# 1. Equivalent class Testing

### AccountStatus

Class	Range (accountFactor)	Test Input Value
invalid	0	0
adverse	[1,99]	7
acceptable	[100,499]	300
good	[500,999]	700
excellent	[1000,∞]	1200

# getAgeFactor

Class	Range (age)	Test Input Value
0	[-∞,14],[110,∞]	0, 120
5	[15,19]	17
10	[20,29]	20
20	[30,49],[65,109]	35, 77
50	[40,65]	55

# getBalanceFactor

Class	Range (age)	Test Input Value
0	[-∞,0],[5000, ∞]	-100, 12000
6	[1,99]	17
16	[100,499]	200
30	[500,999]	550
70	[1000,2999]	1550
200	[3000,4999]	3500

### creditStatus:

Class	Range (creditScore)	Test Input Value
Invalid	[-∞,0],[101, ∞]	-100, 120
Adverse, restricted	[1,49]	17
Adverse, default	[1,74]	200
Good, restricted	[50,100]	55
Good, default	[75,100]	80

# productStatus:

Class	Range (productQuantity)	Test Input Value
soldout	0	0
limited	< inventoryThreshold	inventoryThreshold -10
available	>= inventoryThreshold	inventoryThreshold + 10

# orderHandling:

Class	Range (productQuantity)	Test Input Value
accepted	excellent' accountStatus 'good' accountStatus and creditStatus 'adverse' or 'acceptable' accountStatus, 'good' creditStatus and 'available' productStatus.	accountStatus = 'excellent'
pending	acceptable' accountStatus, 'good' creditStatus and 'limited' or 'soldout' productStatus.	accountStatus = acceptable' creditStatus = 'good' productStatus = 'soldout'
underReview	'good' accountStatus and 'adverse' creditStatus 'acceptable' accountStatus, 'adverse' creditStatus and 'available' productStatus	accountStatus = 'good' creditStatus = 'adverse' productStatus = 'available'
rejected	'acceptable' accountStatus, 'adverse' creditStatus and 'limited' or 'soldout' productStatus. 'adverse'	accountStatus = 'adverse' creditStatus = 'good' productStatus = 'soldout'

accountStatus, 'good' creditStatus and 'soldout' productStatus. 'adverse' accountStatus and 'adverse' creditStatus	
--	--

# 2. Boundary Values Testing

### Account Status:

Class (result)	Test Value(s)
invalid	0
poor	1,99 (both not possible – no test case)
fair	100,499 (both not possible)
good	500,999 (both not possible)
Very good	2000 (both not possible)

# getAgeFactor:

Class (result)	Test Value(s)
0	14,110
5	15,19
10	20,29
20	30,39
50	40,64

# getBalanceFactor:

Class (result)	Test Value(s)
0	0,5000

6	1,99
16	100,499
30	500,999
70	1000,2999
200	3000,4999

#### creditStatus:

Class (result)	Test Value(s)
Invalid	-1, 101
Adverse, restricted	0,49
Adverse, default	0,74
Good, restricted	50,100
Good, default	75,100

# productStatus:

Class (result)	Test Value(s)
soldout	0
limited	inventoryThreshold-1
available	inventoryThreshold+1

orderHandling: boundary test not applicable

Blue	= Conditions						
Green	= Actions						
orderHandling:							
	Rule 1	Rule 2	Rule 3	Rule 4	Rule 5	Rule 6	Rule 7
accountStatus	excellent	good	adverse	acceptable	acceptable	acceptable	adverse
creditStatus	-	good	good	good	good	good	good
productStatus	-	-	available	available	limited	soldout	limited
accepted	х	X	Х	Х			
pending					X	X	Х
underReview							
rejected							
	Rule 8	Rule 9	Rule 10	Rule 11	Rule 12	Rule 13	Rule 14
accountStatus	good	acceptable	invalid	acceptable	acceptable	adverse	adverse
creditStatus	adverse	adverse	invalid	adverse	adverse	good	adverse
productStatus	-	available	invalid	limited	soldout	soldout	-
accepted							
pending							
underReview	Х	Х					
rejected			Х	Х	X	X	Х

