕1号(2005年1月7日)文件指出,“加强教材建设,确保高质量教材进课堂。要大力锤炼精品教材,并把精品教材作为教材选用的主要目标。”“要健全、完善教材评审、评介和选用机制,严把教材质量关。”为了更好地落实教育部的这些要求,我们按照教育部高等学校计算机科学与技术教学指导委员会颁布的《高等学校计算机科学与技术专业发展战略研究报告暨专业规范(试行)》所构建的计算机科学与技术专业本科教育的要求,组织了这套教材。

作为优秀教材的基础,我们首先坚持高标准,以对教育负责的精神去鼓励、发现、动员、选拔优秀作者,并且有意识地培育优秀作者。优秀作者保证了“理论准确到位,既有然,更有所以然;实践要求到位、指导到位”等要求的实现。

其次是按照人才培养的需要强调适当学科形态内容。粗略地讲,计算机科学的根本问题是“什么能被有效地自动计算”,科学型人才强调学科抽象和理论形态的内容;计算机系统工程的根本问题应该是“如何低成本、高效地实现自动计算”,工程型人才强调学科抽象和设计形态的内容;计算机应用的根本问题是“如何方便、有效地利用计算机系统进行计算”,应用型人才的培养偏重于技术层面的内容,强调学科设计形态的内容,在进一步开发基本计算机系统应用的层面上体现学科技术为主的特征。教材针对不同类型人才的培养,在满足基本知识要求的前提下,强调不同形态的内容。

第三是重视知识的载体作用,促进能力培养。在教材内容的组织上,体现大学教育的学科性和专业性特征,参考《高等学校计算机科学与技术专业发展战略研究报告暨专业规范(试行)》示范性课程大纲,覆盖其要求的基本知识单元。叙述上力争引导读者进行深入分析,努力使读

者在知其然的基础上,探究其所以然。通过加强对练习和实践的引导,进一步培养学生的能力,促使相应课程在专业教育总目标的实现中发挥作用。

第四是瞄准教学需要,提供更多支持。近些年来,随着计算机技术、网络技术等在教学上的应用,教学手段、教学方式不断丰富,教材的立体化建设对丰富教学资源发挥了重要作用。通常,除主教材外,还要配套教学参考书、实验指导书、电子讲稿、网站等。

第五是面向主要读者,强调教材的写作特征,努力做到叙述清晰易懂,语言流畅,深入浅出,有吸引力而不晦涩;追求描述的准确性,强调用词和描述的一致性、语言表达的清晰性和叙述的完整性;分散难点,循序渐进,防止多难点、多新概念的局部堆积。

我们相信,这套教材一定能够在培养社会需要的计算机专业人才上发挥重要作用,希望大家广为利用,并在使用中不断提高。

普通高等学校计算机科学与技术专业规划教材编审委员会

2008年1月

软件已经成为当今社会正常运转不可或缺的要素之一。伴随着软件为人类社会运转和发展带来的巨大惊喜,人类也因为软件中的问题付出了沉重的代价。软件以人类脑力劳动为主的生产方式决定了软件中将会存在错误。软件测试是一种以找出隐藏在软件中的错误为主要目的的研发活动。作为保证软件质量的关键性环节,其重要性在软件大规模生产和应用的今天也日益凸显出来。

本书用平实的语言清晰地梳理了软件测试的基础知识,包括相关概念、方法、技术和流程,并结合具体项目案例着重突出这些内容在软件工程实际中的综合运用。本书的特点在于理论联系实际,在讲解方法的时候配有项目案例,这种结构在目前的软件测试教材中是不多见的。

全书共 14 章,从内容上分为两大部分:第一部分是软件测试方法篇,包括第 1 章~第 9 章,重点讲述了与软件测试相关的概念、原理、方法和技术;第二部分是软件测试实践篇,包括第 10 章~第 14 章,这部分内容结合一个缺陷管理工具原型 Easy Bug Manager 的测试案例说明软件测试方法部分中所陈述的内容在实际项目测试过程中的运用。各章具体内容描述如下:

第 1 章 软件测试的任务,主要介绍了与软件测试基本概念相关的各种问题,包括缺陷的概念、软件测试的目的、分类流程、输出物等。

第 2 章 静态测试,主要介绍了两种主流的静态测试方法:评审和静态分析。

第 3 章 黑盒测试用例设计方法,主要介绍了等价类、边界值、决策表、错误猜测、场景法、正交实验法等经典的黑盒测试用例设计方法并结合具体示例说明它们的综合运用策略。

