



东南大学国家示范性软件学院

College of Software Engineering
Southeast University

软件测试基础与实践

实验报告

实验名称： 白盒测试实验一

实验地点： 计算机楼 268

实验日期： 2018.10.25

学生姓名： 杨昱昊

学生学号： 71116216

东南大学 软件学院 制



一、实验目的

- (1) 巩固基于控制流白盒测试知识,对于给定的待测程序,能熟练应用基本控制流覆盖方法设计测试用例;
- (2) 通过绘制程序控制流程图,实现对程序源代码的逻辑描述;
- (3) 掌握逻辑短路对测试的影响;
- (4) 培养严谨和系统的测试精神,学习测试用例的设计和分析。

二、实验内容

(一) 题目 1: 控制流测试技术实验

1. 题目内容

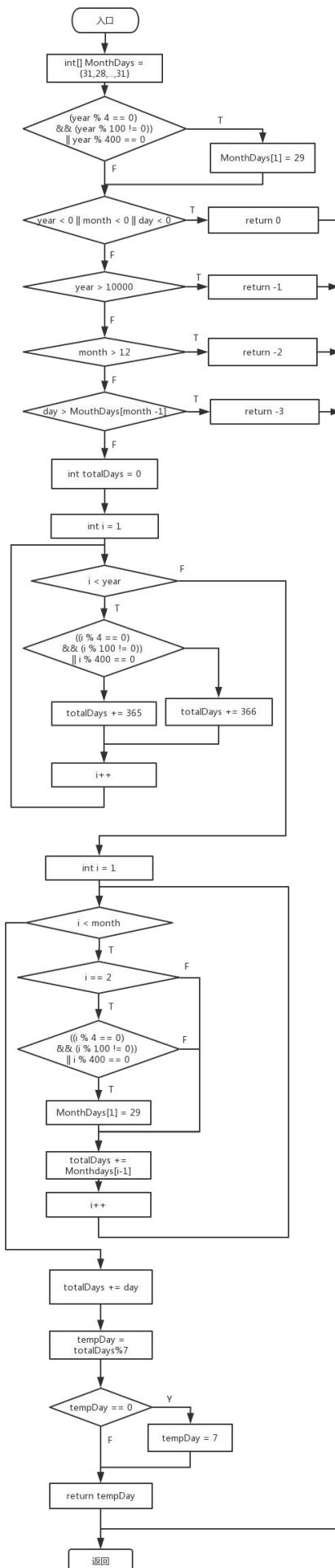
运用基于控制流的动态白盒测试方法,对 WeekA 程序中的方法 getWeekday()进行测试。设计测试用例时,尽可能设计最少的测试用例数,同时保证每种覆盖方法的覆盖率尽可能达到 100%。

2. 题目解答

- (1) 给出 getWeekday()的程序流程图 (请放大查看详情)



1	int[] MonthDays = { 31, 28, 31, 30, 31, 30, 31, 31, 31, 30, 31, 30, 31 };
2	((year % 4 == 0) && (year % 100 != 0)) year % 400 == 0
3	MonthDays[1] = 29;
4	year < 0 month < 0 day < 0
5	return 0;
6	year > 10000
7	return -1
8	month > 12
9	return -2
10	day > MonthDays[month - 1]
11	return -3
12	int totalDays = 0
13	int i = 1
14	i < year
15	((i % 4 == 0) && (i % 100 != 0)) i % 400 == 0
16	totalDays += 366
17	totalDays += 365
18	i++
19	int i = 1
20	i < month
21	i == 2
22	((year % 4 == 0) && (year % 100 != 0)) year % 400 == 0
23	MonthDays[1] = 29
24	totalDays += MonthDays[i - 1]
25	i++
26	totalDays += day
27	int tempDay = totalDays % 7
28	tempDay == 0
29	tempDay = 7
30	return tempDay





