第3章　黑盒测试

黑盒测试是软件测试的核心测试方法之一，是学习本书的重点内容。在黑盒测试期间，把被测程序视为一个黑盒子，测试人员并不清楚被测程序的源代码或者该程序的具体结构，不需要对软件的结构有深层的了解，而是只知道该程序输入和输出之间的关系，依靠能够反映这一关系的功能规格说明书，来确定测试用例和推断测试结果的正确性。

本章介绍黑盒测试的基本概念与基本方法，常用的黑盒测试方法有等价类划分、边界值分析、决策表法、因果图、正交实验法、故障猜测法、状态图法、随机数据法等。每种黑盒测试方法各有所长，应针对软件开发项目的具体特点，选择适当的测试方法，设计高效的测试用例，有效地将软件中隐藏的故障揭露出来。一个好的测试策略和测试方法必将给整个测试工作带来事半功倍的效果。本章的实践性较强，希望能举一反三，将这些测试技术和软件开发结合起来学习。

3.1　黑盒测试概述

黑盒测试(Black Box Testing)也称功能测试，它是通过测试来检测每个功能是否都能正常使用。在黑盒测试中，在完全不考虑程序内部结构和内部特性的情况下，在程序接口进行测试，它只检查程序功能是否按照需求规格说明书的规定正常使用，程序是否能适当地接收输入数据而产生正确的输出信息。黑盒测试着眼于程序外部结构，不考虑内部逻辑结构，主要针对软件界面和软件功能进行测试，黑盒测试示意图如图 3-1 所示。黑盒测试是一种基于用户观点出发的测试。

图 3-1　黑盒测试示意图

软件黑盒测试是以用户的角度，从输入数据与输出数据的对应关系出发进行测试的。很明显，如果外部特性本身有问题或规格说明的规定有误，用黑盒测试方法是发现不了的。

软件黑盒测试法注重于测试软件的功能需求，主要试图发现下列几类错误：功能错误或遗漏、界面错误、数据结构或外部数据库访问错误、性能错误、初始化和终止错误等。

黑盒测试从理论上讲只有采用穷举输入测试，把所有可能的输入都作为测试情况考虑，才能查出程序中所有的错误。实际上，测试情况有无穷多个，人们不仅要测试所有合法的输入，而且还要对那些不合法但可能的输入进行测试。这样看来，完全测试是不可能的，所以要进行有针对性的测试，通过制定测试案例指导测试的实施，保证软件测试有组织、按步骤，以及有计划地进行。软件黑盒测试行为必须能够加以量化，才能真正保证软件质量，而测试用例就是将测试行为具体量化的方法之一。

例如，对 Windows 中文件名的测试。Windows 文件名可以包括除了“、”“/”“:”“.”

“?”“＜ ”“＞”和“\”之外的任意字符。文件名长度是 1～255 个字符。如果为文件名创建测试用例，等价类分为合法字符、非法字符、合法长度的名称、超过长度的名称等。使用穷举设计输入测试用例，其工作量是人们无法承受的。Windows 附件中“写字板”软件的“保存为”对话框如图 3-2 所示。

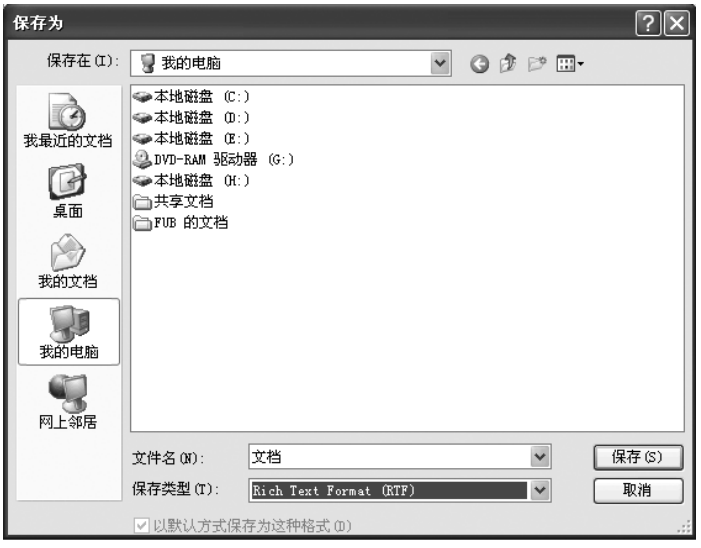

图 3-2 “保存为”对话框

黑盒测试用例设计方法包括等价类划分法、边界值分析法、错误推测法、因果图法、判定表、正交实验设计法、功能图法、场景法等。

3.2 等价类划分法

等价类划分法是一种最常用的黑盒测试方法之一。等价类划分法（Equivalence Partitioning）是把程序的输入域划分成若干部分（子集），然后从每个部分中选取少数代表性数据作为测试用例。每一类的代表性数据在测试中的作用等价于这一类中的其他值。

3.2.1 划分等价类

等价类是指某个输入域的子集合。在该子集合中，各个输入数据对于揭露程序中的错误都是等效的，并合理地假定：测试某等价类的代表值就等于对这一类其他值的测试。因此，可以把全部输入数据合理划分为若干等价类，在每一个等价类中取一个数据作为测试的输入条件，就可以用少量代表性的测试数据，取得较好的测试结果，等价类划分可有两种不同的情况：有效等价类和无效等价类。

