【黑盒技术】

一、 因果图方法的思路是：从用自然语言书写的程序规格说明描述中找出因（输入条件）和果 （输出或程序状态的改变），通过因果图转换为判定表。

分析中国象棋中走马的实际情况 （下面未注明的均指的是对马的说明），马走日字型（邻近交叉点无棋子），遇到对方棋子可以吃掉，遇到本方棋子不能落到该位置。

[问题1]（3分）

应用中可能有多种输入条件，在什么情况下可采用因果图法设计测试用例？

[问题2]（（4.5分）

根据上述说明，利用因果图法，下面列出走棋出现的情况和结果，找出哪些是正确的输 入条件，哪些是正确的输出结果，请把相应的字母编号填入表中。

A.落点在棋盘上；

B.落点与起点构成日字； C.移动棋子；

D.落点处为对方棋子； E.落点处为自己方棋子；

F.移动棋子，并除去对方棋子；

G.落点方向的邻近交叉点无棋子；

H.不移动棋子；

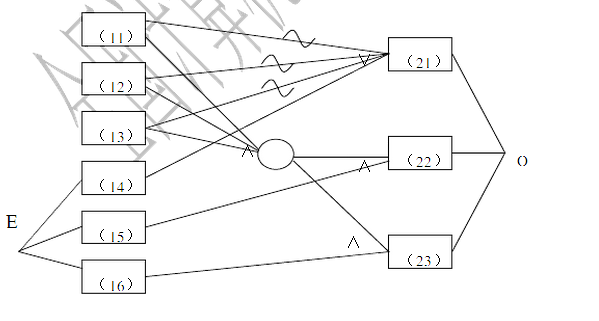
 I.落点处无棋子。



[问题3]（4.5分）

下图画出中国象棋中走马的因果图，请把问题2中列出的输入条件和输出结果的字母编号填入到空白框中相应的位置。

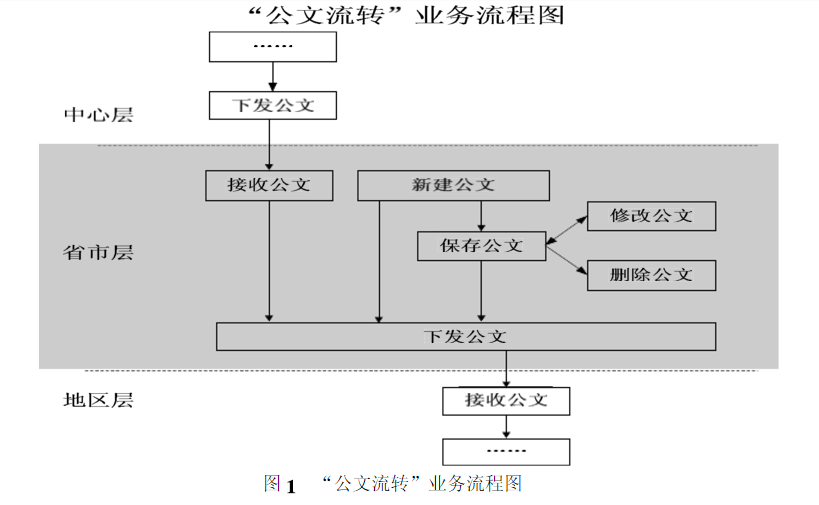
输入条件 输出条件

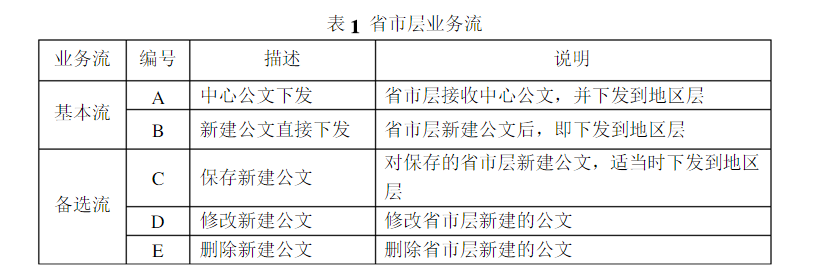


二、场景法是黑盒测试中重要的测试用例设计方法。目前多数软件系统都是用事件触发来控制业务流程，事件触发时的情景便形成了场景，场景的不同触发顺序构成用例。场景法通过场景描述业务流程（包括基本流（基本流程）和备选流（分支流程）），设计用例遍历软件系统功能，验证其正确性。