# Analysis report: Before Code Fixes

PurchaseOrder: 86 total, 41 failed, 45 passe	d	58 ms
	Collapse	Expand
purchaseOrderF19.test.js		58 ms
PurchaseOrder		58 ms
Equivalence Tests		38 ms
getAgeFactor() tests		4 ms
should equal 0	passed	3 ms
should equal 0, case 2	passed	0 ms
should equal 5	passed	1 ms
should equal 10	passed	0 ms
should equal 20	passed	0 ms
should equal 20, case 2	passed	0 ms
should equal 50	passed	0 ms
getBalanceFactor() tests		4 ms
should equal 0	passed	0 ms
should equal 0, case 2	passed	1 ms
should equal 6	passed	0 ms
should equal 16	passed	0 ms
should equal 30	failed	3 ms
should equal 70	passed	0 ms
should equal 200	passed	0 ms
AccountStatus() tests		5 ms
should equal invalid	failed	2 ms
should equal adverse	passed	0 ms
should equal acceptable	failed	1 ms
should equal good	failed	1 ms
should equal excellent	failed	1 ms

28 ms

14 ms

2 ms

#### ■ Boundary Value Tests

#### getAgeFactor() tests

should equal 0	passed	1 ms
should equal 0, case 2	failed	2 ms
should equal 5	passed	0 ms
should equal 5, case2	passed	1 ms
should equal 10	passed	0 ms
should equal 10, case 2	passed	1 ms
should equal 20	passed	1 ms
should equal 20, case 2	passed	0 ms
should equal 20, case 3	failed	2 ms
should equal 20, case 4	failed	2 ms
should equal 50	failed	2 ms
should equal 50, case 2	failed	2 ms

### getBalanceFactor() tests

should equal 0	passed	1 ms
should equal 0, case 2	passed	1 ms
should equal 6	passed	0 ms
should equal 6, case2	passed	0 ms
should equal 16	passed	0 ms
should equal 16, case 2	passed	0 ms
should equal 30	passed	0 ms
should equal 30, case 2	passed	0 ms
should equal 70	passed	0 ms
should equal 70, case 2	passed	0 ms
should equal 200	passed	0 ms
should equal 200, case 2	passed	0 ms

	AccountStatus() tests		8 ms
	should equal invalid	failed	4 ms
	should equal adverse	passed	0 ms
	should equal acceptable	failed	1 ms
	should equal good	failed	2 ms
	should equal excellent	failed	1 ms
	creditStatus() tests		6 ms
	should equal adverse in restricted mode	failed	2 ms
	should equal adverse in default mode	failed	1 ms
	should equal good in restricted mode	passed	1 ms
	should equal good in default mode	failed	0 ms
	should equal invalid	failed	1 ms
	should equal invalid, case 2	failed	1 ms
<b> </b>	productStatus() tests		34 ms
	should equal invalid	failed	30 ms
	should equal soldout	passed	0 ms
	should equal limited	failed	2 ms
	should equal available	failed	2 ms
	orderHandling() tests		7 ms
	should equal accepted	failed	3 ms
	should equal pending	failed	2 ms
	should equal underReview	failed	1 ms
	should equal rejected	failed	1 ms