有效等价类：是指对于程序的规格说明来说是合理的、有意义的输入数据构成的集合，利用有效等价类可检验程序是否实现了规格说明中所规定的功能和性能。

无效等价类：与有效等价类的定义恰巧相反。

例如，输入值是学生成绩，范围是 0～100，其有效等价类和无效等价类划分，可以确定一个有效等价类和两个无效等价类。小于 60 分和大于 100 分为无效等价类，大于或等于 60 分且小于或等于 100 分为有效等价类。

设计测试用例时，要同时考虑这两种等价类。因为软件不仅要能接收合理的数据，也要能经受意外的考验，这样的测试才能确保软件具有更高的可靠性。

下面给出六条确定等价类的原则。

(1) 在输入条件规定了取值范围或值的个数的情况下，可以确立一个有效等价类和两个无效等价类。

(2) 在输入条件规定了输入值的集合或者规定了“必须如何”的条件的情况下，可确立一个有效等价类和一个无效等价类。

(3) 在输入条件是一个布尔量的情况下，可确定一个有效等价类和一个无效等价类。

(4) 在规定了输入数据的一组值(假定 n 个)，并且程序要对每一个输入值分别处理的情况下，可确立 n 个有效等价类和一个无效等价类。

(5) 在规定了输入数据必须遵守的规则的情况下，可确立一个有效等价类(符合规则)和若干个无效等价类(从不同角度违反规则)。

(6) 在确知已划分的等价类中各元素在程序处理中的方式不同的情况下，则应再将该等价类进一步划分为更小的等价类。

3.2.2 设计测试用例

在确立了等价类后，可建立等价类表，列出所有划分出的等价类，如表 3-1 所示。

表 3-1 等价类表

输 入 条 件	有效等价类	无效等价类
⋮	⋮	⋮

然后从划分出的等价类中按以下三个原则设计测试用例。

(1) 为每一个等价类规定一个唯一的编号。

(2) 设计一个新的测试用例，使其尽可能多地覆盖尚未被覆盖的有效等价类，重复这一步，直到所有的有效等价类都被覆盖为止。

(3) 设计一个新的测试用例，使其仅覆盖一个尚未被覆盖的无效等价类，重复这一步，直到所有的无效等价类都被覆盖为止。

3.2.3 等价类划分法举例

1. 登录窗口

以某“学生成绩信息管理系统”为例，登录窗口的界面如图 3-3 所示。

图 3-3 登录窗口

在登录窗口中不考虑身份选择情况,只验证“用户名”“请输入密码”和“请确认密码”的正确性。用户名和密码的输入条件均要求为不超过16位,可以使用汉字、英文字母和数字及各种组合,输入密码和确认密码相同。

首先,等价类划分法对用户名和密码进行等价类划分,建立等价类表,如表3-2所示。在某网站申请免费信箱时,要求用户必须输入用户名、密码及确认密码,对每一项输入条件的要求如下。

(1) 用户名要求为4～16位,可使用英文字母、数字、-、_,并且首字符必须为字母或数字。

(2) 密码要求为6～16位,只能使用英文字母、数字或-、_,并且区分大小写。

其次,根据等价类表生成测试用例,如表3-3所示。

表 3-2 等价类表

输入条件	有效等价类	编号	无效等价类	编号
用户名	4～16位	1	少于4位	9
		2	多于16位	10
	首字符为字母	3	首字符为除字母、数字之外的其他字符	11
	首字符为数字	4	组合中含有除英文字母、数字、-、_之外的其他特殊字符	12
请输入密码	英文字母、数字、-、_组合	5	少于6位	13
		6	多于16位	14
	英文字母、数字、-、_组合	7	组合中含有除英文字母、数字、-、_之外的其他特殊字符	15
请确认密码	内容与密码相同	8	内容与输入密码相同,但确认密码字母大小写不同	16

表 3-3 测试用例

测试用例	用 户 名	密 码	确认密码	预期输出	覆盖的等价类
TC1	ABC_2000	ABC_123	ABC_123	注册成功	1,2,4,5,6,7,8
TC2	2000-ABC	123-ABC	123-ABC	注册成功	1,3,4,5,6,7,8
TC3	ABC	12345678	12345678	提示用户名错误	9
TC4	ABC123456789012345	12345678	12345678	提示用户名错误	10
TC5	_ABC123	12345678	12345678	提示用户名错误	11
TC6	ABC&123	12345678	12345678	提示用户名错误	12

续表

测试用例	用 户 名	密 码	确 认 密 码	预 期 输 出	覆盖的等价类
TC7	ABC_123	12345	12345	提示密码错误	13
TC8	ABC_123	ABC123456	ABCDEFGHIJK123456	提示密码错误	14
TC9	ABC_123	ABC&123	ABC&123	提示密码错误	15
TC10	ABC_123	ABC_123	ABC_123	提示密码错误	16

2. DIMENSION 语句

FORTRAN 编译系统的设计和程序编写工作已经完成，现需对 DIMENSION 语句的实现设计测试用例。已知 DIMENSION 语句的语法规则是：DIMENSION 语句用以规定数组的维数。形式为：

```
DIMENSION AD[;AD]…
```

