第3章

CHAPTER

黑盒测试

本章学习目标

- 认识与理解软件黑盒测试的概念、主要特点与应用策略
- 学习与掌握运用等价类划分方法设计测试用例
- 学习与掌握运用边界值分析方法设计测试用例
- 学习与掌握运用决策表方法设计测试用例
- 学习与掌握运用因果图方法设计测试用例

软件的黑盒测试方法即把被测对象(程序、模块)比作一个看不见内部结构特征的黑盒子,在完全不考虑被测对象内部结构或内部特征的情况下,检测程序是否能适当地接收输入数据而产生正确的输出信息。这就好比一个人使用照相机拍照,他不需要知道照相机内部的复杂工作方式,只是通过拍出来的照片情况判断照相机的拍照功能是否能实现。可见,黑盒测试着眼于对程序的输出结果进行验证,不考虑程序内部的逻辑结构及实现细节。

假设有图 3.1 所示的空纸杯,需要对它进行测试,应该如何开展? 首先,需要确认用户用纸杯主要做什么事情。打个比方,纸杯是用于饮水,还是用于浇花,还是仅用于书桌台面装饰? 如果用户对纸杯的主要需求是饮水,需要测试的结果是纸杯在装水时是否漏水,冷、热水是否都能装,能否装其他饮料(如可乐、果汁、奶茶)等。当然,如果用户只是把纸杯作为装饰物,用于对书桌的台面装饰,实际上只需要测试这个空纸杯能否平稳地放在桌面上,纸杯的花纹颜色能否被用户所接受等。由此可见,对空纸杯的测试的结果完全取决于用户实际需求。所以说,黑盒测试主要对软件产品的各项功能进行验证,用于检测产品的功能是否能达到用户要求,或者说验证软件的某个功能点是否与用户需求规格说明书上所描述的一致。

黑盒测试相对简单,测试人员完全不用考虑程序的内部结构和内部特性,只是检查程序能否正常接收输入数据,并产生正确的输出信息,程序功能是否能按照用户需求规格说明书的规定正常实现。黑盒测试是从客户

图 3.1 空纸杯

需求出发的测试，基于数据驱动，主要测试软件的功能是否正确（符合用户需求），所以又称软件用户级别的正确性测试。黑盒测试如图 3.2 所示。

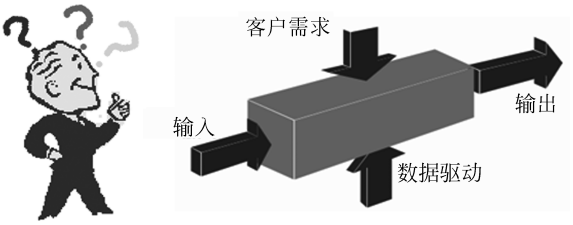

图 3.2 黑盒测试图

黑盒测试主要可以发现以下问题：

① 软件所实现的功能是否完全符合用户需求？是否有被遗漏的（未能实现出的）功能？

② 软件是否忽略了用户的一些隐性需求？

③ 软件的接口或界面是否有错误？

④ 软件能否接收用户输入的任何数据（无论是合法的还是非法的）？能否输出正确的结果？是否会出现用户不能接受的结果？

⑤ 软件是否有数据结构方面的错误？或存在外部信息（如数据文件等）访问错误？

⑥ 软件是否存在功能初始化或异常中止方面的错误？

⑦ 软件的性能、易用性、可靠性以及其他方面的特性是否能够满足用户需求规格说明书上所明确规定的要求？

需要注意的是，使用黑盒测试方法发现软件中的缺陷，必须要在各种可能的输入条件下确定测试数据，来检查程序是否都能产生正确的输出。也就是说，不仅要测试程序在接收合法数据后的输出结果，即测试合理的使用场景，也要对一些不合法的、但是却允许存在或可能被用户输入的非法（错误）数据进行测试，即测试不合理的使用场景。对于测试人员，不能只考虑软件接收了合法数据产生的结果是否符合用户需求，更要考虑软件接收了一些非法的、异常的数据后产生的结果是否与用户所期望的相一致。所以，测试人员在验证完软件的正常功能能否正确实现之后，通常还会进行大量的“破坏性”测试，即选取各类“异常的”数据来“破坏”与“搞垮”软件，观测测试结果，以发现软件中更多的一些隐藏性缺陷。

因为无法做到穷举测试，运用黑盒测试方法设计出的测试用例需要覆盖尽可能多的场景，包括合理的场景与不合理的场景。所以，设计出的测试用例集务必遵循以下两个标准：

① 验证合理的测试场景，所需要设计出的测试用例数目越少越好。

② 验证不合理的场景，所设计出的测试用例能够反映（体现）出当前软件存在哪一种类型的缺陷，而不是仅仅指出与特定测试有关的缺陷是否存在。

黑盒测试的结果只能有两种，即测试通过与测试失败。当然，无论是测试通过还是测试失败，既不能凭借测试人员的主观臆断，也不能由某些日常的生活常识所决定，而是完全取决于用户需求，因为黑盒测试是基于用户的观点，不考虑程序的内部结构，仅仅从输入数据与输出数据的对应关系出发进行测试的。当然，如果用户需求规格说明书上对软件某个功能点的实现要求（规定）描述或定义本身就有误，黑盒测试方法也是发现不了的。

黑盒测试的主要方法有等价类划分方法、边界值分析方法、因果图方法、决策表方法、错误猜测方法等，每种方法各有长处。在实际测试中，综合应用这些方法往往会得到更好的测试效果。

3.1　等价类划分

等价类即指某个输入域的子集合，每一个子集合中的各个输入数据对于测试程序中的错误都是等效的。也就是说，每一个子集合中的代表性数据在测试中的作用等价于该子集合中的其他数据，是测试相同目标或找出（暴露）同一类别软件缺陷的一组测试用例。使用等价类划分方法进行测试时，首先需要在需求规格说明书的基础上把输入域划分成不同的部分（集合），即形成不同的等价类，不同等价类之间不允许有数据交集，再根据等价类设计出测试用例。