should equal bad in restricted mode should equal bad in restricted mode, case 2 failed should equal bad in default mode should equal bad in default mode, case 2 failed should equal good in restricted mode should equal good in restricted mode should equal good in restricted mode should equal good in default mode should equal good in default mode should equal good in default mode, case 2 should equal invalid passed should equal invalid passed should equal invalid, case 2  productStatus() tests should equal soldout should equal limited should equal available  should equal available  should equal available  passed  should equal available  passed  should equal available  passed  should equal accepted should equal accepted, case 2 should equal accepted, case 3 should equal accepted, case 4 should equal pending should equal pending should equal pending, case 3 failed should equal pending, case 3	0 ms 1 ms 2 ms 1 ms 0 ms 1 ms 0 ms 1 ms 0 ms 2 ms 0 ms 0 ms 0 ms
should equal bad in default mode should equal bad in default mode, case 2 should equal good in restricted mode should equal good in restricted mode should equal good in default mode, case 2 should equal invalid should equal invalid should equal invalid, case 2  passed should equal invalid, case 2  passed should equal invalid should equal limited failed should equal limited failed should equal available  passed  should equal available passed should equal available should equal available should equal accepted should equal accepted, case 2 should equal accepted, case 3 should equal accepted, case 4 should equal accepted, case 4 should equal pending should equal pending, case 2 failed should equal pending should equal pending, case 2	2 ms 1 ms 0 ms 1 ms 0 ms 1 ms 0 ms 2 ms
should equal bad in default mode, case 2 should equal good in restricted mode should equal good in restricted mode, case 2 should equal good in default mode should equal good in default mode should equal good in default mode, case 2 should equal invalid passed should equal invalid, case 2 passed should equal invalid, case 2 passed should equal invalid, case 2 passed should equal soldout should equal soldout should equal available passed should equal available passed should equal available passed should equal available passed should equal accepted should equal accepted, case 2 should equal accepted, case 2 should equal accepted, case 3 should equal accepted, case 4 should equal pending should equal pending, case 2 failed should equal pending failed	1 ms 0 ms 1 ms 0 ms 1 ms 0 ms 2 ms
should equal good in restricted mode should equal good in restricted mode, case 2 should equal good in default mode should equal good in default mode should equal good in default mode, case 2 should equal invalid should equal invalid should equal invalid, case 2 passed should equal soldout should equal soldout should equal soldout should equal available should equal available passed should equal available cision Table Testing orderHandling tests should equal accepted should equal accepted, case 2 should equal accepted, case 2 should equal accepted, case 3 should equal accepted, case 4 should equal pending should equal pending, case 2 failer should equal pending, case 2	0 ms 1 ms 0 ms 3 ms 1 ms 0 ms 2 ms
should equal good in restricted mode, case 2 should equal good in default mode should equal good in default mode, case 2 should equal invalid should equal invalid, case 2 passed should equal invalid, case 2 passed productStatus() tests should equal soldout should equal limited should equal available should equal available passed should equal available passed should equal available passed should equal available passed should equal accepted should equal accepted, case 2 should equal accepted, case 2 should equal accepted, case 3 should equal accepted, case 4 should equal pending should equal pending, case 2 failer should equal pending, case 2	1 ms 0 ms 3 ms 1 ms 0 ms
should equal good in default mode should equal good in default mode, case 2 failed should equal invalid passed should equal invalid, case 2 passed productStatus() tests should equal soldout should equal limited should equal available should equal available passed should equal available passed should equal available passed should equal available failed should equal accepted should equal accepted, case 2 should equal accepted, case 2 should equal accepted, case 3 should equal accepted, case 4 should equal accepted, case 4 should equal accepted, case 4 should equal pending failed should equal pending, case 2	0 ms 3 ms 1 ms 0 ms