(2) 分别以语句覆盖和判定覆盖方法设计测试用例,并写出每个测试用例的执行路径。

a. 语句覆盖

编号	执行条件	输入	期望输出	实际输出	执行路径
		yy/mm/dd			
P1	语句覆盖	1999/2/1	星期 0	星期 0	1-2-4-5
P2	语句覆盖	19999/3/5	星期-1	星期-1	1-2-4-6-7
P3	语句覆盖	1999/14/5	星期-2	星期-2	1-2-4-6-8-9
P4	语句覆盖	101/12/40	星期-3	星期-3	1-2-4-6-8-10-11
P5	语句覆盖	4/2/29	星期 7	星期 7	1-2-3-4-6-8-10-12-13-14-15-1 6-17-18-19-20-21-22-23-24-2 5-26-27-28-29-30

b. 判定覆盖

编号	执行 条件	输入	期望 输出	实际 输出	条件判断										执行路径
		yy/mm/dd			2	4	6	8	10	14	15	20	22	28	
P1	判定 覆盖	1999/2/-1	星期 0	星期 F T											1-2-4-5
P2	判定 覆盖	19999/3/5	星期 -1	星期 F F T											1-2-4-6-7
P3	判定 覆盖	1999/14/5	星期 -2	星期 F F F T											1-2-4-6-8-9
P4	判定 覆盖	101/12/40	星期 -3	星期 F F F F T											1-2-4-6-8-10-11
P5	判定 覆盖	4/2/29	星期 7	星期 F F F F F T/ F T/F T/F T/F T											1-2-3-4-6-8-10-12-13-14-15-16-17-18-19-20-21 -22-23-24-25-26-27-28-29-30
P6	判定 覆盖	4/2/30	星期 1	星期 F F F F F T/ F T/F T/F T/F F											1-2-3-4-6-8-10-12-13-14-15-16-17-18-19-20-21 -22-23-24-25-26-27-28-30

(3) 自行写一个小程序,验证当判定中包含多个条件时,条件短路对控制流测试的影响。

(4) 分别以条件覆盖、判定条件覆盖和条件组合覆盖方法设计测试用例,并写出每个测试用例的执行路径。



a. 条件覆盖

编号	执行条件	输入		期望输出	实际输出	执行路径	
		yy/mm/dd					
P1	条件覆盖	-1/12/4		星期 0	星期 0	1-2-4-5	
P2	条件覆盖	5/-2/5		星期 0	星期 0	1-2-4-5	
P3	条件覆盖	1999/2/-1		星期 0	星期 0	1-2-4-5	
P4	条件覆盖	19999/3/5		星期-1	星期-1	1-2-4-6-7	
P5	条件覆盖	1999/14/5		星期-2	星期-2	1-2-4-6-8-9	
P6	条件覆盖	101/12/40		星期-3	星期-3	1-2-4-6-8-10-11	
P7	条件覆盖	5/2/25		星期 5	星期 5	1-2-3-4-6-8-10-12-13-14-15-1 6-17-18-19-20-21-22-23-24-2 5-26-27-28-30	
P8	条件覆盖	400/2/6		星期 7	星期 7	1-2-3-4-6-8-10-12-13-14-15-1 6-17-18-19-20-21-22-23-24-2 5-26-27-28-29-30	

编号	输入	原子条件及判定															
		yy/mm/dd	2		4		6	8	10	14	15			20	22		28
			a	b	c	d	e	f			g	h	i		a	b	c
P1	-1/12/4	F	T	F	T												
P2	5/-2/5	F	T	F	F	T											
P3	1999/2/-1	F	T	F	F	F	T										
P4	19999/3/5	F	T	F	F	F	F	F	T								
P5	1999/14/5	F	T	F	F	F	F	F	T								
P6	101/12/40	F	T	F	F	F	F	F	F	T							
P7	5/2/25	F	T	F	F	F	F	F	F	T/F	T/F	T	F	T/F	F	T	F
P8	400/2/6	T	F	T	F	F	F	F	F	T/F	T/F	T/F	T/F	T/F	T	F	T



b. 判定条件覆盖

编号	执行条件	输入	期望输出	实际输出	执行路径
		yy/mm/d d			
P1	判定条件覆盖	-1/12/4	星期 0	星期 0	1-2-4-5
P2	判定条件覆盖	5/-2/5	星期 0	星期 0	1-2-4-5
P3	判定条件覆盖	1999/2/-1	星期 0	星期 0	1-2-4-5
P4	判定条件覆盖	19999/3/5	星期-1	星期-1	1-2-4-6-7
P5	判定条件覆盖	1999/14/5	星期-2	星期-2	1-2-4-6-8-9
P6	判定条件覆盖	101/12/40	星期-3	星期-3	1-2-4-6-8-10-11
P7	判定条件覆盖	5/2/25	星期 5	星期 5	1-2-3-4-6-8-10-12-13-14-15-1 6-17-18-19-20-21-22-23-24-2 5-26-27-28-30
P8	判定条件覆盖	400/2/6	星期 7	星期 7	1-2-3-4-6-8-10-12-13-14-15-1 6-17-18-19-20-21-22-23-24-2 5-26-27-28-29-30