3.1.1　划分等价类

划分等价类时，首先需要依据软件（程序）的功能规格说明（需求），结合测试输入条件，从逻辑上将所有可能的输入数据划分为若干部分，即数据输入域，不同的输入域之间不允许有数据交集。然后从每一个数据输入域中选取少数有代表性的数据作为测试输入数据，设计相应的测试用例。使用等价类划分方法设计测试用例需要经历两个步骤：一是通过列出等价类划分表的形式划分出等价类，如表 3.1 所示。二是根据每一个等价类选取有代表性的测试输入数据来设计测试用例。再次强调，同一个等价类中的不同（输入）数据对于发现（揭露）程序中的某一类别错误都是等效的。选取一个等价类中的某个代表性数值进行测试，与选择该等价类中的其他数值是完全等同的。

因此，根据测试输入条件划分相应的等价类需要一种系统化的方法。每个等价类代表了一组可能的测试输入集合。测试人员不用为每个等价类中的每一个数据元素设计一个测试用例，而是根据等价类的属性选择一个有代表性的数据元素测试用例即可。所以，划分等价类时需要严谨，必须确保每一个等价类中的所有元素所产生的测试效果都一样或类似。划分等价类时，分为有效等价类与无效等价类两种类别。

表 3.1 等价类划分表

等价类名称	有效等价类	无效等价类
输入等价类 1	……	……
输入等价类 2	……	……
输入等价类 3	……	……
……	……	……

1. 有效等价类

有效等价类即指输入的数据是合理的、有效的、有意义的，是满足程序输入要求的数据所构成的集合，利用这些数据主要可以验证程序是否实现了其需求规格说明书中所要求满足的功能。在实际问题中，有效等价类的个数往往不止 1 个。例如，某程序的输入条件为大于 150 而小于 200 的整数 x，则有效等价类为 $150<x<200$。再例如，要求输入字符串的首字符 x 是英文字母，则有效等价类分别为①首字符＝{x | x 为大写英文字母}；②首字符＝{x | x 为小写英文字母}。

2. 无效等价类

无效等价类与有效等价类相反，即由不合理的、无效的、无意义的、不满足程序输入要求的等一系列“错误”数据所组成的集合，通常是有效等价类的“反面”。利用无效等价类可以检查软件(程序)功能的实现是否有不符合用户需求规格说明的地方。实际上，利用无效等价类中的数据元素可以检验程序是否具有较好的容错性或较高的可靠性。对于具体问题，一般是通过先确定有效等价类后再划分无效等价类。同样，无效等价类至少要有 1 个，但通常会有多个。例如，上述问题(某程序的输入条件为：大于 150 而小于 200 的整数 x)的无效等价类就有 2 个，分别为①$x\leqslant 150$；②$x\geqslant 200$。当然，上述问题(要求输入字符串的首字符 x 是英文字母)的无效等价类可以为：首字符＝{x | x 为非英文字母}。

3. 等价类划分的原则

前面已经提到，等价类划分的原则是不考虑程序内部结构的，只依据用户程序的规格说明来划分。常见的一些等价类划分原则主要有：

① 按输入数值的分布区间划分。如果规定了输入条件的取值范围(无论是连续的还是离散的)，则有效等价类是符合输入条件的取值范围，无效等价类是不符合输入条件的取值范围。这种按照数值的分布区间来划分等价类的方法相对简单。

例如，某参数的输入范围是 100～300。

有效等价类为{100≤参数≤300}。

无效等价类为{参数<100}；{参数>300}。

再例如，某参数的输入范围是 100～200、300～400、600～850。

有效等价类为{100≤参数≤200}、{300≤参数≤400}、{600≤参数≤850}。

无效等价类为{参数＜100}、{200＜参数＜300}、{400＜参数＜600}、{参数＞850}。

② 按具体的输入数值划分。如果规定了一组输入数据，且程序要对每一个输入值分别进行处理，则必须为每一个输入值确立一个有效等价类，而采用综合描述的形式，针对这组值的“否定”来确立一个无效等价类，即它是所有不允许输入值的集合。

例如，某字符的允许输入值是 A、D、T、S、¥。

有效等价类为{A}、{D}、{T}、{S}、{¥}。

无效等价类为{除 A、D、T、S、¥以外的其他字符}。

③ 按输入数值的集合划分。如果输入条件规定了输入值的集合，或输入条件规定了“必须如何”的情况，则可确定一个有效等价类和一个无效等价类（输入值集合有效值之外）。

例如，发票抬头必须写个人姓名。

有效等价类为{个人姓名}。

无效等价类为{企业名称、事业单位名称、……}。

④ 按输入符号的限制条件或限制规则划分。如果规格说明规定了输入数据必须遵守的规则或限制条件，则可以确立一个有效等价类（输入符号规则）和若干个无效等价类（从不同角度违反输入符号规则）。

例如，某系统对用户 ID 的输入要求为 6 位数字。

有效等价类为{任意 6 位数字}。

无效等价类为{数字长度多于 6 位}、{数字长度少于 6 位}（从 ID 长度的角度违反输入符号规则）。

{6 位全部是其他字符}、{6 位数字中含有 1 到 5 个其他字符}（从 ID 内容的角度违反输入符号规则）。

{小于 6 位的其他字符}、{大于 6 位的其他字符}（同时从 ID 长度与内容的角度违反输入符号规则）。

可见，按输入符号的限制条件（规则）划分等价类的情景较为复杂，但是对无效等价类的划分不是唯一的。需要测试人员能从实际出发，充分理解输入数据的限制条件（规则），尽量从输入逻辑上把无效等价类进一步细化成更小的无效等价类。例如，在本例中，可以把无效等价类{6 位数字中含有 1 到 5 个其他字符}进一步细化为 5 个更小的无效等价类，即{6 位数字中含有 1 个其他字符}、{6 位数字中含有 2 个其他字符}、{6 位数字中含有 3 个其他字符}、{6 位数字中含有 4 个其他字符}、{6 位数字中含有 5 个其他字符}。

