Logic Coverage

Software Testing

(3104313)

Amirkabir University of Technology

Spring 1399-1400

Logic Predicates and Clauses

- A predicate is an expression that evaluates to a boolean value

- Predicates can contain
  - boolean variables
  - non-boolean variables that contain >, <, ==, >=, <=, !=
  - boolean function calls

- Internal structure is created by logical operators
  - $\neg$ – the *negation* operator
  - $\wedge$ – the *and* operator
  - $\vee$ – the *or* operator
  - $\rightarrow$ – the *implication* operator
  - $\oplus$ – the *exclusive or* operator
  - $\leftrightarrow$ – the *equivalence* operator

- A clause is a predicate with no logical operators

Example

$$(a < b) \vee f(z) \wedge D \wedge (m >= n*o)$$

- Four clauses
  1. a < b
  2. f(z)
  3. D
  4. m <= n*o

Example

```
if ((a > b) || c) && (x < y))
        o.m();
else
        o.n();
```

$((a > b) \vee C) \wedge (x < y)$

```
button2 = true (when gear = park)
```

$gear = park \wedge button2 = true$

```
"pre: stack Not full AND object
reference parameter not null"
```

$\neg stackFull() \wedge newObj \neq null$

Logic Coverage Criteria

- Develop a **model** of the software as **one or more predicates**
- Require **tests** to satisfy some **combination of clauses**

Predicate & Clause Coverage

- *P* is the set of predicates
- *p* is a single predicate in *P*
- *C* is the set of clauses in *P*
- *c* is a single clause in *C*

Predicate Coverage (PC): For each *p* in *P*, *TR* contains two requirements: *p* evaluates to *true*, and *p* evaluates to *false*

Predicate & Clause Coverage

- *P* is the set of predicates
- *p* is a single predicate in *P*
- *C* is the set of clauses in *P*
- *c* is a single clause in *C*

Predicate Coverage (PC): For each *p* in *P*, *TR* contains two requirements: *p* evaluates to *true*, and *p* evaluates to *false*

Clause Coverage (CC): For each *c* in *C*, *TR* contains two requirements: *c* evaluates to *true*, and *c* evaluates to *false*.

In/After-Class Exercise #17

$$((a < b) \vee D) \wedge (m >= n*o)$$

Give abstract tests applying

1. predicate coverage (PC)
2. clause coverage (CC)

- You have ∞ minutes ☺, but now think 2-3 minutes!
- Do the exercise individually/in groups (of 3)
- Upload your answer in Moodle.

((a < b) ∨ D) ∧ (m >= n*o)

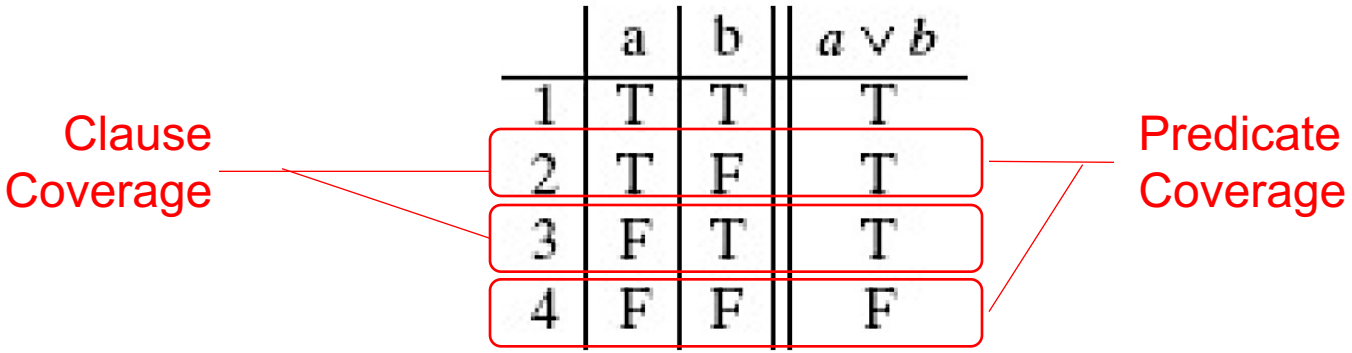
Predicate Coverage

- True
  - a < b = true
  - D = true
  - m <= n∗o = true
- False
  - !(a < b) (false)
  - !D (false)
  - m <= n∗o (true)