图 1 描述了简化的中心层、省市层、地区层三级的“公文流转”业务流程，表 1 描述了省市层（图 1 阴影部分）业务的基本流和备选流。

公文的状态包括：己下发、未下发、已接收、未接收。



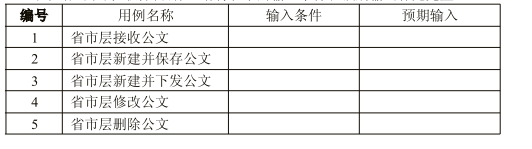


【问题 1】（5 分）

用表 1 中表述的基本流和备选流，使用场景法设计测试场景。基本流和备选流用表 1中对应的字母编号表示。

【问题 2】（10 分）

下表给出了测试用例名称，请将表中的输入条件和预期输出补充完整。



三、场景法是黑盒测试中重要的测试用例设计方法。目前多数软件系统都是用事件触发来控制业务流程，事件触发时的情景便形成了场景，场景的不同触发顺序构成用例。场景法通过场景描述业务流程（包括基本流(基本流程)和备选流(分支流程)），设计用例遍历软件系统功能，验证其正确性。

下面是对网上银行支付交易系统的基本流和备选流的描述：

基本流A：

|  |  |  |
| --- | --- | --- |
| 步骤 | 步骤名称 | 步骤描述 |
| A1 | 网上订购商品 | 用户登录网站，订购所需商品，点击网上银行支付 |
| A2 | 输入银行卡信息 | 输入银行卡号和密码 |
| A3 | 校验银行卡信息 | 系统对银行卡号和密码 |
| A4 | 金额验证1 | 系统确认订单金额不大于卡内余额 |
| A5 | 金额验证2 | 系统确认订单金额不大于银行卡网上可支付额度 |
| A6 | 银行卡扣款 | 支付成功，系统从银行卡中扣除相应金额，返回订单号 |

备选流

|  |  |  |
| --- | --- | --- |
| 编号 | 名称 | 备选流描述 |
| B | 密码不正确 | 在基本流A3步骤中，密码不正确（且密码输入尚未超过三次），重新加入基本流A2；否则退出基本流 |
| C | 银行卡内余额不足 | 在基本流A4步骤中，系统判断银行卡内余额不足以支付订单，退出基本流 |
| D | 银行卡网上可支付额度不够 | 在基本流A5步骤中，系统判断银行卡网上可支付额度小于订单金额，退出基本流 |

注：假定输入的银行卡号是正确的；不考虑备选流内循环情况。

问题1（10分）：

使用场景法设计测试用例，指出所涉及到的基本流和备选流。基本流用字母A表示，备选流用题干中描述对应编号表示。

问题2（10分）：

请针对问题1设计的测试用例，依次将银行卡号、初次输入密码、最终输入密码、卡内余额、银行卡可支付额度等信息填入下述测试用例表中。表中行代表各个测试用例，列代表测试用例的输入值，用V表示有效数据元素，I表示无效数据元素，n/a表示不适用，例如C01表示“成功支付” 用例。

测试用例表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 测试用例ID号 | 银行卡号 | 初次输入密码 | 最终输入密码 | 卡内余额 | 银行卡可支付额度 |
| C01 | V | V | N/a | V | V |
| C02 |  |  |  |  |  |
| C03 |  |  |  |  |  |
| C04 |  |  |  |  |  |
| C05 |  |  |  |  |  |
| C06 |  |  |  |  |  |