其中，AD 是数组描述符，其形式为：

```
n(d [,d] …)
```

其中，n 是数组名，由 1～6 个字母或数字组成。为首的必须是字母；d 是维数说明符，数组维数最大为 7，最小为 1，它的形式为 [lb:]ub。

lb 和 ub 分别表示数组下界和上界，均为－65 534～65 535 的整数，也可以是整型变量名(但不可以是数组元素名)。若未规定 lb，则认为其值为 1，且 ub≥lb。若已规定了 lb，则它可为负数、零或正数。DIMENSION 语句也和其他语句一样，可连续写多行。

注释：以上规则中，[ ]内为任选项，小写字母代表语法单位，…表示它前面的项可重复出现多次。

首先，确定输入条件，并确定等价类，如表 3-4 所示(注：括号中数字为等价类编号)。

表 3-4 等价类表

输 入 条 件	有效等价类	无效等价类
数组描述符个数	1(1)，>1(2)	无数组描述符(3)
数组名称符个数	1～6(4)	0(5)，>6(6)
数组名	有字母(7)，有数字(8)	有其他字符(9)
数组名以字母开头	是(10)	否(11)
数组维数	1～7(12)	0(13)，>7(14)
上界取值	常数(15)，整型变量(16)	数组元素名(17)，其他(18)
数组变量名	有字母(19)，有数字(20)	其他(21)
整型变量名以字母开头	是(22)	否(23)

续表

输 入 条 件	有效等价类	无效等价类
上下界取值	−65 534～65 535(24)	＜−65 534(25),＞65 535(26)
是否定义下界	是(27),否(28)	
上界对下界关系	＞(29),＝(30)	＜(31)
下界定义	负数(32),0(33),正数(34)	
下界取值	常数(35),整型变量(36)	数组元素名(37),其他(38)
语句多于一行	是(39),否(40)	

其次,确定覆盖有效等价类的测试用例。每一个测试用例覆盖一个或多个有效等价类。

测试用例编号(1):DIMENSION A(2)。

覆盖有效等价类:1,4,7,10,12,15,24,28,29,40。

测试用例编号(2):DIMENSION A12345(1,9,J4YYY,65 535,H,JKL,100),BB(−65 534:100,0:1000,10:10,1:65 535)。

覆盖其余有效等价类:2,8,16,19,20,22,27,30,32,33,34,35,36,39。

第三,确定覆盖无效等价类的测试用例。每一个测试用例覆盖一个无效等价类,如表3-5所示。

表 3-5　覆盖无效等价类的测试用例

编号	输 入 条 件	输 入 数 据	覆盖无效等价类
3	数组描述符个数——无数组描述符	DIMENSION	3
4	数组名称符个数——0	DIMENSION (10)	5
5	数组名称符个数——＞6	DIMENSION A12345678(2)	6
6	数组名——有其他字符	DIMENSION A.1(2)	9
7	数组名以字母开头——否	DIMENSION 1A(10)	11
8	数组维数——0	DIMENSION B	13
9	数组维数——＞7	DIMENSION B(8,8,8,8,8,8,8,8,8)	14
10	上界取值——数组元素名	DIMENSION B(4,A(2))	17
11	上界取值——其他	DIMENSION B(4,7)	18
12	数组变量名——其他	DIMENSION C(R＊S,10)	21
13	整型变量名以字母开头——否	DIMENSION C(10,3L)	23
14	上下界取值—— ＜−65 534	DIMENSION D(−65535:1)	25
15	上下界取值—— ＞65 535	DIMENSION D(65536)	26
16	上界对下界关系—— ＜	DIMENSION D(4:3)	31
17	下界取值——数组元素名	DIMENSION D(A(2):4)	37
18	下界取值——其他	DIMENSION D(:4)	38

3. 三角形问题

输入三个整数 a、b 和 c 分别作为三角形的三条边，通过程序判断由这三条边构成的三角形类型是：等边三角形、等腰三角形、一般三角形或非三角形(不能构成一个三角形)。另外，假定三个输入 a、b 和 c 在 1～100 中取值(整数)。

下面对题目进行更详细的分析。

输入三个整数 a、b 和 c 分别作为三角形的三条边，要求 a、b 和 c 必须满足以下条件。

(1) 整数。

(2) 三个数。

(3) 边长大于或等于 1 且小于或等于 100。

(4) 任意两边之和大于第三边。

输出为以下 5 种情况之一。

(1) 如果不满足条件 1、2、3，则程序输出为“输入错误”。

(2) 如果不满足条件 4，则程序输出为“非三角形”。

(3) 如果三条边相等，则程序输出为“等边三角形”。

(4) 如果恰好有两条边相等，则程序输出为“等腰三角形”。

(5) 如果三条边都不相等，则程序输出为“一般三角形”。

输入域等价类划分和输出域等价类划分如表 3-6 所示。

表 3-6 输入域等价类划分和输出域等价类划分

有效等价类	编 号	无效等价类	编 号
a 为整数	1	a 非整数	13
b 为整数	2	b 非整数	14
c 为整数	3	c 非整数	15
三个数	4	大于 3	16
		小于 3	17
$1 \leqslant a \leqslant 100$	5	小于 1	18
		大于 100	19
$1 \leqslant b \leqslant 100$	6	小于 1	20
		大于 100	21
$1 \leqslant c \leqslant 100$	7	小于 1	22
		大于 100	23
两边之和大于第三边	8	$a+b<c$	24
		$a+c<b$	25
		$b+c<a$	26
非三角形	9		
等边三角形	10		
等腰三角形	11		
一般三角形	12		