should equal good in default mode, case 2 should equal invalid should equal invalid, case 2 passed  productStatus() tests should equal soldout should equal limited should equal available should equal available should equal available passed  should equal available passed should equal available passed should equal available passed should equal available passed should equal available passed should equal available passed should equal available passed should equal available passed should equal available should equal accepted should equal accepted, case 2 should equal accepted, case 3 should equal accepted, case 4 should equal accepted, case 4 should equal pending should equal pending should equal pending should equal pending, case 2	3 ms 1 ms 0 ms
should equal invalid, case 2  passed  productStatus() tests  should equal soldout  should equal limited  should equal available  passed  passed  should equal available  passed  passed  should equal available  passed  passed  passed  should equal available  passed  passed  passed  passed  should equal available  passed  passed  passed  passed  passed  passed  passed  passed  should equal available  passed  passe	1 ms 0 ms 2 ms
should equal invalid, case 2  productStatus() tests  should equal soldout passed should equal limited failed should equal available passed  should equal available passed  should equal available failed should equal available passed  cision Table Testing orderHandling tests  should equal accepted failed should equal accepted, case 2 should equal accepted, case 3 should equal accepted, case 4 should equal pending failed should equal pending failed should equal pending, case 2	0 ms
should equal soldout passed should equal limited failed should equal available passed  should equal available passed  should equal available passed  should equal available passed  cision Table Testing orderHandling tests  should equal accepted failed should equal accepted, case 2 failed should equal accepted, case 3 failed should equal accepted, case 4 failed should equal accepted, case 4 failed should equal pending failed should equal pending failed should equal pending, case 2 failed	2 ms
should equal limited failed should equal available passed  should equal available passed  should equal available passed  should equal available passed  cision Table Testing orderHandling tests  should equal accepted failed should equal accepted, case 2 failed should equal accepted, case 3 failed should equal accepted, case 4 failed should equal accepted, case 4 failed should equal pending failed should equal pending failed	
should equal limited passed  should equal available passed  should equal available passed  cision Table Testing orderHandling tests  should equal accepted failer should equal accepted, case 2 failer should equal accepted, case 3 failer should equal accepted, case 4 failer should equal pending failer should equal pending failer should equal pending failer should equal pending, case 2 failer should equal pending, case 2	0 ~~
should equal available  passed  cision Table Testing orderHandling tests  should equal accepted should equal accepted, case 2 should equal accepted, case 3 should equal accepted, case 3 should equal accepted, case 4 should equal accepted, case 4 should equal pending should equal pending should equal pending, case 2	UIIIS
should equal available  cision Table Testing orderHandling tests  should equal accepted failer should equal accepted, case 2 should equal accepted, case 3 should equal accepted, case 4 should equal accepted, case 4 should equal pending failer should equal pending failer should equal pending, case 2	2 ms
cision Table Testing orderHandling tests  should equal accepted should equal accepted, case 2 should equal accepted, case 3 should equal accepted, case 4 should equal accepted, case 4 should equal pending should equal pending should equal pending, case 2	0 ms
should equal accepted, case 2 should equal accepted, case 3 should equal accepted, case 4 should equal pending should equal pending should equal pending, case 2 failer	12 12
should equal accepted, case 3 should equal accepted, case 4 should equal pending failer should equal pending, case 2 failer	1
should equal accepted, case 4 failer should equal pending failer should equal pending, case 2 failer	1
should equal pending failed should equal pending, case 2 failed	2
should equal pending, case 2 failed	1
one and a square per a square p	1
should equal pending, case 3	1
	1
should equal underReview faile	
should equal underReview, case 2 failed	2
should equal rejected faile	
should equal rejected, case 2 passed	1
should equal rejected, case 3 passed	1
should equal rejected, case 4 passed	1 0
should equal rejected, case 5 passed	1 0 1 0