问题3（10分）：

在上述系统中，假设银行卡号只能输入0~9的数字，请参考下表，给出用边界值法检查卡号字符合法性的关键测试数据（字符或ASCII值）。

|  |  |  |  |
| --- | --- | --- | --- |
| 字符 | ASCII值 | 字符 | ASCII值 |
| NULL | 0 | B | 66 |
| Space | 32 | Y | 89 |
| / | 47 | Z | 90 |
| 0 | 48 | [ | 91 |
| 1 | 49 | ‘ | 96 |
| 2 | 50 | a | 97 |
| 9 | 57 | b | 98 |
| ： | 58 | y | 121 |
| @ | 64 | z | 122 |
| A | 65 | { | 123 |

【白盒技术】

一、使用基本路径法设计出的测试用例能够保证程序的每一条可执行语句在测试过程中至少执行一次。以下代码由C语言书写，请按要求回答问题。

Int IsLeap(int year)

{

if (year % 4 == 0)

{

　if (year % 100 == 0)

　{

　　if ( year % 400 == 0)

　　　leap = 1;

　　else

　　　leap = 0;

　　}

　else

　　 leap = 1;

　}

else

　leap = 0;

return leap;

}

【问题1】（3分）

　　请画出以上代码的控制流图

【问题2】（4分）

　　请计算上述控制流图的圈复杂度V（G）（独立线性路径数）

【问题3】（8分）

假设输入的取值范围是1000 < year < 2001,请使用基本路径测试法为变量year设计测试用例，使其满足基本路径覆盖的要求。

二、阅读下列说明，回答问题1至问题3，将解答填入答题纸的对应栏内。

以下代码由C语言书写，在输入三个整数后，能够输出最大数和最小数。

int main( void )

{

int a, b, c, max, min;

printf( "input thr numbers: " );

scanf( "%d%d%d", &a, &b, &c );

if( a > b ) /\*判断 1\*/

{

max = a;

min = b;

}

els

{

max = b;

min = a;

}

if( max < c ) /\*判断 2\*/

max = c;

els if( min > c ) /\*判断 3\*/

min = c;

printf( "max=%d\nmin=%d", max, min );

return 0;

}

[问题1] （3 分）

请画出该程序段的控制流图，并分析该控制流图中有多少条线性独立路径 （V(G)的值）。

[问题2] （2 分）

为了用分支 （判定）覆盖方法测试该程序，需要设计测试用例，使其能对该程序中的每个判断语句的各种分支情况全部进行过测试。

对两组输入数据 （1）a=3，b=5，c=7；（2）a=4，b=6，c=5，请分别写出程序中各个判断语句的执行结果 （以T表示真，以F表示假），填入答题纸的相应栏中。

[问题3] （5 分）

上述两组测试数据是否能实现该程序的分支（判定）覆盖？如果能，请说明理由。如果不能，请再增设一组输入数据，使其能实现分支（判定）覆盖。

三、逻辑覆盖法是设计白盒测试用例的主要方法之一，它是通过对程序逻辑结构的遍历实现程序的覆盖。针对以下由 C 语言编写的程序，按要求回答问题。

getit( int m）

｛

int i ，k;

k＝sqrt( m）；

for（ i ＝2； i ＜＝k; i++）

if （m%i==0）break;

if （ i ＞＝k＋1）

printf(“%d is aselected number\n”， m）：

else

printf(“%d is not a selected number\n”， m）；

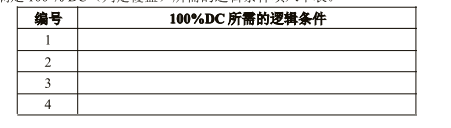
｝

【问题 1】（3 分）

请找出程序中所有的逻辑判断子语句。

【问题 2】（4 分）

请将满足 100 % DC（判定覆盖）所需的逻辑条件填入下表。



【问题 3】 (9 分）

请画出上述程序的控制流图，并计算其控制流图的环路复杂度 V (G)。假设函数 getit的参数 m 取值范围是 150＜m＜160，请使用基本路径测试法设计测试用例，将参数 m 的取值填入下表，使之满足基本路径覆盖要求。