覆盖有效等价类的测试用例如表 3-7 所示，覆盖无效等价类的测试用例如表 3-8 所示。

表 3-7　覆盖有效等价类的测试用例

测试用例编号	输入数据	输出结果	覆盖的有效等价类
TC1	30,30,30	等边三角形	1,2,3,4,5,6,7,8,10
TC2	30,30,20	等腰三角形	1,2,3,4,5,6,7,8,11
TC3	30,40,50	一般三角形	1,2,3,4,5,6,7,8,12
TC4	30,40,90	非三角形	1,2,3,4,5,6,7,9

表 3-8　覆盖无效等价类的测试用例

测试用例编号	输入数据	输出结果	覆盖的无效等价类
TC5	11.1,10,10	输入错误	13
TC6	10,5.5,10	输入错误	14
TC7	9,10,3.3	输入错误	15
TC8	10,10,10,4	输入错误	16
TC9	10,10	输入错误	17
TC10	0,10,10	输入错误	18
TC11	101,50,50	输入错误	19
TC12	10,－1,10	输入错误	20
TC13	50,110,50	输入错误	21
TC14	10,10,0	输入错误	22
TC15	50,50,110	输入错误	23
TC16	10,10,50	输入错误	24
TC17	10,60,10	输入错误	25
TC18	110,10,30	输入错误	26

3.3　边界值分析法

边界值分析法是用于对输入或输出的边界值进行测试的一种黑盒测试方法。在测试过程中，边界值分析法是作为对等价类划分法的补充，专注于每个等价类的边界值，两者的区别在于前者在等价类中随机选取一个测试点。边界值分析法采用一到多个测试用例来测试一个边界，不仅重视输入条件边界值，而且重视输出域。边界值分析法比较简单，仅用于考察正处于等价划分边界或边界附近的状态，考虑输出域边界产生的测试情况，针对各种边界情况设计测试用例，发现更多的错误。边界值分析法的测试用例是由等价类

的边界值产生的，根据输入/输出等价类，选取稍高于边界值或稍低于边界值等特定情况作为测试用例。

3.3.1 边界值分析法的含义

在等价类划分基础上进行边界值分析测试的基本思想是，选取正好等于、刚刚大于或刚刚小于等价类边界的值作为测试数据，而不是选取等价类中的典型值或任意值为测试数据。

边界值分析法(Boundary Value Analysis)是一种补充等价类划分法的测试用例设计技术，它不注重选择等价类的任意元素，而是注重选择等价类边界的测试用例。在测试过程中，可能会忽略边界值的条件，大量的错误是发生在输入或输出范围的边界上，而不是发生在输入/输出范围的内部。因此针对各种边界情况设计测试用例，可以查出更多的错误。

边界值测试主要考虑以下几条原则。

(1) 如果输入条件规定了值的范围，则应取刚达到这个范围边界的值，以及刚刚超过这个范围边界的值作为测试输入数据。

例如，一个单位对身高的要求是1.70～1.90m，则测试用例为1.69、1.70、1.71、1.89、1.90、1.91，以及典型值1.80。

(2) 如果输入条件规定了值的个数，则用最大个数、最小个数、比最小个数小1的数、比最大个数大1的数作为测试数据。

例如，一个系统规定可以存储文件1～128个，则测试用例为0、1、128、129。

(3) 如果程序的规格说明给出的输入域或输出域是有序集合，则应选取集合的第一个元素和最后一个元素作为测试用例。

例如，假设C语言中数组长度为n，则测试用例是数组下标为0和数组下标为$n-1$。

(4) 如果程序中使用了一个内部数据结构，则应当选择这个内部数据结构的边界上的值作为测试用例。

(5) 分析程序规格说明，找出其他可能的边界条件。

边界值和等价类密切相关，输入等价类和输出等价类的边界是要着重测试的边界情况。在等价类的划分过程中产生了许多等价类边界。边界是最容易出错的地方，所以，从等价类中选取测试数据时应该关注边界值。

边界值分析法的必要性体现在，软件测试常用的一个方法是把测试工作按同样的形式划分。对数据进行软件测试，就是检查用户输入的信息、返回结果以及中间计算结果是否正确。实践表明，输入域的边界值比中间的值更加容易发现错误。

3.3.2 边界值分析法原理

1. 边界值分析测试

这里讨论有两个变量X_1和X_2的程序P。假设输入变量X_1和X_2在下列范围内

取值:

$$a \leqslant X_1 \leqslant b, c \leqslant X_2 \leqslant d$$

边界值分析利用输入变量的最小值(min)、稍大于最小值(min+)、域内任意值(nom)、稍小于最大值(max-)和最大值(max)来设计测试用例。即通过使所有变量取正常值,只使一个变量分别取最小值、略高于最小值、略低于最大值和最大值,如图 3-4 所示。