#### After Code Fixes

maset	Order: 86 total, 86 passed	
		Collapse   I
urchaseC	OrderF19.test.js	
Purchas	aseOrder	
Equ	uivalence Tests	
	getAgeFactor() tests	
	should equal 0	passed
	should equal 0, case 2	passed
	should equal 5	passed
	should equal 10	passed
	should equal 20	passed
	should equal 20, case 2	passed
	should equal 50	passed
	getBalanceFactor() tests	
	should equal 0	passed
	should equal 0, case 2	passed
	should equal 6	passed
	should equal 16	passed
	should equal 30	passed
	should equal 70	passed
	should equal 200	passed
	AccountStatus() tests	
	should equal invalid	passed
	should equal adverse	passed
	should equal acceptable	passed
	should equal good	passed
	should equal excellent	passed
reditStat	itus() tests	
should	equal adverse in restricted mode	passed
should	equal adverse in default mode	passed
	equal good in restricted mode	passed
	l equal good in default mode	passed
	l equal invalid	passed
	l equal invalid, case 2	passed
roductS <sup>r</sup>	Status() tests	
	l equal invalid	passed
	l equal soldout	passed
should	equal limited	passed
	l equal available	passed
should		
should should	hdling() tasts	
should should rderHand	ndling() tests	hassan
should should rderHand should	equal accepted	passed
should should rderHand should should		passed passed passed