第 4 章 白盒测试用例设计方法,主要介绍了路径测试、分支-条件测试、循环测试等经典的白盒测试用例设计方法和基于控制流的覆盖准则。

第 5 章 软件测试过程,主要介绍了软件测试过程框架和各种在软件测试过程中需要开发的测试类文档及其关系。

第 6 章 单元测试,主要讨论了单元测试的观点和在单元测试过程包含的各种活动:单元测试计划、单元测试用例设计、单元测试执行、单元测试报告等。

第 7 章 集成测试,主要讨论了各种集成策略和在集成测试过程包含的各种活动:集成测试计划、集成测试用例设计、集成测试执行、集成测试报告等。

第 8 章 系统测试,主要讨论了系统测试的观点和在系统测试过程中包含的各种活动:系统测试计划、系统测试用例设计、系统测试执行、系统测试报告等。

第 9 章 非功能性测试,介绍了几种常用的非功能性测试方法,包括性能测试、兼容性测试、用户界面测试、安装测试、安全性测试和文档测试等。

第 10 章 缺陷管理和缺陷管理工具 EBM,在本章中,结合缺陷管理的一般内容;重点讲述一款为软件测试教学而开发的缺陷管理工具原型 Easy Bug Manager 的需求规格说明、架构设计、数据库设计、界面设计等内容,为后续测试案例的讲解奠定基础。

第 11 章 EBMV1.0 测试策略,给出了测试案例 EBMV1.0 的测试策略。

第 12 章 EBMV1.0 单元测试过程,给出了测试案例 EBMV1.0 的单元测试计划、测试规格说明、测试用例、测试报告等。

第 13 章 EBMV1.0 集成测试过程,给出了测试案例 EBMV1.0 的集成测试计划、测试规格说明、测试用例、测试报告等。

第 14 章 EBMV1.0 系统测试过程,给出了测试案例 EBMV1.0 的系统测试计划、测试规格说明、测试用例、测试报告等。

本书适合作为软件学院本科生软件测试课程的基础教材,也适合作为计算机学院本科生相关课程的教材,还可作为软件测试工程师和其他对软件测试领域感兴趣的 IT 人员学习和参考的资料。书中所涉及的参考资料,包括演示文稿、测试案例代码、文档等,均可到 <http://edu.tqbooks.net> 下载。

全书由孙海英设计并担任主要编写工作,其他的参编人员还包括黄国兴、赵慧、姜宁康、王政、杨林、王林和陈涛。书中的部分内容参阅了一些专家的著作、论文和教材,在此向他们表示深深的敬意和谢意。

在本书的编写过程中,得到了众多同事和朋友的支持和鼓励。在此,特别感谢华东师范大学软件学院的黄国兴教授,正是在黄教授热情的鼓励和悉心的指导下,本书才得以完成。最后,感谢我的学生、朋友、同事和家人,正是他们的殷殷关爱支持我完成了此书。

由于本人才疏学浅,加之精力有限,书中难免存在疏漏和不妥之处,恳请各位读者与同行专家提出宝贵的意见和建议。

编 者

2009 年 1 月

第 1 章 软件测试的任务	1
1.1 软件测试的任务	2
1.2 软件测试相关概念	4
1.2.1 缺陷	4
1.2.2 软件测试	5
1.2.3 软件测试方法的分类	6
1.2.4 测试用例	8
1.2.5 缺陷报告单	10
1.3 软件测试的流程	10
本章小结	13
习题一	15
第 2 章 静态测试	16
2.1 同行评审	17
2.1.1 审查	17
2.1.2 小组评审	20
2.1.3 走查	20
2.1.4 同级桌查	20
2.1.5 临时评审	20
2.2 软件评审指导书	21
2.2.1 目的	21
2.2.2 范围	21
2.2.3 评审角色及职责	21
2.2.4 过程准则	23
2.2.5 相关资料	26
2.2.6 过程监控	26
2.3 评审软件需求规格说明书	26
2.4 评审源代码	27
2.5 代码静态分析	35
2.6 代码静态分析工具	36
本章小结	37
习题二	37
第 3 章 黑盒测试用例设计方法	38
3.1 等价类	39