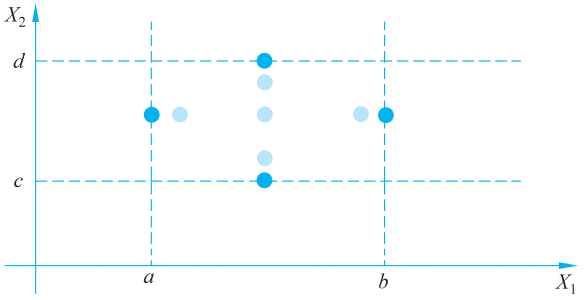

图 3-4 边界值分析法测试用例

对于一个 n 个变量的程序,边界值分析测试会产生 $4n+1$ 个测试用例。

2. 健壮性边界值测试

健壮性测试是边界值分析的一种扩展。变量除了取 min、min+、nom、max-和 max 五个边界值外,还要考虑采用一个略超过最大值(max+)以及一个略小于最小值(min-)的取值,看看超过极限值时系统会出现什么情况,如图 3-5 所示。

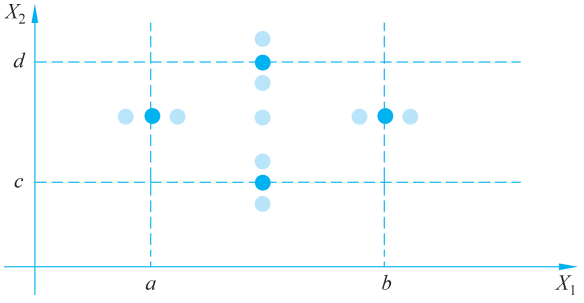

图 3-5 健壮性边界值分析法测试用例

健壮性测试最有意义的部分不是输入,而是预期的输出,观察例外情况如何处理。

健壮性边界值测试将产生 $6n+1$ 个测试用例。

3. 最坏情况边界值分析法

对每一个变量首先进行包含最小值、略高于最小值、正常值、略低于最大值、最大值五个元素集合的测试,然后对这些集合进行笛卡儿积计算,以生成测试用例。最坏情况测试显然更彻底,但测试工作量较大。

一个变量个数为 n 的最坏情况测试会产生 5^n 个测试用例，如图 3-6 所示。

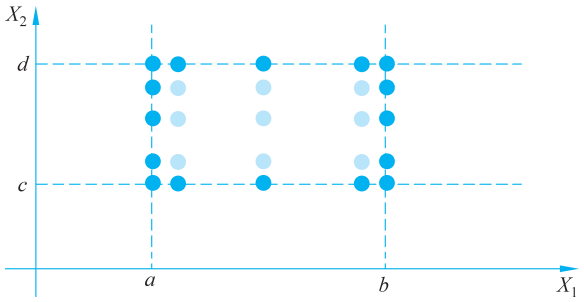

图 3-6　最坏情况边界值分析法测试用例

4. 健壮最坏情况测试

对每一个变量，首先进行包含最小值、略高于最小值、正常值、略低于最大值、最大值五个元素集合的测试，还要采用一个略超过最大值的取值，以及一个略小于最小值的取值。然后对这些集合进行笛卡儿积计算，以生成测试用例。

n 个变量的函数的健壮最坏情况测试会产生 7^n 个测试用例，如图 3-7 所示。

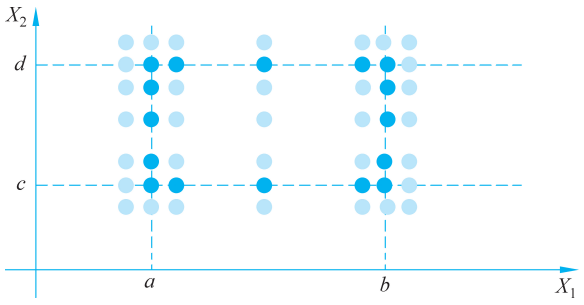

图 3-7　健壮最坏情况边界值分析法测试用例

3.3.3　边界值分析法举例

1. NextDate()函数

NextDate()是一个有三个变量(月份、日期和年)的函数。函数返回输入日期后面的那个日期。变量月份、日期和年都为整数，且 NextDate()函数规定了 Month、Day、Year 相应的取值范围，即 $1 \leqslant Month \leqslant 12$，$1 \leqslant Day \leqslant 31$，$1912 \leqslant Year \leqslant 2050$，NextDate()函数的边界值分析法测试用例如表 3-9 所示。

2. 三角形问题

Glenford Myers 的经典著作 *The Art of Software Testing* 中描述了三角形的问题。

表 3-9　NextDate()函数的边界值分析法测试用例

测试用例编号	Month	Day	Year	预 期 输 出
TC1	6	15	1911	Year 超出[1912,2050]
TC2	6	15	1912	1912,6,16
TC3	6	15	1913	1913,6,16
TC4	6	15	1975	1975,6,16
TC5	6	15	1949	1949,6,16
TC6	6	15	1950	1950,6,16
TC7	6	15	1951	Year 超出[1912,2050]
TC8	6	−1	2001	Day 超出[1,31]
TC9	6	1	2001	2001,6,2
TC10	6	2	2001	2001,6,3
TC11	6	30	2001	2001,7,1
TC12	6	31	2001	输入日期超过范围
TC13	6	32	2001	Day 超出[1,31]
TC14	−1	15	2001	Month 超出[1,12]
TC15	1	15	2001	2001,1,16
TC16	2	15	2001	2001,2,16
TC17	11	15	2001	2001,11,16
TC18	12	15	2001	2001,12,16
TC19	13	15	2001	Month 超出[1,12]

输入三个整数 a、b 和 c,分别作为三角形的三条边,通过程序判断由这三条边构成的三角形类型:等边三角形、等腰三角形、一般三角形或非三角形(不能构成一个三角形)。