以上等价类划分的原则只是在测试时可能遇到的一些常见情况，而实际的测试情况往往千变万化，尤其是不能完全列出所有的无效等价类。测试人员需要从用户的角度正确分析被测程序的功能，也要不断积累测试经验，还要从数据输出的角度充分考虑被测程序的检错功能。

一些高级软件测试书籍中还提到了等价类划分形式的问题，如标准等价类划分形式、健壮等价类划分形式等，感兴趣的读者可以查阅有关书籍。

4. 运用等价类划分方法设计测试用例

首先，确立了测试问题的等价类之后，把针对测试对象的数目众多的各种输入数据（包括有效的和无效的）划分为若干有效等价类与无效等价类，建立相应的等价类表，如表3.1所示，列出所划分等价类的名称。

接下来，在所建立的等价类表的基础上，按照以下步骤设计测试用例。

① 为每一个等价类的名称进行唯一编号。

② 设计一个新的测试用例，要求尽可能多地覆盖尚未被覆盖的有效等价类。重复这一步，直到测试用例覆盖了所有的有效等价类。

③ 设计一个新的测试用例，要求使其仅覆盖一个尚未被覆盖的无效等价类。重复这一步，直至测试用例覆盖了所有的无效等价类。

④ 对于每一个测试用例，设计出对输入数据的预期结果。

3.1.2 运用等价类划分方法设计测试用例举例

1. 报表日期

某商业管理系统的报表要求用户输入年份与月份的数据值来显示当前日期。假设合法的日期限定在2010年1月至2020年12月之间，并规定日期数据值由6位数字组成。其中前4位表示年份，后2位表示月份（如输入201905表示2019年5月份）。要求使用等价类划分法设计测试用例，测试该报表的日期判定功能。

测试用例设计过程如下。

① 按照日期的类型与长度、年份范围、月份范围设计有效的与无效的等价类，并编号，如表3.2所示。

表3.2 日期判定的等价类划分表

输入等价类	有效等价类	无效等价类
日期类型与长度	6位数字 ①	日期中含有非数字的字符 ④ 多于6位数字 ⑤ 少于6位数字 ⑥
年份范围	年份在2010至2020之间 ②	大于2020 ⑦ 小于2010 ⑧
月份范围	月份在01至12之间 ③	大于12 ⑨ 小于01 ⑩

② 设计具体的输入数据，覆盖有效等价类，如表3.3所示。

表3.3 覆盖有效等价类的测试数据表

输入数据	预期结果	覆盖的有效等价类（编号）
201905	显示正确	①②③

③ 设计具体的输入数据，覆盖无效等价类，如表 3.4 所示。

表 3.4　覆盖无效等价类的测试数据表

输入数据	预期结果	覆盖的无效等价类(编号)
20ad05	显示错误	④
2018051	显示错误	⑤
20131	显示错误	⑥
2025	显示错误	⑦
2008	显示错误	⑧
14	显示错误	⑨
00	显示错误	⑩

2. 三角形形状判定

三角形形状判定问题是软件黑盒测试中关于等价类划分的一个经典案例。假设任意输入三个整数 a(1≤a≤50)、b(1≤a≤50)和 c(1≤a≤50)，分别作为一个三角形的三条边，程序判断由这三条边所构成的三角形类型是等边三角形(三边均相等)、等腰三角形(只要任意两边相等)、一般三角形或非三角形(不能构成一个三角形)。

注：在几何上，一个等边三角形亦可看作等腰三角形，但从等价类划分的角度看，本案例把等边三角形与等腰三角形看作是两种不同类别的三角形。此外，为了方便初学者理解，本案例中也不考虑直角三角形的情况。

分析：三个整数 a、b 和 c 分别作为一个三角形的三条边，根据几何中三角形的构成规则(“任意两边之和大于第三边”或者“任意两边之差小于第三边”)以及本案例中对三条边的长度限定，要求 a、b 和 c 必须满足以下基本条件：

条件 1：a(1≤a≤50)。

条件 2：b(1≤b≤50)。

条件 3：c(1≤c≤50)。

条件 4：a＋b＞c。

条件 5：a＋c＞b。

条件 6：b＋c＞a。

输出则只会产生下列 4 种情况之一(4 种情况相互排斥)：

① 输出为“非三角形”(条件 4、条件 5、条件 6 只要有一个不满足)。

② 输出为“等边三角形”(以上 6 个条件同时满足的基础上，且 a＝b＝c)。

③ 输出为“等腰三角形”(以上 6 个条件同时满足的基础上，必须再满足 a＝b、a＝c、b＝c 中的任意一个)。

④ 输出为“一般三角形”(以上 6 个条件同时满足的基础上，且 a≠b≠c)。

也就是说，无论产生以上哪一种结果，其数据的输入都是合法的，是符合题目要求的，均可归结为数据输入的有效等价类范畴，这一点请初学者务必注意。根据上述情况，按照三角形 a、b、c 三边的长度要求，设计有效等价类并编号，如表 3.5 所示。

表 3.5 三角形三边判定有效等价类划分表

	输入条件(a,b,c)	输出结果	编号
有效等价类	输入 3 个正整数,均在 1～50,且 a=b=c	等边三角形	①
	输入 3 个正整数,均在 1～50,且满足 a=b、a=c、b=c 中的任意一个	等腰三角形	②
	输入 3 个正整数,均在 1～50,同时满足 a+b>c、a+c>b、b+c>a(任意两边之和都大于第三边),且 a≠b≠c	一般三角形	③
	输入 3 个正整数,均在 1～50,不满足任意两边之和都大于第三边	非三角形	④

(1) 设计具体的输入数据,覆盖有效等价类,如表 3.6 所示。

表 3.6 覆盖有效等价类的测试数据表

测试用例	输入数据			预 期 结 果	覆盖的有效等价类
	a	b	c		
Test Case1	10	10	10	等边三角形	①
Test Case2	10	10	5	等腰三角形	②
Test Case3	4	3	5	一般三角形	③
Test Case4	10	11	22	非三角形	④