	undary Value Tests		7 m
			-
	getAgeFactor() tests		3 m
	should equal 0	passed	0 m
	should equal 0, case 2	passed	0 m
	should equal 5	passed	1 m:
	should equal 5, case2	passed	0 m:
	should equal 10	passed	0 m
	should equal 10, case 2	passed	1 m:
	should equal 20	passed	0 m:
	should equal 20, case 2	passed	0 m:
	should equal 20, case 3	passed	1 m
	should equal 50, case 1	passed	0 m:
	should equal 50, case 2	passed	0 m
	should equal 50, case 4	passed	0 m
	getBalanceFactor() tests		2 m
	should equal 0	passed	0 m
	should equal 0, case 2	passed	0 m
	should equal 6	passed	0 m
	should equal 6, case2	passed	0 m
	should equal 16	passed	0 m
	should equal 16, case 2	passed	0 m
	should equal 30	passed	1 m
	should equal 30, case 2	passed	0 m
	should equal 70	passed	0 m
	should equal 70, case 2	passed	0 m
	should equal 200	passed	1 m
	should equal 200, case 2	passed	0 m
Ac	countStatus() tests		0
Т	should equal invalid	passed	0
т	should equal excellent	passed	0 1
	•		
cre	editStatus() tests		21
		passed	11
L	should equal adverse in restricted mode	passed	1 :
	should equal adverse in restricted mode should equal adverse in restricted mode, case 2	passed	1 i
	should equal adverse in restricted mode should equal adverse in restricted mode, case 2 should equal adverse in default mode	passed passed	0
	should equal adverse in restricted mode should equal adverse in restricted mode, case 2 should equal adverse in default mode should equal adverse in default mode, case 2	passed passed passed	1 0 0
	should equal adverse in restricted mode should equal adverse in restricted mode, case 2 should equal adverse in default mode should equal adverse in default mode, case 2 should equal good in restricted mode	passed passed passed passed	0 0 0 0
	should equal adverse in restricted mode should equal adverse in restricted mode, case 2 should equal adverse in default mode should equal adverse in default mode should equal good in restricted mode should equal good in restricted mode should equal good in restricted mode, case 2	passed passed passed passed passed	1 0 0 0 0 0 1
	should equal adverse in restricted mode should equal adverse in restricted mode, case 2 should equal adverse in default mode should equal adverse in default mode should equal good in restricted mode, case 2 should equal good in restricted mode, case 2 should equal good in restricted mode, case 2 should equal good in default mode	passed passed passed passed passed passed	1 0 0 0 0 1
	should equal adverse in restricted mode should equal adverse in restricted mode, case 2 should equal adverse in default mode should equal adverse in default mode should equal good in restricted mode should equal good in restricted mode should equal good in restricted mode, case 2 should equal good in default mode should equal good in default mode should equal good in default mode, case 2	passed passed passed passed passed passed passed	1 0 0 0 0 1 0
	should equal adverse in restricted mode should equal adverse in restricted mode, case 2 should equal adverse in default mode should equal adverse in default mode should equal good in restricted mode should equal good in restricted mode should equal good in restricted mode should equal good in default mode, case 2 should equal good in default mode, case 2 should equal good in default mode, case 2	passed passed passed passed passed passed passed passed passed	1 0 0 0 0 1 0 0
	should equal adverse in restricted mode should equal adverse in restricted mode, case 2 should equal adverse in default mode should equal adverse in default mode should equal good in restricted mode should equal good in restricted mode should equal good in restricted mode, case 2 should equal good in default mode should equal good in default mode should equal good in default mode, case 2	passed passed passed passed passed passed passed	1 0 0 0 0 1 0 0
	should equal adverse in restricted mode should equal adverse in restricted mode, case 2 should equal adverse in default mode should equal good in restricted mode, case 2 should equal good in restricted mode should equal good in restricted mode should equal good in festirited mode should equal good in default mode, case 2 should equal good in default mode should equal good in default mode should equal good in default mode should equal invalid should equal invalid, case 2	passed passed passed passed passed passed passed passed passed	1 0 0 0 0 1 0 0
pre	should equal adverse in restricted mode should equal adverse in restricted mode, case 2 should equal adverse in default mode should equal good in restricted mode should equal good in restricted mode should equal good in restricted mode should equal good in festirited mode should equal good in default mode, case 2 should equal good in default mode should equal good in default mode should equal good in westirited mode should equal invalid should expect in westirited mode shoul	passed passed passed passed passed passed passed passed	1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
pre	should equal adverse in restricted mode should equal adverse in restricted mode, case 2 should equal adverse in default mode should equal good in restricted mode, case 2 should equal good in restricted mode should equal good in restricted mode, case 2 should equal good in default mode, case 2 should equal good in default mode should equal good in default mode should equal good in default mode, case 2 should equal good in default mode, case 2 should equal invalid, case 2	passed passed passed passed passed passed passed passed passed	1 0 0
pre	should equal adverse in restricted mode should equal adverse in restricted mode, case 2 should equal adverse in default mode should equal good in restricted mode should equal good in restricted mode should equal good in restricted mode should equal good in festirited mode should equal good in default mode, case 2 should equal good in default mode should equal good in default mode should equal good in westirited mode should equal invalid should expect in westirited mode shoul	passed passed passed passed passed passed passed passed	1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