3.1.1 划分等价类	39
3.1.2 识别测试用例	40
3.2 边界值	43
3.3 决策表	47
3.4 错误猜测法	51
3.5 场景测试	51
3.6 正交试验法	56
3.7 综合运用黑盒测试用例设计方法进行功能测试	60
3.8 功能测试工具	62
本章小结	63
习题三	64
第4章 白盒测试用例设计方法	65
4.1 基本路径测试	66
4.2 分支-条件测试	71
4.3 循环测试	75
4.4 其他白盒测试方法	77
4.4.1 数据流测试	77
4.4.2 程序插桩	78
4.4.3 符号测试	78
4.5 综合运用白盒测试方法进行程序结构测试	78
4.5.1 运用基本路径测试方法设计相关测试用例	79
4.5.2 运用分支-条件测试方法设计相关测试用例	80
4.5.3 运用循环测试方法设计相关测试用例	80
4.5.4 结果合并	81
4.5.5 增加相关测试用例	81
4.6 白盒测试工具	82
4.7 覆盖准则	84
4.7.1 语句覆盖	84
4.7.2 分支覆盖	85
4.7.3 条件覆盖	85
4.7.4 分支-条件覆盖	86
4.7.5 多条件覆盖	86
4.7.6 路径覆盖	87
4.8 覆盖率测试工具	87
本章小结	88
习题四	89
第5章 软件测试过程	90
5.1 全过程的测试	91

5.2 软件测试过程	92
5.3 软件测试类文档	94
5.3.1 测试策略	96
5.3.2 测试计划	96
5.3.3 测试设计规格说明书	97
5.3.4 测试用例规格说明书	97
5.3.5 测试用例执行过程规格说明书	97
5.3.6 测试日志	98
5.3.7 意外事件报告	98
5.3.8 测试报告	98
本章小结	99
习题五	100
第 6 章 单元测试	101
6.1 单元测试过程	102
6.2 单元测试计划	103
6.3 单元测试用例设计	106
6.4 单元测试执行	107
6.5 单元测试报告	108
本章小结	108
习题六	109
第 7 章 集成测试	110
7.1 集成策略	111
7.1.1 基于功能分解的集成	111
7.1.2 基于 MM 路径的集成	113
7.2 集成测试过程	119
7.3 集成测试执行和报告	120
本章小结	120
习题七	121
第 8 章 系统测试	122
8.1 系统测试过程	123
8.2 系统测试计划	124
8.3 系统测试用例设计	126
8.4 系统测试执行	126
8.5 系统测试小结	127
8.6 回归测试	127
8.7 验收测试	128
本章小结	128
习题八	129

第 9 章 非功能测试	130
9.1 性能测试	130
9.1.1 性能测试规划	133
9.1.2 性能测试设计与开发	135
9.1.3 性能测试执行与监控	137
9.1.4 测试结果分析和系统调整	137
9.1.5 性能测试工具	139
9.2 兼容性测试	139
9.3 用户界面测试	141
9.3.1 界面整体测试	141
9.3.2 界面元素测试	142
9.4 安装测试	145
9.5 安全性测试	145
9.6 文档测试	147
9.6.1 测试用户手册	148
9.6.2 测试在线帮助	149
本章小结	149
习题九	150
第 10 章 缺陷管理和缺陷管理工具 EBM	151
10.1 软件缺陷管理	151
10.1.1 缺陷生命周期	152
10.1.2 缺陷处理流程	153
10.1.3 缺陷管理工具	153
10.1.4 缺陷报告单	154
10.2 缺陷管理系统 Easy Bug Manager	156
10.3 EBM 设计说明	158
10.3.1 EBMV1.0 系统架构设计	158
10.3.2 数据库设计	160
10.3.3 通信协议设计	163
10.3.4 界面设计	164
第 11 章 EBMV1.0 测试策略	169
11.1 简介	169
11.1.1 目的	169
11.1.2 范围	169
11.2 测试综述	170
11.2.1 测试过程定义	170
11.2.2 测试活动	171