(2) 按照三角形 a、b、c 三边的长度要求设计无效等价类并编号,如表 3.7 所示。

表 3.7 三角形三边判定无效等价类划分表

	输入条件(a,b,c)	输出结果	编号
无效等价类	1≤a, b, c≤50,且其中一边为小数	输入非法	⑤
	1≤a, b, c≤50,且其中二边为小数	输入非法	⑥
	1≤a, b, c≤50,且其中三边为小数	输入非法	⑦
	其中一边小于 1	输入非法	⑧
	其中两边小于 1	输入非法	⑨
	其中三边小于 1	输入非法	⑩
	其中一边大于 50	输入非法	⑪
	其中两边大于 50	输入非法	⑫
	其中三边大于 50	输入非法	⑬
	只输入 1 个数	输入非法	⑭
	只输入 2 个数	输入非法	⑮
	输入 3 个以上的数	输入非法	⑯
	输入非数值型的数据	输入非法	⑰

(3) 设计具体的输入数据,覆盖无效等价类,如表 3.8 所示。

表 3.8　覆盖无效等价类的测试数据表

测试用例	输入数据			预 期 结 果	覆盖的无效等价类
	a	b	c		
Test Case5	9.5	9	9	输入非法	⑤
Test Case6	9.5	9.5	10	输入非法	⑥
Test Case7	9.5	9.5	9.5	输入非法	⑦
Test Case8	2	2	0	输入非法	⑧
Test Case9	2	0	0	输入非法	⑨
Test Case10	0	0	0	输入非法	⑩
Test Case11	55	47	47	输入非法	⑪
Test Case12	55	54	46	输入非法	⑫
Test Case13	55	55	55	输入非法	⑬
Test Case14	10			输入非法	⑭
Test Case15	8	8		输入非法	⑮
Test Case16	9	9	9,9	输入非法	⑯
Test Case17	10	10	x	输入非法	⑰

3. 电话号码

假设某地区电话号码由三部分组成,分别是:

① 区号:3 位数字或空白。

② 前缀号码:非 0 或 1 开头的任意 4 位数字。

③ 后缀号码:0～9 中的任意 4 位数字。

被测试程序能接受符合上述要求的任意号码,拒绝任何不符合上述要求的号码。请使用等价类划分方法设计测试用例。

分析:一个完整的电话号码由三部分组成,每一部分需要遵循相应的规则。例如,符合要求的电话号码可以是:63837569(无区号)、02157891264(带区号)。

所以,利用相应的规则划分的三个有效等价类,如表 3.9 所示。

表 3.9　电话号码判定有效等价类划分表

	输 入 条 件		输出	编号
有效等价类	区号	3 位数字	正常	①
		空白	正常	②
	前缀号码	非 0 或 1 开头的任意 4 位数字	正常	③
	后缀号码	0～9 中的任意 4 位数字	正常	④

设计具体的输入数据，覆盖有效等价类，如表 3.10 所示。

表 3.10 电话号码判定有效等价类划分表

测试用例	输入数据			预期结果	覆盖的有效等价类
	区号	前缀号码	后缀号码		
Test Case1		6723	3249	号码正确	②③④
Test Case2	021	6542	2358	号码正确	①③④

根据不符合上述要求的电话号码规则划分的无效等价类，如表 3.11 所示。

表 3.11 电话号码判定无效等价类划分表

	输入条件		输出	编号
无效等价类	区号	有非数字字符(前缀号码与后缀号码规则正确)	非法	⑭
		多于 3 位数字(前缀号码与后缀号码规则正确)	非法	⑮
		少于 3 位数字(前缀号码与后缀号码规则正确)	非法	⑤
	前缀号码	有非数字字符(区号与后缀号码规则正确)	非法	⑥
		数字以 0 开头(区号与后缀号码规则正确)	非法	⑦
		数字以 1 开头(区号与后缀号码规则正确)	非法	⑧
		多于 4 位数字(区号与后缀号码规则正确)	非法	⑨
		少于 4 位数字(区号与后缀号码规则正确)	非法	⑩
	后缀号码	有非数字字符(区号与前缀号码规则正确)	非法	⑪
		多于 4 位数字(区号与前缀号码规则正确)	非法	⑫
		少于 4 位数字(区号与前缀号码规则正确)	非法	⑬

设计具体的输入数据，覆盖有效等价类，如表 3.12 所示。

表 3.12 电话号码判定无效等价类划分表

测试用例	输入数据			预期结果	覆盖的无效等价类
	区号	前缀号码	后缀号码		
Test Case5	A21	6782	2345	号码非法	⑭
Test Case6	0216	6782	2345	号码非法	⑮
Test Case7	02	6782	2345	号码非法	⑤
Test Case8	021	6a82	2345	号码非法	⑥
Test Case9	021	0782	2345	号码非法	⑦
Test Case10	021	1782	2345	号码非法	⑧
Test Case11	021	67823	2345	号码非法	⑨

续表

测试用例	输入数据			预期结果	覆盖的无效等价类
	区号	前缀号码	后缀号码		
Test Case12	021	678	2345	号码非法	⑩
Test Case13	021	6782	A345	号码非法	⑪
Test Case14	021	6782	23456	号码非法	⑫
Test Case15	021	6782	234	号码非法	⑬

3.2　边界值分析

边界值分析方法主要是从数据的定义域的边界数据进行分析，对于合法与不合法的边界数据进行选取和测试，用来检查用户输入的信息、返回的结果以及中间计算结果是否正确。通常边界值分析方法作为对等价类划分法的补充，其测试用例中对输入数值的选取大都来自等价类的边界值。