Clause Coverage

1. Test 1
   - a < b = true
   - D = true
   - m <= n*o = true
2. Test 2
   - a < b = false
   - D = false
   - m <= n*o = false

Problems with PC and CC!



	a	b	$a \vee b$
1	T	T	T
2	T	F	T
3	F	T	T
4	F	F	F

Problems with PC and CC!

Neither Predicate Coverage nor Clause Coverage subsumes the other.

Combinatorial Coverage (CoC)

Combinatorial Coverage (CoC): For each *p* in *P*, *TR* has test requirements for the clauses in *Cp* to evaluate to each possible combination of truth values.

In/After-Class Exercise #17

$$((a < b) \vee D) \wedge (m >= n*o)$$

Give abstract tests using

3. combinatorial coverage (CoC).

- You have ∞ minutes ☺, but now think 2-3 minutes!
- Do the exercise individually/in groups (of 3)
- Upload your answer in Moodle.

Combinatorial Coverage (CoC)

Combinatorial Coverage (CoC): For each *p* in *P*, *TR* has test requirements for the clauses in *Cp* to evaluate to each possible combination of truth values.

	a < b	D	m >= n*o	((a < b) $\vee$ D) $\wedge$ (m >= n*o)
1	T	T	T	T
2	T	T	F	F
3	T	F	T	T
4	T	F	F	F
5	F	T	T	T
6	F	T	F	F
7	F	F	T	F
8	F	F	F	F

ACTIVE CLAUSES

To really test the results of a clause, the clause should be the determining factor in the value of the predicate

P = (a & (b | c))

Test 1: (true & (true | true))
Test 2: (false & (false | false))

ACTIVE CLAUSES

To really test the results of a clause, the clause should be the determining factor in the value of the predicate

Active Clauses - Determination

Clause c_i **determines** the value of its predicate when the other clauses have certain values, such that if c_i is changed the value of the predicate changes.

- c_i is called the **major** clause.
- Other clauses are **minor** clauses

Determining Predicates

$P = A \vee B$	$P = A \wedge B$
if B = true, p is always true.	if B = false, p is always false.
so if B = false, A determines p.	so if B = true, A determines p.
if A = false, B determines p.	if A = true, B determines p.

Goal:

Find tests for each clause when the clause determines the value of the predicate.

In/After-Class Exercise #18

P = (a ∧ (b ∨ c))