11.3 单元测试策略	171
11.3.1 入口准则	171
11.3.2 出口准则	171
11.3.3 测试质量要求	172
11.3.4 需要的软件工具	172
11.3.5 人力资源和硬件资源	172
11.3.6 被测对象依赖关系	172
11.4 集成测试策略	172
11.4.1 入口准则	173
11.4.2 出口准则	173
11.4.3 测试质量要求	173
11.4.4 需要的软件工具	173
11.4.5 人力资源和硬件环境	173
11.5 系统测试策略	173
11.5.1 入口准则	174
11.5.2 出口准则	174
11.5.3 测试质量要求	174
11.5.4 人力资源和硬件资源	175
11.6 相关约定	175
11.6.1 缺陷严重级别说明	175
11.6.2 缺陷优先级说明	175
第 12 章 EBMV1.0 单元测试过程	176
12.1 EBMV1.0 单元测试计划	176
12.1.1 概述	176
12.1.2 测试交付件	178
12.1.3 测试方法	178
12.1.4 被测单元	179
12.1.5 不被测单元	180
12.1.6 过程准则	181
12.1.7 测试进度	181
12.1.8 测试资源	182
12.1.9 风险列表	182
12.2 EBMV1.0 单元测试设计说明书	183
12.2.1 概述	183
12.2.2 待测单元	183
12.2.3 BugProcess.java 测试设计说明	184
12.2.4 EBM.Server.DB.BugManage.java 测试设计说明	185
12.2.5 EBM.Server.DB.BugDAO.java 测试设计说明	186
12.3 EBMV1.0 单元测试用例	187

12.3.1	BugProcess.java 测试用例	187
12.3.2	BugManage.java 测试用例	190
12.3.3	BugDAO.java 测试用例	199
12.4	EBMV1.0 单元测试执行	203
12.5	EBMV1.0 单元测试报告	203
12.5.1	概述	203
12.5.2	测试结果统计	204
12.5.3	遗留问题统计	204
12.5.4	被测对象评价	205
12.5.5	测试结果	205
第 13 章	EBMV1.0 集成测试过程	206
13.1	EBMV1.0 集成测试计划	206
13.1.1	概述	206
13.1.2	测试交付件	208
13.1.3	测试方法	208
13.1.4	被测接口	209
13.1.5	过程准则	210
13.1.6	测试进度	210
13.1.7	测试资源	211
13.1.8	风险列表	212
13.2	EBMV1.0 集成测试设计说明书	212
13.2.1	概述	212
13.2.2	待测特征	213
13.2.3	测试数据设计	213
13.2.4	“newbugsubmit” 消息路径测试设计说明	215
13.3	EBMV1.0 集成测试用例	216
13.4	EBMV1.0 集成测试执行	218
13.5	EBMV1.0 集成测试报告	218
13.5.1	概述	218
13.5.2	测试结果统计	219
13.5.3	遗留问题统计	219
13.5.4	被测对象评价	219
13.5.5	测试结果	220
第 14 章	EBMV1.0 系统测试过程	221
14.1	EBMV1.0 系统测试计划	221
14.1.1	概述	221
14.1.2	测试交付件	222
14.1.3	测试方法	222
14.1.4	被测特性	223

14.1.5	过程准则	224
14.1.6	测试进度	224
14.1.7	测试资源	226
14.1.8	风险列表	226
14.2	EBMV1.0 系统测试设计说明书	227
14.2.1	概述	227
14.2.2	待测特征	227
14.2.3	测试方法总体说明	228
14.2.4	缺陷处理测试设计说明	228
14.3	EBMV1.0 系统测试用例	231
14.3.1	EBMV1.0 系统测试数据	231
14.3.2	缺陷处理测试用例	232
14.4	EBMV1.0 系统预测试	245
14.5	EBMV1.0 系统测试执行	246
14.6	EBMV1.0 系统测试报告	247
14.6.1	概述	247
14.6.2	测试结果统计	248
14.6.3	被测对象评价	248
14.6.4	测试结果	248
参考文献		249

第 1 章

软件测试的任务

软件已经成为当今社会正常运转不可或缺的元素之一，人类社会需要更多高质量软件，但是由于客观存在的大大小小的软件问题，软件的质量令人担忧。

软件测试，简单地说，是一种发现隐藏在软件中问题的研发活动，其主要任务就是要尽可能多地将隐藏在软件中的问题找出来以保证软件的质量达到规定的目标。

本章主要讨论与软件测试相关的各种基本概念，说明软件测试的流程。

本章要点

- 软件缺陷
- 软件测试
- 软件测试的目的和分类
- 测试用例
- 缺陷报告单
- 软件测试流程

伴随着计算机技术为人类发展带来的惊喜，人类也因为计算机中尤其是软件中存在的问题付出了巨大甚至惨重的代价。例如，上世纪 60 年代中期，由于飞行控制软件中缺少了一个逗号导致美国首次金星探测飞行的失败；1981 年 10 月，首次发射的航天飞机因为在发射前发现了一个软件错误而使发射被迫推迟；1996 年 6 月，由于惯性制导系统软件中的错误导致耗资 80 亿美元的欧洲航天局阿丽亚娜 501 火箭发射升空 37s 后爆炸；海湾战争期间，爱国者导弹因软件问题误伤了美国士兵；2005 年，某软件公司因提供的更新病毒代码出错，造成日本电车售票系统瘫痪……。据美国 NIST 在 2002 年发布的一项研究估计，美国经济每年因软件中的 Bug 会损失 600 亿美金，在约合 0.6% 的国民生产总值。面对一个个惨痛的教训、一个个触目惊心的数字，人们不断地反思为什么软件中会存在问题？又该如何尽量避免软件出现问题？如何提高软件的质量？