长期的测试工作经验告诉我们，大量的错误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。

边界值分析方法也是一种有效的黑盒测试方法，可以单独设计测试用例。通常更多的是与等价类划分方法结合使用。

3.2.1　边界值的选取

使用边界值分析方法设计测试用例，首先应该确定等价类的边界情况，即选取的测试数据应该针对程序的边界数值。也就是说，测试数据的选取应该刚好等于、略微大于及略微小于边界值。

从纯数学的角度看，对于按照以输入数值分布区间来划分的等价类而言，测试数据除了选取处在该等价类中间区域的一个正常值之外，还应该选取 4 个数值，即处于等价类区间边界点上的最小值(合法的)、最大值(合法的)以及“略微大于区间边界点”的数值(非法的)、“略微小于区间边界点”的数值(非法的)。

例如，在 3.1.2 节中案例 1 报表日期等价类划分的测试用例中，可以选择 2009、2010、2020、2021 等合法年份边界附近的输入数据作为年份范围的测试数据，选择 00、01、12、13 等合法月份边界附近的输入数据作为月份范围的测试数据。

以上情况比较简单，但是在很多情况下，软件测试包含的边界值会有数字、字符、位置、大小、速度、方位、尺寸、空间等，那么以上类型的边界值选取思路应该为最大/最小、首位/末位、上/下、最快/最慢、最高/最低、最短/最长、空/满等情况。当然，具体的边界值选取情况还要结合测试程序要求分析。

对于只含有一个输入变量 x(min≤x≤max)的程序，采用边界值分析方法，x 取值情

况为：x 取最大值(max)、略微小于最大值(max－)、正常值(normal)、略微大于最小值(min＋)和最小值(min)，即 x 分别取 5 个值进行测试，则一共产生 5 个测试用例。而对于一个含有 n 个输入变量的程序，保留其中一个变量，让其余的变量取正常值，被保留的变量依次取最大值(max)、略微小于最大值(max－)、正常值(normal)、略微大于最小值(min＋)和最小值(min)，对每一个变量都重复进行。因此，对于一个有 n 个变量的程序，若采用边界值分析方法，会生成 $4n+1$ 个测试用例，请初学者务必认真思考原因。

边界值的获取及生成测试用例的步骤如下：

① 使用一元划分方法划分输入域。此时，有多少个输入变量就形成多少种划分。

② 为每种划分确定边界，也可利用输入变量之间的特定关系确定边界。

③ 设计测试用例，确保每个边界至少出现在一个测试输入数据中。

3.2.2 健壮性测试

健壮性测试是边界值分析方法的一种扩展，即测试输入数据除了选取上面提及的最大值(max)、略微小于最大值(max－)、正常值(normal)、略微大于最小值(min＋)和最小值(min)共 5 种边界值之外，还要考虑选取 2 种超出边界范围的值。即比最小值(min)还要略小一些，比最大值(max)还要略大一些。因此，对于一个含有 n 个输入变量的程序，采用健壮性测试方法设计测试用例，同样保留一个变量，让其余变量取正常值，这个保留的变量依次取 7 个值，即分别为最大值(max)、略微小于最大值(max－)、正常值(normal)、略微大于最小值(min＋)、最小值(min)，以及比最小值(min)还要略小一点的值 min－，比最大值(max)还要略大一点的值 max＋，每个变量重复进行，所以健壮性测试方法将产生 $6n+1$ 个测试用例。

3.2.3 运用边界值分析方法设计测试用例举例

为了方便初学者更好地学习与理解，这里选取的仍然是 3.1.2 节中的案例 2——三角形形状判定问题，只是要求运用边界值分析方法及健壮性测试方法来设计相应的测试用例。

1. 三角形形状判定(边界值分析方法)

假设与 3.1.2 节案例 2 中对三角形 a(1≤a≤50)、b(1≤a≤50)、c(1≤a≤50)三边的长度要求一样，运用边界值分析方法设计测试用例，判断由这三条边所构成的三角形类型是等边三角形(三边均相等)、等腰三角形(只要任意两边相等)、一般三角形或非三角形(不能构成一个三角形)。

分析：

① 运用边界值分析方法，三角形三边的边长将选取其所允许的最大值(max)、略微小于最大值(max－)、正常值(normal)、略微大于最小值(min＋)和最小值(min)共 5 种边界值。因为 a、b、c 三边的取值范围均一样，因此可以对应选取数值 50、49、25、2、1，代表所允许的最大值(max)、略微小于最大值(max－)、正常值(normal)、略微大于最小值(min＋)和最小值(min)来设计相应的测试用例。

② 对于三角形问题，由于有三条边 a、b、c 作为输入变量，即测试输入变量的个数为 3，即产生 13(4×3+1)个测试用例。

所以，运用边界值分析方法设计三角形形状判定问题的测试用例如表 3.13 所示。

表 3.13　边界值分析方法的测试用例表

测试用例	输入数据			预期结果	取值说明
	a	b	c		
Test Case1	25	25	1	等腰三角形	a、b 取正常值(normal)，c 取最小值(min)
Test Case2	25	25	2	等腰三角形	a、b 取正常值(normal)，c 取略微大于最小值(min+)
Test Case3	25	25	25	等边三角形	a、b、c 均取正常值(normal)
Test Case4	25	25	49	等腰三角形	a、b 取正常值(normal)，c 取略微小于最大值(max−)
Test Case5	25	25	50	非三角形	a、b 取正常值(normal)，c 取最大值(max)
Test Case6	25	1	25	等腰三角形	a、c 取正常值(normal)，b 取最小值(min)
Test Case7	25	2	25	等腰三角形	a、c 取正常值(normal)，b 取略微大于最小值(min+)
Test Case8	25	49	25	等腰三角形	a、c 取正常值(normal)，b 取略微小于最大值(max−)
Test Case9	25	50	25	非三角形	a、c 取正常值(normal)，b 取最大值(max)
Test Case10	1	25	25	等腰三角形	b、c 取正常值(normal)，a 取最小值(min)
Test Case11	2	25	25	等腰三角形	b、c 取正常值(normal)，a 取略微大于最小值(min+)
Test Case12	49	25	25	等腰三角形	b、c 取正常值(normal)，a 取略微小于最大值(max−)
Test Case13	50	25	25	非三角形	b、c 取正常值(normal)，a 取最大值(max)