pre	should equal adverse in restricted mode should equal adverse in restricted mode, case 2 should equal adverse in default mode should equal good in restricted mode, case 2 should equal good in restricted mode should equal good in restricted mode, case 2 should equal good in default mode, case 2 should equal good in default mode should equal good in default mode should equal good in default mode, case 2 should equal good in default mode, case 2 should equal invalid, case 2	passed	11 00 00 00 11 00 00 00 00
pre	should equal adverse in restricted mode should equal adverse in restricted mode, case 2 should equal adverse in default mode should equal good in restricted mode, case 2 should equal good in restricted mode, case 2 should equal good in restricted mode, case 2 should equal good in default mode should equal good in default mode should equal good in default mode should equal good in default mode, case 2 should equal invalid should equal invalid, case 2  oductStatus() tests should equal soldout should equal soldout should equal limited	passed	11 00 00 00 11 00 00 00 00
pre	should equal adverse in restricted mode should equal adverse in restricted mode, case 2 should equal adverse in default mode should equal good in restricted mode, case 2 should equal good in restricted mode, case 2 should equal good in default mode, case 2 should equal good in default mode, case 2 should equal good in default mode should equal good in default mode should equal good in setricted mode should equal good in default mode, case 2 should equal good in default mode, case 2 should equal invalid, case 2  oductStatus() tests should equal invalid, case 2 should equal soldout should equal soldout should equal limited should equal available	passed	1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
pro	should equal adverse in restricted mode, case 2 should equal adverse in default mode, case 2 should equal good in restricted mode, case 2 should equal good in restricted mode, case 2 should equal good in restricted mode, case 2 should equal good in default mode should equal good in default mode should equal good in default mode, case 2 should equal invalid should equal invalid, case 2  oductStatus() tests should equal soldout should equal soldout should equal soldout should equal limited should equal available	passed	11 00 00 00 11 00 00 00 00 00
pro	should equal adverse in restricted mode should equal adverse in default mode should equal adverse in default mode, case 2 should equal good in restricted mode, case 2 should equal good in restricted mode, case 2 should equal good in festurited mode, case 2 should equal good in default mode, case 2 should equal good in default mode, case 2 should equal good in default mode should equal good in well to make the should equal good in well to make the should equal invalid should equal invalid, case 2  oductStatus() tests should equal soldout should equal soldout should equal limited should equal available  cision Table Testing orderHandling tests	passed	1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
pro	should equal adverse in restricted mode should equal adverse in default mode should equal adverse in default mode, case 2 should equal good in restricted mode, case 2 should equal good in restricted mode should equal good in default mode should equal good in default mode should equal good in default mode, case 2 should equal invalid should equal invalid should equal invalid, case 2  oductStatus() tests should equal soldout should equal soldout should equal limited should equal available  cision Table Testing orderHandling tests should equal accepted	passed	1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
pro	should equal adverse in restricted mode should equal adverse in default mode should equal adverse in default mode, case 2 should equal good in restricted mode, case 2 should equal good in restricted mode, case 2 should equal good in festurited mode, case 2 should equal good in default mode, case 2 should equal good in default mode, case 2 should equal good in default mode should equal good in well to make the should equal good in well to make the should equal invalid should equal invalid, case 2  oductStatus() tests should equal soldout should equal soldout should equal limited should equal available  cision Table Testing orderHandling tests	passed	1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
pro	should equal adverse in restricted mode should equal adverse in default mode should equal adverse in default mode, case 2 should equal good in restricted mode, case 2 should equal good in restricted mode should equal good in default mode should equal good in default mode should equal good in default mode, case 2 should equal invalid should equal invalid should equal invalid, case 2  oductStatus() tests should equal soldout should equal soldout should equal limited should equal available  cision Table Testing orderHandling tests should equal accepted	passed	1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
pro	should equal adverse in restricted mode should equal adverse in default mode should equal good in restricted mode, case 2 should equal good in restricted mode should equal good in restricted mode should equal good in festricted mode should equal good in default mode, case 2 should equal good in default mode, case 2 should equal good in default mode, case 2 should equal good in default mode, case 3 should equal good in default mode, case 2 should equal invalid should equal invalid should equal invalid, case 2  oductStatus() tests should equal soldout should equal soldout should equal limited should equal available  cision Table Testing orderHandling tests should equal accepted, case 2	passed	11 00 00 00 11 00 00 00 00 00 00 00 00 0
pro	should equal adverse in restricted mode, case 2 should equal adverse in default mode should equal adverse in default mode, case 2 should equal good in restricted mode should equal good in restricted mode should equal good in restricted mode should equal good in default mode, case 2 should equal invalid should equal invalid should equal invalid, case 2  cision Table Testing orderHandling tests  should equal accepted, case 3 should equal accepted, case 3 should equal accepted, case 3	passed	11 00 00 00 11 00 00 00 00 00 00 00 00 0