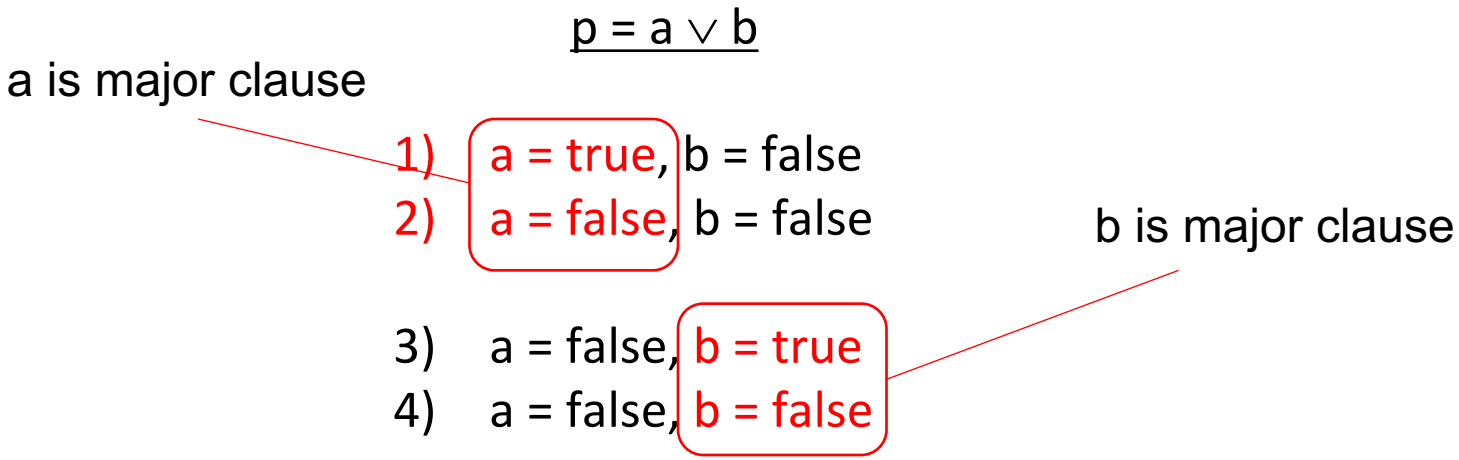
1. Write truth values for *b and c* that make clause ***a*** **active**

   For example: Pa : b=?? or c=??
2. Write truth values for *a and c* that make clause ***b*** **active**
3. Write truth values for *a and b* that make clause ***c*** **active**

- You have ∞ minutes ☺, but now think 2-3 minutes!
- Do the exercise individually/in groups (of 3)
- Upload your answer in Moodle.

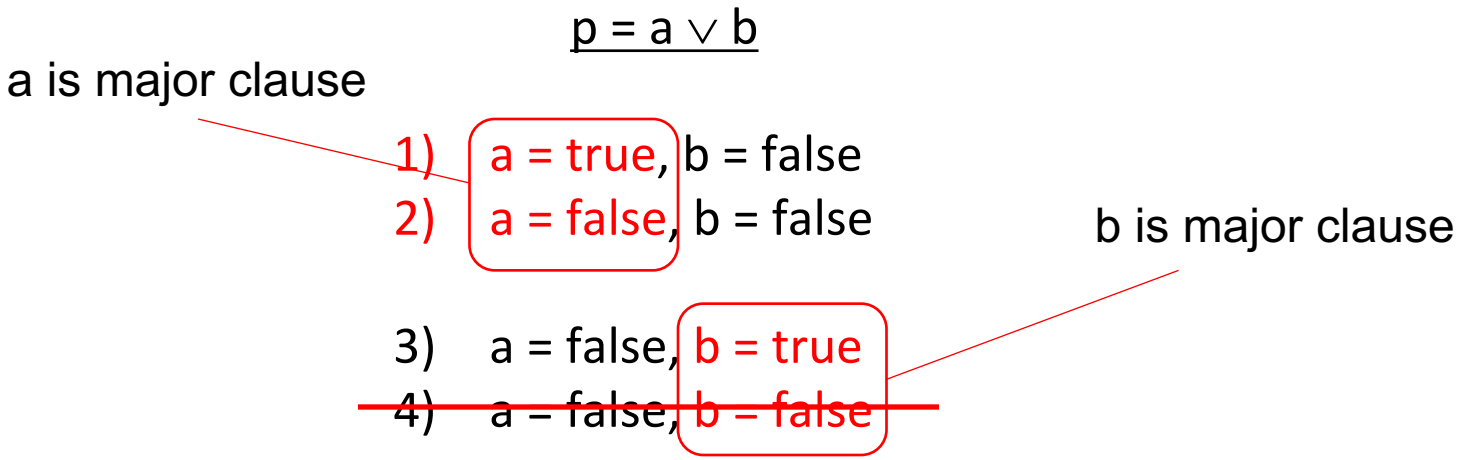
Active Clause Coverage

Active Clause Coverage (ACC): For each clause c_i in each predicate p, choose values for the other clauses to make c_i active. TR has two requirements for each c_i: c_i evaluates to true and c_i evaluates to false.