请读者务必认真思考本例中 Test Case5、Test Case9 与 Test Case13 的三种情况。这三个测试用例的预期输出结果应该是“非三角形”，即在这三个测试用例中，尽管 a、b、c 三条边的取值无法构成一个三角形，但就 a、b、c 三条边的取值数值而言，却均在其限制的数值范围内。也就是说，a、b、c 三条边的取值都是合法的(只是不能构成一个三角形而已)。

2. 三角形形状判定(健壮性测试方法)

在 3.3.3 小节案例 1 的基础上，运用健壮性测试方法设计三角形形状判定问题的测试用例。

分析：前面已经说过，对于三条边 a、b、c 作为输入变量(测试输入变量的个数为 3)，

运用健壮性测试方法即产生 19(6×3+1)个测试用例。在这 19 个测试用例中,有 13 个可以与上面运用边界值分析方法所生成的测试用例等同(本例中不予列出)。多出的 6 个测试用例则是 a、b、c 中的任意两个变量取正常值(这两个正常值可以相同,也可以不同),而另一个变量分别选取比其最小值(min)还要略小一点的值:min−,以及比其最大值(max)还要略大一点的值:max+。根据排列组合,a、b、c 三个变量则总共产生 6 种取值可能性,即再生成 6 个测试用例,分别用编号 Test Case14 至 Test Case19 来表示。表 3.14 为完整的健壮性测试方法的测试用例表。

表 3.14 健壮性测试方法的测试用例表

测试用例	输入数据			预期结果	取值说明
	a	b	c		
Test Case1	25	25	1	等腰三角形	a、b 取正常值(normal),c 取最小值(min)
Test Case2	25	25	2	等腰三角形	a、b 取正常值(normal),c 取略微大于最小值(min+)
Test Case3	25	25	25	等边三角形	a、b、c 均取正常值(normal)
Test Case4	25	25	49	等腰三角形	a、b 取正常值(normal),c 取略微小于最大值(max−)
Test Case5	25	25	50	非三角形	a、b 取正常值(normal),c 取最大值(max)
Test Case6	25	1	25	等腰三角形	a、c 取正常值(normal),b 取最小值(min)
Test Case7	25	2	25	等腰三角形	a、c 取正常值(normal),b 取略微大于最小值(min+)
Test Case8	25	49	25	等腰三角形	a、c 取正常值(normal),b 取略微小于最大值(max−)
Test Case9	25	50	25	非三角形	a、c 取正常值(normal),b 取最大值(max)
Test Case10	1	25	25	等腰三角形	b、c 取正常值(normal),a 取最小值(min)
Test Case11	2	25	25	等腰三角形	b、c 取正常值(normal),a 取略微大于最小值(min+)
Test Case12	49	25	25	等腰三角形	b、c 取正常值(normal),a 取略微小于最大值(max−)
Test Case13	50	25	25	非三角形	b、c 取正常值(normal),a 取最大值(max)
Test Case14	25	25	0.5	输入非法	a、b 取正常值(normal),c 取比其最小值(min)还要略小一点的值:−min
Test Case15	25	25	51	输入非法	a、b 取正常值(normal),c 取比其最大值(max)还要略大一点的值:max+
Test Case16	25	0.5	25	输入非法	a、c 取正常值(normal),b 取比其最小值(min)还要略小一点的值:−min

续表

测试用例	输入数据			预期结果	取 值 说 明
	a	b	c		
Test Case17	25	51	25	输入非法	a、c 取正常值(normal)，b 取比其最大值(max)还要略大一点的值：max+
Test Case18	0.5	24	25	输入非法	b、c 取正常值(normal)，a 取比其最小值(min)还要略小一点的值：min−
Test Case19	51	23	25	输入非法	b、c 取正常值(normal)，a 取比其最大值(max)还要略大一点的值：max+

注：在本例的 Test Case18 与 Test Case19 中，b、c 取的都是正常值，但二者数值不相同，这也是可以的。

3.3 决　策　表

前面介绍的等价类划分与边界值分析的黑盒测试方法主要着眼于对具体输入数据(可以是 1 个，也可以是多个)的测试。但是对于很多实际应用程序，除了数据外，还需要验证程序实现的逻辑流程，把复杂的逻辑关系与多种输入条件组合表达得明确、具体，体现出严谨的业务逻辑关系。介绍决策表方法前先举个实例。

假设某电子书籍阅读软件的用户界面在读者阅读电子书期间会提示 3 个问题，要求读者同时对这 3 个问题做“是”与“否”的选择，然后软件界面会根据读者的选择情况弹出相应的建议结果。

问题 1：你现在觉得眼睛疲劳吗？

问题 2：你对书中的内容感兴趣吗？

问题 3：你对所阅读的内容有困惑吗？

结果 1：请继续往后阅读！

结果 2：请返回本章从头阅读！

结果 3：请停止阅读，休息一会儿！

软件功能描述简介：

① 如果读者觉得眼睛疲劳，无论其对书中的内容是否感兴趣，无论对所阅读的内容是否有困惑，软件都需要显示“请停止阅读，休息一会儿！”的结果。

② 在读者觉得眼睛不疲劳的前提下，如果其对书中的内容不感兴趣，无论其对所阅读的内容是否有困惑，软件都显示“请停止阅读，休息一会儿！”的结果。

③ 在读者觉得眼睛不疲劳的前提下，如果对书中的内容感兴趣，但是却对所阅读的内容有困惑，软件显示“请返回本章从头阅读！”的结果；如果对书中的内容感兴趣，并对所阅读的内容也没有困惑，软件则显示“请继续往后阅读！”的结果。