另外,假定三个输入 a、b 和 c 在 1～100 中取值(整数),覆盖有效等价类的测试用例如表 3-10 所示。

表 3-10　覆盖有效等价类的测试用例

测试用例编号	*a*	*b*	*c*
TC1	50	50	1
TC2	50	50	2
TC3	50	50	50
TC4	50	50	99
TC5	50	50	100

续表

测试用例编号	*a*	*b*	*c*
TC6	50	1	50
TC7	50	2	50
TC8	50	99	50
TC9	50	100	50
TC10	1	50	50
TC11	2	50	50
TC12	99	50	50
TC13	100	50	50

在进行等价类分析时，往往先要确定边界。如果不能确定边界，就很难定义等价类所在的区域。只有边界值确定下来，才能划分出有效等价类和无效等价类。边界确定清楚了，等价类就自然产生了。边界值分析方法是对等价类划分法的补充。在测试中，会将两个方法结合起来共同使用。

3.4 决策表法

决策表法是把作为条件的所有输入的各种组合值以及对应输出值都罗列出来而形成的表格。它能够将复杂的问题按照各种可能的情况全部列举出来，简明并避免遗漏。因此，利用决策表能够设计出完整的测试用例集合。在所有的黑盒测试方法中，基于决策表的测试是最严格、最具有逻辑性的测试方法。

3.4.1 决策表的含义

决策表(Decision Table)又称判定表，是一种呈表格状的图形工具，适用于描述处理判断条件较多，各条件又相互组合、有多种决策方案的情况。精确而简洁描述复杂逻辑的方式，将多个条件与这些条件满足后要执行的动作相对应。但不同于传统程序语言中的控制语句，决策表能将多个独立的条件和多个动作直接的联系清晰地表示出来。

决策表由条件桩、条件项、动作桩和动作项组成，如图 3-8 所示。

条件桩：列出所有的问题，即条件。

条件项：针对条件桩中的条件列出所有的取值。

动作桩：列出针对问题可能采取的操作。

动作项：针对条件项中的各组取值列出所要采取的动作。

规则：条件项和动作项中的一列称为一条规则。

图 3-8 决策表组成

决策表法测试适用于具有以下特征的应用程序。

(1) if-then-else 结构。

(2) 输入变量之间存在逻辑关系。
(3) 涉及输入变量子集的计算。
(4) 输入和输出之间存在因果关系。
适用于使用决策表设计测试用例的情况。
(1) 规格说明以决策表形式给出,或者较容易转换为决策表。
(2) 条件的排列顺序不会也不应该影响执行的操作。
有 n 个条件的决策表,对应的规则将有 2^n 条。

3.4.2 决策表法举例

创建决策表有5个步骤:
(1) 列出所有的条件桩和动作桩。
(2) 确定规则的个数。
(3) 填入输入项。
(4) 填入动作项,得到初始的决策表。
(5) 对初始的决策表化简。

1. 三角形问题

输入三条边 a、b、c,判断是否是三角形;如果是三角形,继续判断是等腰三角形还是等边三角形。试画出决策表,并设计测试用例。

第一步,列出所有的条件桩和行动桩。
三角形问题的条件桩:
(1) $a<b+c$?
(2) $b<a+c$?
(3) $c<a+b$?
(4) $a=b$?
(5) $a=c$?
(6) $b=c$?
三角形问题的动作桩:
(1) 非三角形。
(2) 一般三角形。
(3) 等腰三角形。
(4) 等边三角形。
(5) 不可能。
第二步,确定规则的个数,并填入输入项和动作项。
有 n 个条件的决策表,对应的规则将有 2^n 条,本题的规则数 $2^6=64$。
第三步,生成决策表及简化的决策表。
当 n 非常大的时候,生成的决策表非常大。因此,应对决策表进行化简。化简的原

则是如果决策表中有两条规则相同则生成简化的决策表，如表 3-11 所示。

表 3-11 “三角形问题”简化的决策表

条件与动作	内容	1～32	33～48	48～56	57	58	59	60	61	62	63	64
条件	$a<b+c$?	F	T	T	T	T	T	T	T	T	T	T
	$b<a+c$?		F	T	T	T	T	T	T	T	T	T
	$c<a+b$?			F	T	T	T	T	T	T	T	T
	$a=b$?				T	T	T	T	F	F	F	F
	$a=c$?				T	T	F	F	T	T	F	F
	$b=c$?				T	F	T	F	T	F	T	F
动作	非三角形	√	√	√								
	一般三角形											√
	等腰三角形							√		√	√	
	等边三角形				√							
	不可能					√	√		√			

最后，生成测试用例，如表 3-12 所示。

表 3-12 “三角形问题”的测试用例

编　号	*a*	*b*	*c*	预 期 输 出
1	5	1	2	非三角形
2	1	5	2	非三角形
3	1	2	5	非三角形
4	5	5	5	等边三角形
5	—	—	—	不可能
6	—	—	—	不可能
7	2	2	3	等腰三角形
8	—	—	—	不可能
9	2	3	2	等腰三角形
10	3	2	2	等腰三角形
11	3	4	5	一般三角形

2. 成绩录入窗口

某信息科学与技术学院成绩录入窗口如图 3-9 所示，其需求规格说明包括三个下拉列表，分别用于显示各学院名称、各系部名称及各班级名称。只有选择了某一个学院后，系部列表框才为可用，列表中将显示出所选择学院对应的所有系部；同样，只有选择了某一个学院，又选择了某一个系部后，此时班级列表框才为可用，列表中将显示出所选择系部对应的所有班级。当三个选项都已经完成选择后，界面则会显示出所选班级名单，这时可录入成绩。