Active Clause Coverage

Active Clause Coverage (ACC): For each clause c_i in each predicate p, choose values for the other clauses to make c_i active. TR has two requirements for each c_i: c_i evaluates to true and c_i evaluates to false.



ACC Ambiguity

$$\mathbf{p = a \wedge (b \vee c)}$$

Major clause: a

a = true, b = true, c = true

a = false, b = true, c = true

ACC Ambiguity

$$\mathbf{p = a \wedge (b \vee c)}$$

Major clause: a

a = true, b = true, c = true

a = false, b = true, c = false

ACC Ambiguity

p = a ∧ (b ∨ c)

Major clause: a

a = true, b = true, c = true

a = false, b = true, c = false

Correlated Active Clause Coverage (CACC): For each *p* in *P* and each major clause c_i in C_p, choose minor clauses c_j, j != i, so that c_i determines *p*. TR has two requirements for each c_i: c_i evaluates to true and c_i evaluates to false. The values chosen for the minor clauses c_j must cause p to be true for one value of the major clause c_i and false for the other, that is, it is required that

$$p(c_i = true) != p(c_i = false)$$

CACC Example

	a	b	c	$a \wedge (b \vee c)$
1	T	T	T	T
2	T	T	F	T
3	T	F	T	T
4	T	F	F	F
5	F	T	T	F
6	F	T	F	F
7	F	F	T	F
8	F	F	F	F

CACC Example

	a	b	c	$a \wedge (b \vee c)$
1	T	T	T	T
2	T	T	F	T
3	T	F	T	T
4	T	F	F	F
5	F	T	T	F
6	F	T	F	F
7	F	F	T	F
8	F	F	F	F

For a to determine the value of the predicate

P_a: b=true or c = true

So we can use ANY OF the 9 pair of rows: (1,5), (1,6), (1,7), (2,5),(2,6),(2,7), (3,5),(3,6),(3,7)

CACC Example

	a	b	c	$a \wedge (b \vee c)$
1	T	T	T	T
2	T	T	F	T
3	T	F	T	T
4	T	F	F	F
5	F	T	T	F
6	F	T	F	F
7	F	F	T	F
8	F	F	F	F

For a to determine the value of the predicate

P_a: b=true or c = true

So we can use ANY OF the 9 pair of rows: (1,5), (1,6), (1,7), (2,5),(2,6),(2,7), (3,5),(3,6),(3,7)

For b to determine the value of the predicate

P_b: a=true and c = false

Rows 2 and 4

For c to determine the value of the predicate

P_c: a=true and b = false

Rows 3 and 4

Making Clauses Determine a Predicate?

- Informal by **inspection**
- **Tabular** method
- **Definitional** method

Tabular Method

	a	b	c	$a \wedge (b \vee c)$	p_a	p_b	p_c
1	T	T	T	T			
2	T	T	F	T			
3	T	F	T	T			
4	T	F	F	F			
5	F	T	T	F			
6	F	T	F	F			
7	F	F	T	F			
8	F	F	F	F			

After-Class Exercise #19

$$P = (a \wedge (b \vee c))$$

1. Use the tabular method to solve for Pa, Pb, and Pc.

Give solutions as pairs of rows

- You have ∞ minutes ☺, but now think 2-3 minutes!
- Do the exercise individually/in groups (of 3)
- Upload your answer in Moodle.

Definitional method

- To find values for the minor clauses, connect $p_{c=true}$ and $p_{c=false}$ with exclusive OR

$$p_c = p_{c=true} \oplus p_{c=false}$$

$p = a \vee (b \wedge c)$

$$\begin{aligned} p_a &= p_{a=true} \oplus p_{a=false} \\ &= (true \vee (b \wedge c)) \oplus (false \vee (b \wedge c)) \\ &= true \oplus (b \wedge c) \\ &= !(b \wedge c) \\ &= !b \vee !c \end{aligned}$$

After-Class Exercise #19

$P = (a \wedge (b \vee c))$

2. Use the definitional method to solve for Pa, Pb, and Pc.

- You have ∞ minutes ☺, but now think 2-3 minutes!
- Do the exercise individually/in groups (of 3)
- Upload your answer in Moodle.