从软件的功能描述并结合常识不难发现，3 个问题之间及 3 个结果之间是存在某些隐性的联系及制约关系的。例如：

如果读者对问题 1 选择“是”,即读者已感到视疲劳,不能再进行阅读。也就是说,无论对问题 2 和问题 3 的选择为“是”还是“否”,其结果都已经不会再受问题 2 及问题 3 选择的影响。同理,若读者对问题 1 和问题 2 同时选择“否”,即读者不觉得眼睛疲劳,并且对书中内容已不感兴趣,尽管从逻辑上来说没有问题,但根据常识得知,此时问题 3 已无再讨论的必要,即无论对所阅读的内容是否有困惑,都不会影响结果(请停止阅读,休息一会儿!)。

同样,问题 1 与结果 1 之间也存在约束关系,即当对问题 1 选择“是”,结果 1 的显示结果必须是“否”,否则会产生逻辑错误。当然,结果 1 与结果 3 之间也是一种制约关系,即两种结果不能同时显示,否则会产生矛盾(到底是建议读者继续往后阅读?还是停止阅读?)。

最后,结果 2 与结果 3 之间也是相互制约关系,即不允许两个结果同时出现,只允许出现其中一个结果。

综上所述,测试人员需要认真地从软件的功能描述出发,并充分结合认知常识,认真分析问题与问题之间、问题与结果之间以及结果与结果之间存在的正确的、严谨的业务逻辑关系。

在本案例中,每一个问题都有“是”与“否”两种情况供选择输入,那么 3 个问题一共会有 8(2^3)种可能性。同样,可以用“是”与“否”来表示 3 种结果显示与否。为了方便读者理解,表 3.15 给出了这 3 个问题的 8 种输入可能性(情况)及其对应结果,表中的黑色阴影方框表示对“是”或“否”的选择情况没有意义,也不会影响程序的功能。

表 3.15 人机交互功能“问题-结果”表

选项		1	2	3	4	5	6	7	8
问题	问题 1:你现在觉得眼睛疲劳吗?	是	是	是	是	否	否	否	否
	问题 2:你对书中的内容感兴趣吗?	■	■	■	■	是	是	否	否
	问题 3:你对所阅读的内容有困惑吗?	■	■	■	■	是	否	■	■
显示结果	结果 1:请继续往后阅读!	否	否	否	否	否	是	否	否
	结果 2:请返回本章从头阅读!	否	否	否	否	是	否	否	否
	结果 3:请停止阅读,休息一会儿!	是	是	是	是	否	否	是	是

实际上,可以在表 3.15 的基础上对不影响输入问题取值的黑色阴影方框部分进行有效合并,即合并为 4 种输入可能性(情况),如表 3.16 所示。这也成为测试人员设计测试用例的依据。

表 3.16 合并后的人机交互功能“问题-结果”表

选项		1-4	5	6	7-8
问题	问题 1:你现在觉得眼睛疲劳吗?	是	否	否	否
	问题 2:你对书中的内容感兴趣吗?	■	是	是	否
	问题 3:你对所阅读的内容有困惑吗?	■	是	否	■

续表

选　　项		1-4	5	6	7-8
显示结果	结果 1：请继续往后阅读！	否	否	是	否
	结果 2：请返回本章从头阅读！	否	是	否	否
	结果 3：请停止阅读，休息一会儿！	是	否	否	是

3.3.1　决策表及其组成元素

1. 决策表的概念

决策表又称为判定表，是一种分析和描述多种逻辑条件下执行不同操作情况的测试技术。决策表能够按照各种可能的情况将复杂的问题全部列举出来，以此设计出完整的测试用例集合。在黑盒测试方法中，决策表适合测试多个输入条件（变量）之间存在逻辑关系，以及输入与输出之间存在某种因果关系的应用程序。所以，运用决策表方法可以把复杂的逻辑关系和多种条件组合情况表达明确，避免一些测试遗漏情况。

2. 决策表的组成要素

决策表由条件桩、动作桩、条件项和动作项 4 部分要素以及决策表的规则组成，如图 3.3 所示。

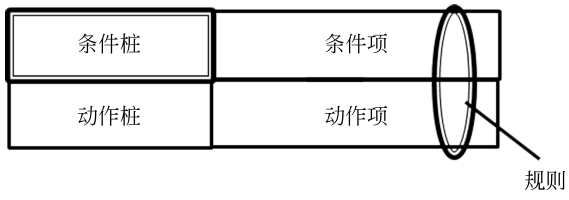

图 3.3　决策表的组成要素

① 条件桩。列出测试问题的所有输入条件。通常，对于多个条件，不同输入条件的排列次序无关紧要。

② 条件项。每一个条件的取值。在决策表中，每一个条件的取值只允许有两种情况，即“是”与“否”。有些测试书籍则用 Y（真值）与 N（假值）表示，或者使用逻辑值 1 与 0 表示。初学者可以把“是”理解成“条件发生”，“否”理解成“条件不发生”。

③ 动作桩。列出测试问题可能产生的操作（输出）结果。同样，对于多个操作结果，对这些操作结果的排列顺序没有约束。

④ 动作项。列出在条件项的各种取值情况下应该采取的动作。与条件项的结果一样，每一个动作的取值也只允许有两种情况，即“是”与“否”。初学者可以把“是”理解成“结果发生（显示）”，“否”理解成“结果不发生（不显示）”。

在决策表中，规则是指任何一组条件组合的特定取值及其要执行的相应操作。在表 3.15 中所贯穿的问题项和结果项的一列就可以看成是一条规则。决策表中列出多少组

输入条件取值，就有多少条规则，即条件项的列数。对于很多实际测试问题，规则往往取决于被测程序的用户需求。

注：决策表中如果有两条或多条规则具有完全相同的动作或产生了完全相同的显示结果，并且其条件项之间存在极为相似的关系，则具有相同动作的规则可以进行合并。

3.3.2 决策表的建立步骤

这里再通过一个简单的测试案例详细介绍构造决策表的基本步骤。

假设某检修软件对 A 机器的检修要求如下：如果 A 机器的使用功率大于 100kW，或已经使用了 15 年以上，或中途出现过故障，软件界面显示“需要仔细检修”的建议。当且仅当上述条件都不满足时，软件界面显示允许 A 机器“继续使用”的建议。