软件中存在问题是由目前的软件生产模式决定的。在可以预期的未来，软件的开发模式仍然将是以人类为主的脑力密集型活动，这种开发模式注定人是软件开发中的主角。在软件开发各个环节中产生的人为错误，使得软件中客观存在着大大小小的问题。这些问题在一定的条件下被激活，便会使软件的运行出现差错，导致或大或小的经济损失，甚至会威胁人类生命导致令人痛心疾首的人间悲剧。虽然，质量专家认为良好的过程可以消除因

为人类粗心而造成的产品质量损失。但是，软件不同于生产线上的电子产品、食品或者其他产品，它不是原料的机械加工生产，软件是个非常特殊的产物，是智力密集型产品，单单通过良好的过程不足以避免软件问题的产生。

为了解决软件的质量问题，在过去的几十年里科学家们进行了不懈的努力和研究。其中，以 C.A.Hoare 和 E.W.Dijkstra 为代表的科学家提出了以形式化方法提高软件质量的途径，为人类描绘了一幅可以大幅度提高软件质量，减少甚至完全消除软件测试与软件维护的美好远景。但是，由于客观存在的复杂原因，使得在过去的几十多年中软件开发并没有按照形式化的途径方向发展；相反，以软件测试为中心的软件质量保证技术却在生产实践中得以广泛地应用，成为当前保证软件质量的主力。不过，尽管软件测试从业者和研究人员建立了很多软件测试理论和方法，开发了不少软件测试工具，但是非常遗憾的是，目前的软件测试方法并不能彻底地找到软件中存在的所有问题，不能给出软件是正确的证据，而只能说明软件中存在的问题。也正因为如此，软件测试的机遇与挑战同在。

1.1 软件测试的任务

软件测试的主要任务，简单地概括，就是尽可能多地将隐藏在软件中的问题找出，并保证软件的质量达到既定的要求。那么，在如今的生产模式下，软件从业者又是如何发现隐藏在软件中的问题？可以用经典的软件测试教学案例三角形问题为例说明这个问题。三角形问题的功能描述如下：

[三角形问题] 输入三个整数，这三个数分别代表三角形三条边的长度，判断这三个数构成的三角形是等边三角形、等腰三角形还是不等边三角形并输出相应的结果。

图 1-1 所示的是用 Java 语言实现三角形问题的运行界面。现在，评价一下这个程序是否满足上述三角形问题的功能说明，所提供的资料中不包括源代码，仅有这份功能说明和可执行文件 isTriangle.class。在这种情况下，应该如何完成这个任务？

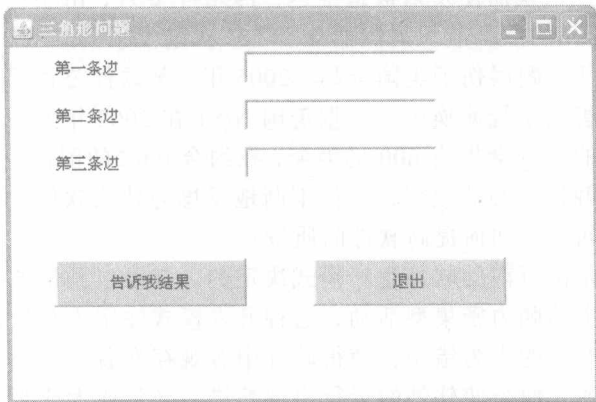


图 1-1 三角形问题

很自然地，首先想到采用下面这种类似于试验的方法：运行程序，输入一些数据，执行三角形判断的命令，查看程序的输出结果是否和预期的一致。如果一致，则认为被测试的功能没有问题；否则，被测试的功能可能存在问题并且将问题记录下来以备修改。不错，选择输入数据、运行程序、输入测试数据、执行操作、查看输出结果是否和预期结果一致，