Applying
Logic Coverage Criteria

Programs

Structural Logic Coverage

- Predicates are derived directly from decision statements in the programs (if, case, and loop statements).

Finding Values?

- Reachability
- Controllability
- Internal variables

```
public int checkVal(int x) {
   int y = x*2;
   if (x>0)
      if ((x>10 && x<20) || y==50)
         return 1;
   else
      if ((x<-10 && x>-20) || y<-60)
         return 2;
   return 42;
}
```

After-Class Exercise #20

Go to https://cs.gmu.edu/~offutt/softwaretest/java/Thermostat.java

1. Identify the predicates.
2. Consider reachability, and determine conditions under which each predicate is reached.
   - Choose values to make the predicates reachable.
3. Which variables are internal? How are they handled?

- You have ∞ minutes ☺, but now think 5 minutes!
- Do the exercise individually/in groups (of 3)
- Upload your answer in Moodle.

Programs

Structural Logic Coverage

- Predicates are derived directly from decision statements in the programs (if, case, and loop statements).

- Predicate Transformation Issue?
- Side Effects in Predicates?

Finding Values?

- Reachability
- Controllability
- Internal variables

Logic Coverage for Specification

"list all the wireless mice that either retail for more than $100 or for which the store has more than 20 items. Also list non-wireless mice that retail for more than $50."

```
public static int cal (int month1, int day1, int month2,
                       int day2, int year) {
//***********************************************
// Calculate the number of Days between the two given days in
// the same year.
// preconditions : day1 and day2 must be in same year
//                  1 <= month1, month2 <= 12
//                  1 <= day1, day2 <= 31 // month1 <= month2
//                  The range for year: 1 ... 10000
//***********************************************
```

Logic Coverage for Specification

"lis ... h
the ... or
... more than $50.