(1) 列出所有的条件桩和动作桩。

通过对案例的分析，列出相应的条件桩，并予以编号：

C1：功率大于 100kW。

C2：已经使用了 15 年以上。

C3：中途发生过故障。

列出相应的动作桩，并予以编号：

A1：需要仔细检修。

A2：继续使用。

(2) 确定规则的个数。

因为条件桩(输入条件)的个数为 3，则一共有 8(2^3)种规则。

(3) 填入条件项与动作项，得到初始决策表，如表 3.17 所示。

表 3.17 人机交互功能决策表

选　　项		1	2	3	4	5	6	7	8
条件	C1：使用功率大于 100kW	是	是	是	是	否	否	否	否
	C2：已经使用了 15 年以上	是	是	否	否	是	是	否	否
	C3：中途发生过故障	是	否	是	否	是	否	是	否
动作	A1：需要仔细检修	是	是	是	是	是	是	是	否
	A2：继续使用	否	否	否	否	否	否	否	是

(4) 合并相似规则，化简决策表，如表 3.18 所示。

注：用 Y 与 N 表示“是”与“否”，那些不影响输入问题取值的条件用黑色阴影方框表示。

表 3.18 人机交互功能决策表

选　　项		1-4	5-6	7	8
条件	C1：使用功率大于 100kW	Y	N	N	N
	C2：已经使用了 15 年以上	■	Y	N	N
	C3：中途发生过故障	■	■	Y	N

续表

选　　项		1-4	5-6	7	8
动作	A1：需要仔细检修	Y	Y	Y	N
	A2：继续使用	N	N	N	Y

通过表 3.18 得知，化简后的决策表规则只有 4 个，所以可以据此设计出 4 个测试用例，如表 3.19 所示。

表 3.19　人机交互功能决策表

测 试 用 例	测 试 输 入	预 期 结 果
Test Case1	使用功率大于 100kW	需要仔细检修
Test Case2	使用功率不大于 100kW，但是已经使用了 15 年以上	需要仔细检修
Test Case3	使用功率不大于 100kW，也没有使用 15 年以上，但是中途发生过故障	需要仔细检修
Test Case4	使用功率不大于 100kW，也没有使用 15 年以上，中途也没有发生过故障	继续使用

3.3.3　运用决策表方法设计测试用例举例

这里仍然选取 3.1.2 节中的案例 2——三角形形状判定问题，三个整数 a(1≤a≤50)、b(1≤a≤50)和 c(1≤a≤50)分别作为一个三角形的三条边，只是要求运用决策表方法来设计相应的测试用例，以方便初学者理解。

(1) 列出所有的条件桩和动作桩。

通过对三角形形状(一般三角形/等腰三角形/等边三角形/非三角形)判定条件的分析列出相应的条件桩，并予以编号：

C1：a、b、c 能否构成一个三角形？

C2：a＝b？

C3：b＝c？

C4：a＝c？

列出相应的动作桩，并予以编号：

A1：一般三角形

A2：等腰三角形

A3：等边三角形

A4：非三角形

A5：不存在(不可能)

(2) 因为条件桩的个数是 4，所以规则的个数是 16(2^4)。

(3) 依次填入动作项，构造初始的决策表，如表 3.20 所示。

表 3.20 人机交互功能“问题-结果”表

规 则		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
条件	C1：a、b、c 能否构成一个三角形？	是	是	是	是	是	是	是	是	否	否	否	否	否	否	否	否
	C2：a＝b？	是	是	是	是	否	否	否	否	是	是	是	是	否	否	否	否
	C3：b＝c？	是	是	否	否	是	是	否	否	是	是	否	否	是	是	否	否
	C4：a＝c？	是	否	是	否	是	否	是	否	是	否	是	否	是	否	是	否
结果	A1：一般三角形	■	■	■	■	■	■	■	是	■	■	■	■	■	■	■	■
	A2：等腰三角形	■	■	■	是	■	是	是	■	■	■	■	■	■	■	■	■
	A3：等边三角形	是	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
	A4：非三角形	■	■	■	■	■	■	■	■	是	是	是	是	是	是	是	是
	A5：不存在	■	是	是	■	是	■	■	■	■	■	■	■	■	■	■	■

注：在本例中，由于把等边三角形与等腰三角形看作两种不同类别的三角形，所以列出的 5 个动作桩（三角形形状的判定结果）是互斥的。为了增强阅读效果，决策表中（判定）结果为“否”的选项全部用黑色阴影方框表示。

（4）合并初始表中的相似规则，化简决策表，把条件 C1（a、b、c 能否构成一个三角形？）为“否”的规则（9～16 列）合并为 1 列，即化简后的决策表只有 9 列，如表 3.21 所示。

注：同样，对取值为“否”的选项以及那些不影响输入问题取值的条件用黑色阴影方框表示。

表 3.21 人机交互功能“问题-结果”表

规 则		1	2	3	4	5	6	7	8	9～16
条件	C1：a、b、c 能否构成一个三角形？	是	是	是	是	是	是	是	是	否
	C2：a＝b？	是	是	是	是	否	否	否	否	■
	C3：b＝c？	是	是	否	否	是	是	否	否	■
	C4：a＝c？	是	否	是	否	是	否	是	否	■
结果	A1：一般三角形	■	■	■	■	■	■	■	是	■
	A2：等腰三角形	■	■	■	是	■	是	是	■	■
	A3：等边三角形	是	■	■	■	■	■	■	■	■
	A4：非三角形	■	■	■	■	■	■	■	■	是
	A5：不存在	■	是	是	■	是	■	■	■	■

（5）根据决策表 3.21 中的每一条规则设计一个测试用例。但是规则 2、规则 3 与规则 5 实际是无法满足输入数据要求的（输入不存在），无法生成对应的测试用例。所以测试用例列表中仅含有 6 个测试用例，如表 3.22 所示。