pro	should equal adverse in restricted mode, case 2 should equal adverse in default mode should equal adverse in default mode, case 2 should equal good in restricted mode, case 2 should equal good in restricted mode, case 2 should equal good in default mode should equal good in the fault mode should equal good in default mode should equal soldout should equal soldout should equal soldout should equal available  should equal available  should equal accepted should equal accepted, case 2 should equal accepted, case 3 should equal accepted, case 4	passed	11 00 00 00 00 00 00 00 00 00 00 00 00 0
pro	should equal adverse in restricted mode should equal adverse in default mode should equal adverse in default mode should equal good in restricted mode, case 2 should equal good in restricted mode, case 2 should equal good in restricted mode, case 2 should equal good in default mode should equal invalid should equal invalid, case 2  dductStatus() tests should equal soldout should equal available  cision Table Testing orderHandling tests  should equal accepted, case 2 should equal accepted, case 3 should equal accepted, case 2 should equal accepted, case 3 should equal pending, case 2	passed	11 00 00 00 00 00 00 00 00 00 00 00 00 0
pro	should equal adverse in restricted mode, case 2 should equal adverse in default mode should equal adverse in default mode should equal adverse in default mode should equal good in restricted mode, case 2 should equal good in restricted mode, case 2 should equal good in restricted mode, case 2 should equal good in festricted mode, case 2 should equal good in default mode should equal good in default mode should equal invalid should equal invalid, case 2  oductStatus() tests should equal invalid, case 2  oductStatus() tests should equal soldout should equal soldout should equal acapted should equal acapted should equal accepted, case 2 should equal accepted, case 3 should equal accepted, case 4 should equal accepted, case 4 should equal accepted, case 4 should equal pending, case 2 should equal pending, case 3 should equal pending, case 3 should equal pending, case 3	passed	11 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
pro	should equal adverse in restricted mode should equal adverse in default mode should equal good in restricted mode should equal good in restricted mode should equal good in festricted mode should equal good in default mode should equal good in default mode should equal good in default mode should equal invalid should equal invalid, case 2  oductStatus() tests should equal soldout should equal aloidout should equal	passed	11 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
pro	should equal adverse in restricted mode, case 2 should equal adverse in default mode, case 2 should equal adverse in default mode, case 2 should equal good in restricted mode, case 2 should equal good in restricted mode should equal good in default mode should equal invalid should equal invalid, case 2  ***OutcitStatus() tests** should equal soldout should equal allimited should equal allimited ***should equal equa	passed	11 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
pro	should equal adverse in restricted mode, case 2 should equal adverse in default mode should equal adverse in default mode, case 2 should equal adverse in default mode, case 2 should equal good in restricted mode, case 2 should equal good in restricted mode, case 2 should equal good in default mode, case 2 should equal social mode, case 2 should equal available should equal available should equal available should equal accepted, case 3 should equal pending, case 3	passed	11 00 00 00 00 00 00 00 00 00 00 00 00 0
pro	should equal adverse in restricted mode should equal adverse in refault mode should equal adverse in default mode should equal adverse in default mode should equal adverse in default mode should equal good in restricted mode should equal good in restricted mode should equal good in restricted mode, case 2 should equal good in default mode should equal good in default mode, case 2 should equal good in default mode, case 2 should equal good in default mode, case 2 should equal pool in default mode, case 2 should equal invalid, case 2  oductStatus() tests  should equal available  cision Table Testing  orderHandling tests  should equal accepted, case 3 should equal accepted, case 3 should equal accepted, case 4 should equal accepted, case 3 should equal pending, case 3 should equal moderReview should equal rejected, case 2 should equal pending, case 3	passed	11 00 00 00 00 00 00 00 00 00 00 00 00 0
pro	should equal adverse in restricted mode, case 2 should equal adverse in default mode, case 2 should equal adverse in default mode, case 2 should equal good in restricted mode, case 2 should equal good in restricted mode, case 2 should equal good in restricted mode, case 2 should equal good in default mode, case 2 should equal should equal for individual mode, case 2 should equal should equal should equal invalid, case 2  dductStatus() tests should equal soldout should equal soldout should equal soldout should equal accepted, case 3 should equal pending, case 2 should equal pending, case 2 should equal pending, case 3	passed	11 00 00 00 00 00 00 00 00 00 00 00 00 0
pro	should equal adverse in restricted mode should equal adverse in refault mode should equal adverse in default mode should equal adverse in default mode should equal adverse in default mode should equal good in restricted mode should equal good in restricted mode should equal good in restricted mode, case 2 should equal good in default mode should equal good in default mode, case 2 should equal good in default mode, case 2 should equal good in default mode, case 2 should equal pool in default mode, case 2 should equal invalid, case 2  oductStatus() tests  should equal available  cision Table Testing  orderHandling tests  should equal accepted, case 3 should equal accepted, case 3 should equal accepted, case 4 should equal accepted, case 3 should equal pending, case 3 should equal moderReview should equal rejected, case 2 should equal pending, case 3	passed	11 00 00 00 00 00 00 00 00 00 00 00 00 0