图 3-9　某信息科学与技术学院成绩录入窗口

操作步骤如下。

第一步，列出所有的条件桩和行动桩。

由规格说明可以分析出，输入事件即条件桩。

C1：选择学院。

C2：选择系部。

C3：选择班级。

输出事件及行动桩。

a1：显示所选班级名单。

a2：学院列表框可用。

a3：系部列表框可用。

a4：班级列表框可用。

a5：显示各学院名称。

a6：显示各系部名称。

a7：显示各班级名称。

a8：不能显示具体选项（如没有选择学院，系部列表框中将不能显示所对应系部）。

第二步，确定规则的个数，并填入输入项和动作项。

本题的规则数为 $2^3=8$。

第三步，生成决策表及简化的决策表，建立如表 3-13 所示的决策表。

表 3-13　决策表

选　　项	1	2	3	4	5	6	7	8
选择学院	T	T	T	T	F	F	F	F
选择系部	T	T	F	F	T	T	F	F
选择班级	T	F	T	F	T	F	T	F
显示所选班级名单	√							
学院列表框可用		√	√	√	√	√	√	√
系部列表框可用		√	√	√				

续表

选　　项	1	2	3	4	5	6	7	8
班级列表框可用		√						
显示各学院名单		√	√	√				
显示各系部名单								
显示各班级名单								
不能形式具体选项			√		√	√	√	

最后，生成的成绩录入窗口测试用例如表 3-14 所示。

表 3-14　成绩录入窗口测试用例

测试用例	操 作 描 述	输 入 数 据	预 期 输 出
TC1	单击并选择学院 单击并选择系部 单击并选择班级	学院：信息科学与技术学院 系部：软件工程系 班级：信息 1201	学院列表可用 系部列表可用 班级列表可用
TC2	单击并选择学院 单击并选择系部 单击但不选择班级	学院：信息科学与技术学院 系部：软件工程系 班级：信息 1201	学院列表可用 系部列表可用 班级列表可用
TC3	单击并选择学院 单击但不选择系部 单击并选择班级	学院：信息科学与技术学院 系部：空 班级：空	学院列表可用 系部列表可用 班级列表不能显示对应数据项
TC4	单击并选择学院 单击但不选择系部 单击但不选择班级	学院：信息科学与技术学院 系部：空 班级：空	学院列表可用 系部列表可用 班级列表不能显示对应数据项
TC5	单击但不选择学院 单击并选择系部 单击并选择班级	学院：空 系部：空 班级：空	学院列表可用 系部列表不能显示对应数据项 班级列表不能显示对应数据项
TC6	单击但不选择学院 单击并选择系部 单击但不选择班级	学院：空 系部：空 班级：空	学院列表可用 系部列表不能显示对应数据项 班级列表不能显示对应数据项
TC7	单击但不选择学院 单击但不选择系部 单击并选择班级	学院：空 系部：空 班级：空	学院列表可用 系部列表不能显示对应数据项 班级列表不能显示对应数据项
TC8	单击但不选择学院 单击但不选择系部 单击但不选择班级	学院：空 系部：空 班级：空	学院列表可用 系部列表不能显示对应数据项 班级列表不能显示对应数据项

决策表法测试的优点是能把复杂的问题按各种可能的情况一一列举出来，简明而易于理解，也可避免遗漏，其缺点是不能表达重复执行的动作，例如循环结构。

3.5 因果图分析法

前面介绍的等价类划分和边界值分析这两种方法并没有考虑到输入情况的各种组合，也没有考虑到各个输入情况之间的依赖关系。输入条件之间的相互组合，可能会产生一些新的情况。前面两种测试方法可以检测到各个输入条件可能出错的情况，却忽略了多个条件组合起来时出错的情况。但要检查输入条件的组合不是一件容易的事情，即使把所有输入条件划分成等价类，它们之间的组合情况也相当多。因此必须考虑采用一种适合于描述对于多种条件的组合，相应产生多个动作的形式来考虑设计测试用例，这就是因果图分析法。

3.5.1 因果图法的含义

因果图法即因果分析图，又叫特性要因图、石川图或鱼翅图，它是由日本东京大学教授石川馨提出的一种通过带箭头的线，将质量问题与原因之间的关系表示出来，是分析影响产品质量的诸因素之间关系的一种工具。从用自然语言书写的程序规格说明的描述中找出因（输入条件）和果（输出或程序状态的改变），可以生成因果图。

因果图法是一种适合于描述对于多种输入条件组合的测试方法，根据输入条件的组合、约束关系和输出条件的因果关系，分析输入条件的各种组合情况，从而设计测试用例的方法，它适合于检查程序输入条件涉及的各种组合情况。因果图法一般和决策表结合使用，通过映射同时发生相互影响的多个输入来确定判定条件。因果图法最终生成的就是决策表，它适合于检查程序输入条件的各种组合情况。采用因果图法能帮助我们按照一定的步骤选择一组高效的测试用例，同时，还能指出程序规范中存在什么问题，鉴别和制作因果图。