编 号	输入	原子条件及判定																				2							
		2			2		4			4		6		8		10		14		15			15	20	22			2	2
		yy/mm/dd			a	b	c	d	e	f	g	h	i	a	b	c	g	h	i	a	b	c							
		a	b	c	d	e	f	g	h	i	g	h	i	a	b	c	g	h	i	a	b	c							
P1	-1/12/4	F	T	F	F	T			T																				
P2	5/-2/5	F	T	F	F	F	T		T																				
P3	1999/2/-1	F	T	F	F	F	F	T	T																				
P4	19999/3/5	F	T	F	F	F	F	F	F	T																			
P5	1999/14/5	F	T	F	F	F	F	F	F	F	T																		
P6	101/12/40	F	T	F	F	F	F	F	F	F	F	T																	
P7	5/2/25	F	T	F	F	F	F	F	F	F	F	F	T/F	T/F	T	F	T/F	T/F	F	T	F	F	F						
P8	400/2/6	T	F	T	T	F	F	F	F	F	F	F	T/F	T/F	T/F	T/F	T/F	T/F	T	F	T	T	T						



c. 条件组合覆盖

编号	执行条件	输入	期望输出	实际输出	执行路径
		yy/mm/d d			
P1	条件组合覆盖	-1/12/4	星期 0	星期 0	1-2-4-5
P2	条件组合覆盖	5/-2/5	星期 0	星期 0	1-2-4-5
P3	条件组合覆盖	1999/2/-1	星期 0	星期 0	1-2-4-5
P4	条件组合覆盖	19999/3/5	星期-1	星期-1	1-2-4-6-7
P5	条件组合覆盖	1999/14/5	星期-2	星期-2	1-2-4-6-8-9
P6	条件组合覆盖	101/12/40	星期-3	星期-3	1-2-4-6-8-10-11
P7	条件组合覆盖	5/2/25	星期 5	星期 5	1-2-3-4-6-8-10-12-13-14-15-1 6-17-18-19-20-21-22-23-24-2 5-26-27-28-30
P8	条件组合覆盖	400/2/6	星期 7	星期 7	1-2-3-4-6-8-10-12-13-14-15-1 6-17-18-19-20-21-22-23-24-2 5-26-27-28-29-30
P9	条件组合覆盖	2016/2/1	星期 1	星期 1	1-2-3-4-6-8-10-12-13-14-15-1 6-17-18-19-20-21-22-23-24-2 5-26-27-28-30
P10	条件组合覆盖	2100/2/1	星期 1	星期 1	1-2-3-4-6-8-10-12-13-14-15-1 6-17-18-19-20-21-22-23-24-2 5-26-27-28-30



P6	101/12/4 0	F	T	F	F	F	F	F	F	F	T												
P7	5/2/25	F	T	F	F	F	F	F	F	F	F	T/F	T/F	T	F	T/F	T/F	F	T	F	F	F	
P8	400/2/6	T	F	T	T	F	F	F	F	F	F	T/F	T/F	T/F	T/F	T/F	T/F	T	F	T	T	T	
P9	2016/2/1	T	F	F	F	F	F	F	F	F	F	T/F	T/F	T/F	T/F	T/F	T/F	T	F	F	F	F	
P10	2100/2/1	T	T	F	T	F	F	F	F	F	F	T/F	T/F	T/F	T/F	T/F	T/F	T	T	F	T	F	

(5) 给出对程序中循环的测试用例,并说明测试用例设计的理由。

对循环进行测试的测试用例：400/3/1

设计理由：代码中有两个 for 循环，第一个 for 循环为语句 13-18，其中在语句 15 进行了一次判断，对 year 进行了判定，选择 year==400 作为测试用例，可以对这个判断的所有可能条件组合达到覆盖，即 (F,T,F)(T,F,T)(T,F,F)(T,T,-)；第二个 for 循环为语句 19-25，为了测试到 2 月的特殊情况，使语句 23 执行，所以让 month>2。实际上语句 21 和 22 这两次判断是冗余的。

(6) 如果要进一步用路径覆盖准则来测试 getWeekday(),请基于程序流程图计算其中可能的路径条共有多少条?是否包含不可达路径?依照你设计测试用例的速度,完成所有路径的测试需要多少时间?注意:

A: 正确分析程序可能的执行路径;

