Testing II

Week 15
Agenda (Lecture)

• White box testing
  – Statement coverage
  – Branch coverage
Agenda (Lab)

- Review of SRS/SDD documents
- Implementation
- Submit a weekly project progress report at the end of this week lab session
White Box Testing

- White box testing
- Glass box testing
- Structural testing
- Logic-driven testing
- Path-oriented testing
- Testing to code

Input 1
Input 2
...
Input n

Expected Output 1
Expected Output 2
...
Expected Output n

Test Case 1
Test Case 2
...
Test Case n

Output 1
Output 2
...
Output n

The most common form of testing to code requires that each path through the code be executed at least once.
White Box Testing

• Statement coverage
• Branch coverage
• Condition coverage
• Loop coverage
• Path coverage
White Box Testing

• Advantages
  – Improve the quality of code
  – Reveals errors in "hidden" code

• Disadvantage
  – Expensive
  – Need access to source
Flow Graph

• A directed graph
  – A set of $N$ of nodes
  – A set of $E \subseteq N \times N$ of directed edges between nodes
  – Formal definition: A graph $G$ is a pair $(V, E)$, where $V$ is a set of vertexes, and $E$ is a set of edges between the vertexes $E = \{(u, v) \mid u, v \in V\}$. 
Flow Graph

$\text{n} \text{th statement}$

Begin node

Transfer of control (2, 4)

End node
Conditions of Triangle

- If you have three numbers, a, b, and c, which are the lengths of the side of a triangle, the conditions in triangle problem might be
  - $a = b$ or $a = c$ or $b = c$
  - $a = b$ and $b = c$
  - $a < b + c$ and $b < a + c$ and $c < a + b$
  - $a > 0$ and $b > 0$ and $c > 0$
Triangle Problem

• Equivalence classes
  – Scalene
  – Isosceles
  – Equilateral
  – Not a triangle
  – Bad inputs
Triangle Problem

• Equivalence classes
  – Scalene
    • Increasing size - (3, 4, 5)
    • Decreasing size - (5, 4, 3)
    • Largest as second - (4, 5, 3)
  – Isosceles
    • a = b & other side larger – (5, 5, 8)
    • a = c & other side larger – (5, 8, 5)
    • b = c & other side larger – (8, 5, 5)
    • a = b & other side smaller – (8, 8, 5)
    • a = c & other side smaller – (8, 5, 8)
    • b = c & other side smaller – (5, 8, 8)
Triangle Problem (cont’d)

– Equilateral
  • All sides equal (5, 5, 5)

– Not a triangle
  • Largest first (6, 4, 2)
  • Largest second (4, 6, 2)
  • Largest third (1, 2, 3)

– Bad inputs
  • One bad input (-1, 2, 4)
  • Two bad inputs (3, -2, -5)
  • Three bad inputs (0, 0, 0)
Statement Coverage

• Every statement in program is executed at least once
• Every node in Control Flow Graph is visited at least once
• Minimize number of test cases
Statement Coverage

Node  | Source Line
--- | ---
A    | read a, b, c
B    | type = “scalene”
C    | if (a==b || b==c || a==c)
D    | type = “isosceles”
E    | if (a == b && b == c)
F    | type = “equilateral”
G    | if ( a >= b + c || b > a + c || c > = a + b)
H    | type = “not a triangle”
I    | if ( a <= 0 || b <= 0 || c <=0)
J    | type = “bad inputs”
K    | print type

* **Minimal** number of test cases for the statement coverage: **??**
Statement Coverage

Example – Euclid’s algorithm to compute GCD

A  read (x);
B  read (y);
C  while x ≠ y loop
D  if x > y then
   E       x := x – y;
F  else
G       y := y – x;
H  end loop
I  gcd := x;

Find minimal test cases for the statement coverage.
Statement Coverage

if x < 0 then
    x := -x;
end;
z := log(x);
Statement Coverage

• Drawbacks
  – Some control transfers may be missed
  – Statement coverage does not adequately take into account the fact that many lines of code (and many bugs) involve branching and decision-making.
    • Simple if statements
    • Logical operators
    • Consecutive switch labels
    • Loop termination decisions
    • Do-while loops
Branch Coverage

• Select test cases such that every edge in the graph is visited
  – Guarantees that every branch in the source code is executed at least once
• More thorough than statement coverage
  – More likely to reveal logical errors
• However, even if all branches are exercised, this does not mean that all combinations of control transfers are checked
Branch Coverage

Node | Source Line
---|---
A  | read a, b, c
B  | type = “scalene”
C  | if (a==b || b==c || a==c)
D  | type = “isosceles”
E  | if (a == b && b == c)
F  | type = “equilateral”
G  | if ( a >= b + c || b > a + c || c >= a + b)
H  | type = “not a triangle”
I  | if ( a <= 0 || b <= 0 || c <=0)
J  | type = “bad inputs”
K  | print type

* **Minimal** number of test cases
  for the branch coverage: ??
Branch Coverage

Example – Euclid’s algorithm to compute GCD

A  read (x);
B  read (y);
C  while x \neq y loop
D  if x > y then
E       x := x – y;
F  else
G      y := y – x;
H  end loop
I  gcd := x;

Find minimal test cases for the branch coverage.