因果图法着重分析输入条件的各种组合，每种组合条件就是“因”，它必然有一个输出的结果，这就是“果”。

利用因果图导出测试用例一般要经过以下几个步骤。

（1）分析软件规格说明的描述中哪些是原因，哪些是结果。原因是输入条件的等价类，结果是输出条件。给每个原因和结果并赋予一个标识符，根据这些关系画出因果图。

（2）因果图上用一些记号表明约束条件或限制条件。

（3）对需求加以分析并把它们表示为因果图之间的关系图。

（4）把因果图转换成决策表。

（5）将决策表的每一列作为依据，设计测试用例。

3.5.2 因果图法的原理

因果图法中使用的基本符号如图 3-10 所示。左结点表示输入状态即原因，右结点表示输出状态即结果。

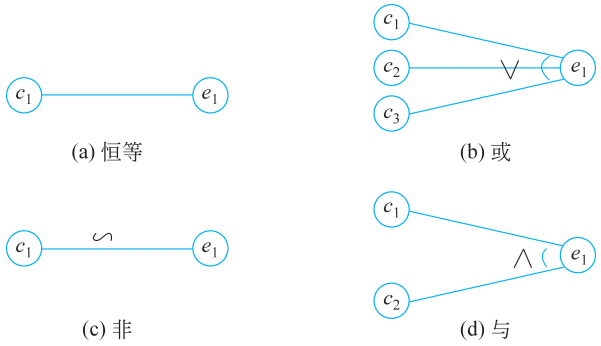

图 3-10　因果图法中使用的基本符号

恒等：如果 c_1 是 1，则 e_1 也是 1；否则 e_1 是 0。

或 ∨：如果 c_1 或 c_2 或 c_3 是 1，则 e_1 也是 1；否则 e_1 是 0。

非 ～：如果 c_1 是 1，则 e_1 是 0；否则 e_1 是 1。

与 ∧：如果 c_1 和 c_2 是 1，则 e_1 也是 1；否则 e_1 是 0。

约束：在实际问题中，输入状态之间还可能存在某些依赖关系，称为约束。例如，某些输入条件不可能同时出现。在因果图中用一些特殊的符号表示这些约束，如图 3-11 所示。

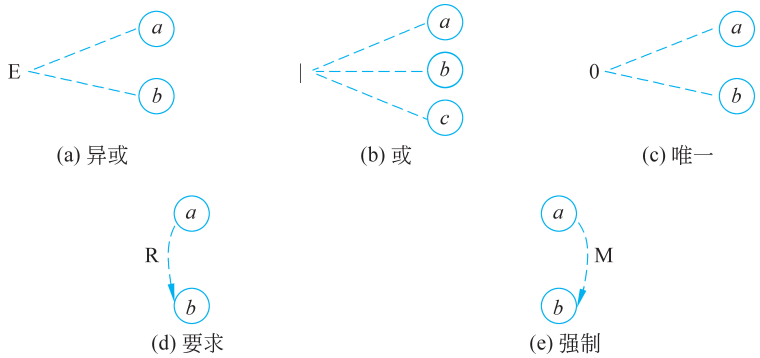

图 3-11　因果图中的约束符号

输入条件的约束如下。

E(Exclusive，异或)：表示至多 1 个为 1。如在图 3-11(a)中，a 和 b 只能有一个为 1。

I(Inclusive，或)：表示至少 1 个为 1。如在图 3-11(b)中，a 和 b 至少有一个为 1。

O(One and Only，唯一)：必有且只有一个为 1。如在图 3-11(c)中，a 和 b 必须有且仅有一个为 1。

R(Require，要求)：表示 a 是 1，则 b 必须是 1，参见图 3-11(d)。

输出条件的约束如下。

M(Mask，强制)：表示 a 是 1，则 b 必须是 0，参见图 3-11(e)。

3.5.3 因果图法举例

1. 某个软件规格说明书

某个软件规格说明书中规定：第一列字符必须是 * 或 #，第二列字符必须是一个数字，在此情况下进行文件的修改。但如果第一列字符不正确，则给出信息 M；如果第二列字符不正确，则给出信息 N。

软件测试的设计步骤如下。

首先，分析软件规格说明书找出原因和结果。

原因：

1——第一列字符是 * 。

2——第一列字符是 # 。

3——第二列字符是一个数字。

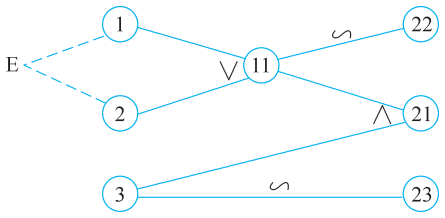

图 3-12 因果图

结果：

21——修改文件。

22——给出信息 M。

23——给出信息 N。

其次，找出原因和结果之间的因果关系，原因与原因之间的约束关系，画出如图 3-12 所示的因果图。

第三，根据因果图建立如表 3-15 所示的决策表和测试用例。

表 3-15 根据因果图建立的决策表和测试用例

列 项		1	2	3	4	5	6
原因	1	1	1	0	0	0	0
	2	0	0	1	1	0	0
	3	1	0	1	0	1	0
结果	21	1	0	1	0	0	0
	22	0	0	0	0	1	1
	23	0	1	0	1	0	1
测试用例		* 3	* M	# 5	# N	C2	DY
		ok	N	ok	N	M	M，N

2. 中国象棋中跳马

中国象棋中跳马的规则如下。