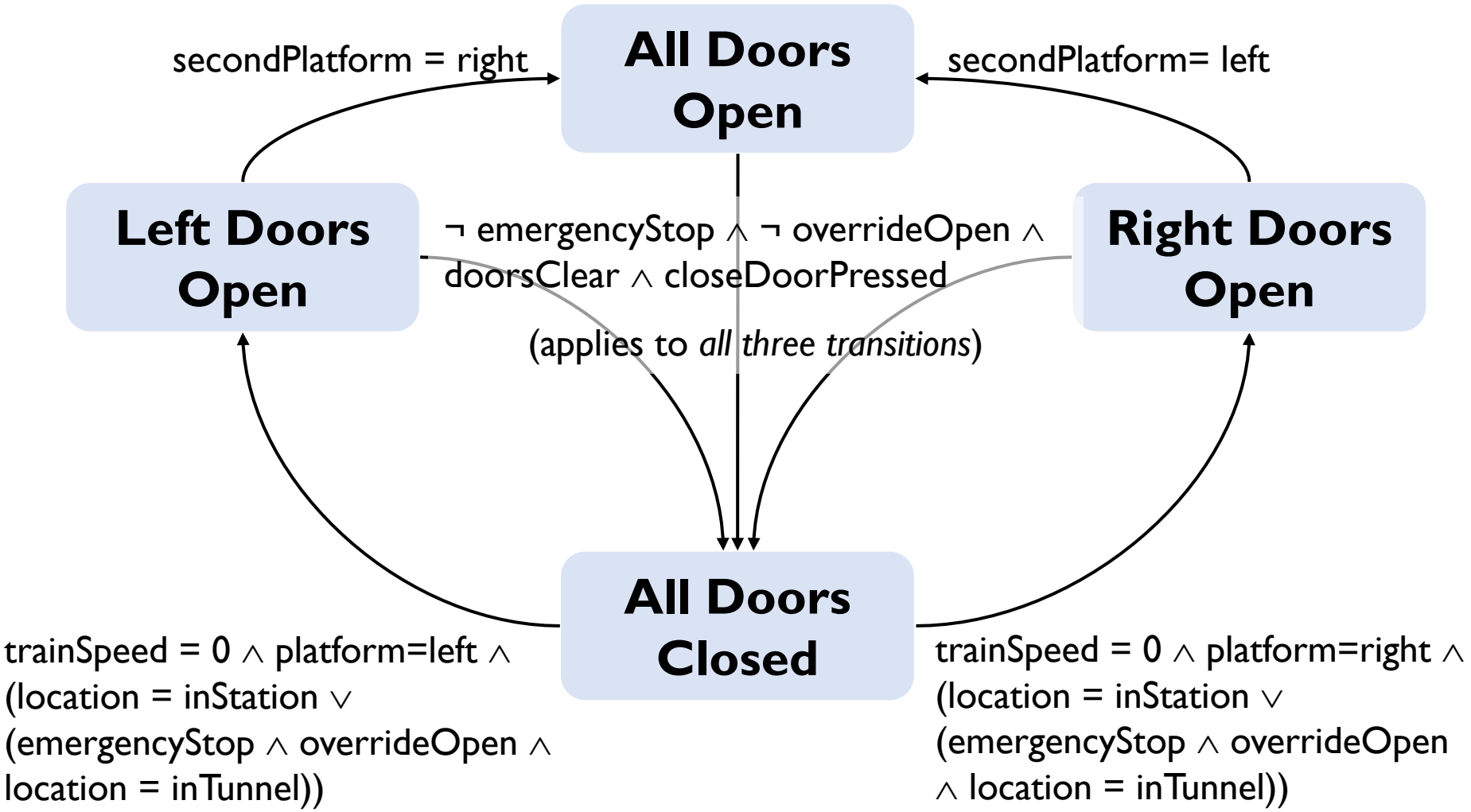
$$((mouseType = wireless) \wedge ((retail > 100) \vee (stock > 20))) \vee (\neg(mouseType = wireless) \wedge (retail > 50))$$

```
public static int cal (int month1, int day1, int month2,
                       int day2, int year) {
//*************************************************
```

$$month1 >= 1 \wedge month1 <= 12 \wedge month2 >= 1 \wedge month2 <= 12 \wedge month1 <= month2$$

$$\wedge day1 >= 1 \wedge day1 <= 31 \wedge day2 >= 1 \wedge day2 <= 31 \wedge year >= 1 \wedge year <= 10000$$

```
//                    1 <= month1, month2 <= 12
//                    1 <= day1, day2 <= 31 // month1 <= month2
//                    The range for year: 1 ... 10000
//*************************************************
```


All Doors Open
secondPlatform = right
secondPlatform= left
Left Doors Open
Right Doors Open
¬ emergencyStop ∧ ¬ overrideOpen ∧ doorsClear ∧ closeDoorPressed
(applies to all three transitions)
All Doors Closed
trainSpeed = 0 ∧ platform=left ∧ (location = inStation ∨ (emergencyStop ∧ overrideOpen ∧ location = inTunnel))
trainSpeed = 0 ∧ platform=right ∧ (location = inStation ∨ (emergencyStop ∧ overrideOpen ∧ location = inTunnel))

- Durelli et al., 2016. *What to Expect of Predicates: An Empirical Analysis of Predicates in Real World Programs*, Journal of Systems and Software.

- Botelho et al., 2017. *On the costs of applying logic-based criteria to mobile applications: An empirical analysis of predicates in real-world Objective-C and Swift applications.* In Proc. of the 2nd Brazilian Symposium on Systematic and Automated Software Testing.

Logic Coverage Criteria

Find a logical expression and cover it!