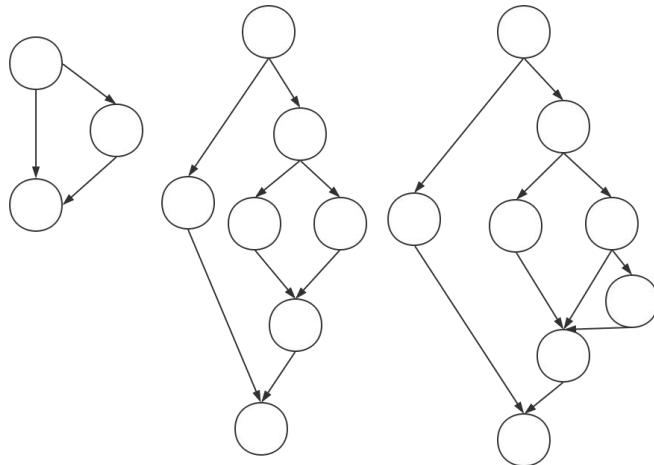
B: 对于涉及循环的路径,统计时可简化为:执行 $N > 1$ 次视为同一条路径,执行 0 次(即跳过)视为另一条路径;

可能路径数量：可能路径的数量取决于执行的判定，在 getWeekday() 中，有 11 个判定，分别在语句 2、4、6、8、10、14、15、20、22、28。其中语句 4、6、8、10 都在进行异常处理过程。共有 8 条路径。将 for 简化为 if 后，下图依次是两次 if，第一个 for 循环，第二个 for 循环的流图。得到正常执行路径有 $2 \times 2 \times 3 \times 4 = 48$ 条。所以共有 56 条路径。

不存在不可达路径。



完成所有路径测试需要约 1 小时。



(7) 给出 MC/DC(修订的判定条件覆盖)方法对下列 2 处语句的测试用例。

```
...
int[] MonthDays = { 31, 28, 31, 30, 31, 30, 31,
31, 30, 31, 30, 31 };
if (((year % 4 == 0) && (year % 100 != 0)) || year % 400 == 0) {
    MonthDays[1] = 29;
}
...
...
```

```
...
if (((i % 4 == 0) && (i % 100 != 0)) || i % 400 == 0) {
    totalDays += 366;
} else {
    totalDays += 365;
}
...
...
```

不考虑逻辑短路

编号	执行条件	输入	期望输出	实际输出	条件			判定
		yy/mm/dd			a	b	c	
P1	MC/DC	3/2/1	星期 6	星期 6	F		F	F
P2	MC/DC	400/2/1	星期 2	星期 2	T	F	T	T
P3	MC/DC	2100/2/1	星期 1	星期 1	T	F	F	F
P4	MC/DC	2016/2/1	星期 1	星期 1	T	T		T



由用例 P2 和 P3, 条件 c 独立

由用例 P3 和 P4, 条件 b 独立

由用例 P1 和 P4, 条件 a 独立

三、实验体会

(1)通过测试,是否发现程序中存在的缺陷?

测试没有得出缺陷。

(2)程序中是否还存在采用动态白盒测试技术未能发现的缺陷?

缺陷是存在的, 比如错误输入 0/0/0 没有被处理, 有冗余的判断语句等。

(3)在存在逻辑短路的情况下,请分析条件覆盖、判定条件覆盖、条件组合覆盖、MC/DC 覆盖的联系和区别。

条件覆盖：设计测试用例使得每一判定语句中的每个逻辑条件的可能去职至少满足一次，只考慮到每个判定语句中的每个表达式，涉及不到各个条件分支。

判定条件覆盖：设计测试用例使得判定中的每个条件的所有可能至少出现一次，并且每个判定本身的判定结果也至少出现一次。没有考虑单个判定对整体结果的影响，无法发现逻辑错误

条件组合覆盖：设计测试用例使得每个判定中条件的各种可能组合都至少出现一次，可以满足条件覆盖和判定条件覆盖的需求，但是当判定条件较多的时候，条件组合值较多

(4)在白盒测试实践中,你觉得哪些步骤是关键的?并谈谈你所体会到的白盒测试过程中的难点。

通过本次实验，我感受到的最关键的部分是流程图的绘制，画出清晰明确的流程图并做好标记，对之后设计测试用例有着莫大的帮助。

难点则在于测试用例的选择，这个过程中我设计了冗余的用例，浪费了一点时间。



东南大学国家示范性软件学院

College of Software Engineering
Southeast University