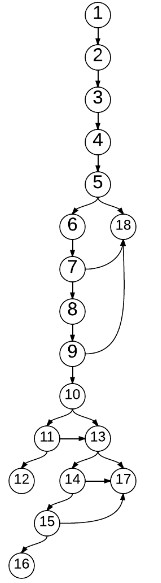
**Java Code**

1. public int test(int a, int b, int c){
2. int s;
3. boolean IsTriangle;
4. if ((a<b+c)&&(b<a+c)&&(c<a+b)){
5. IsTriangle = true;
6. }
7. else{
8. IsTriangle = false;
9. }
10. if (IsTriangle){
11. if((a==b)&&(b==c)){
12. s = 1;
13. }
14. else if((a!=b)&&(a!=c)&&(b!=c)){
15. s = 3;
16. }
17. else{
18. s = 2;
19. }
20. }
21. else{
22. s = 4;
23. }
24. return s;
25. }

**Jimple Code**

1. i0 := @parameter0: int
2. i1 := @parameter1: int
3. i2 := @parameter2: int
4. $i3 = i1 + i2
5. if i0 >= $i3 goto return 4
6. $i4 = i0 + i2
7. if i1 >= $i4 goto return 4
8. $i5 = i0 + i1
9. if i2 >= $i5 goto return 4
10. if i0 != i1 goto (branch)
11. if i1 != i2 goto (branch)
12. return 1
13. if i0 == i1 goto return 2
14. if i0 == i2 goto return 2
15. if i1 == i2 goto return 2
16. return 3
17. return 2
18. return 4

**Condition Extraction**

|  |  |  |
| --- | --- | --- |
| Node No. | Jimple Statement | Condition |
| 5 | i0 > i3; i3 = i1 + i2 | a >= b + c |
| 7 | i1 > i4; i4 = i0 + i2 | b >= a + c |
| 9 | i2 > i5; i5 = i0 + i1 | c >= a + b |
| 10 | i0 != i1 | a != b |
| 11 | i1 != i2 | b != c |
| 13 | i0 == i1 | a = b |
| 14 | i0 == i2 | a = c |
| 15 | i1 == i2 | b = c |

**All Path Coverage**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Covered nodes | Input | Output |
| P1 | 1,2,3,4,5,18 | 17 5 8 | 4 |
| P2 | 1,2,3,4,5,6,7,18 | 4 9 3 | 4 |
| P3 | 1,2,3,4,5,6,7,8,9,18 | 8 11 22 | 4 |
| P4 | 1,2,3,4,5,6,7,8,9,10,11,12 | 22 22 22 | 1 |
| P5 | 1,2,3,4,5,6,7,8,9,10,11,13,17 | 6 6 4 | 2 |
| P6 | 1,2,3,4,5,6,7,8,9,10,11,13,14,17 | / | / |
| P7 | 1,2,3,4,5,6,7,8,9,10,11,13,14,15,17 | / | / |
| P8 | 1,2,3,4,5,6,7,8,9,10,11,13,14,15,16 | / | / |
| P9 | 1,2,3,4,5,6,7,8,9,10,13,17 | / | / |
| P10 | 1,2,3,4,5,6,7,8,9,10,13,14,17 | 17 23 17 | 2 |
| P11 | 1,2,3,4,5,6,7,8,9,10,13,14,15,17 | 9 16 16 | 2 |
| P12 | 1,2,3,4,5,6,7,8,9,10,13,14,15,16 | 33 25 27 | 3 |

Branch coverage is closely related to decision coverage and at 100% coverage they give exactly the same results. Decision coverage measures the coverage of conditional branches; branch coverage measures the coverage of both conditional and unconditional branches.

*Unconditional branch definition: In programming, a GOTO, BRANCH or JUMP instruction that passes control to a different part of the program. Contrast with conditional branch.*

**Statement Coverage**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | Covered nodes | Input | Output |
| 1 | P1 | 1,2,3,4,5,18 | 17 5 8 | 4 |
| P2 | 1,2,3,4,5,6,7,8,9,10,11,12 | 22 22 22 | 1 |
| P3 | 1,2,3,4,5,6,7,8,9,10,11,13,17 | 6 6 4 | 2 |
| P4 | 1,2,3,4,5,6,7,8,9,10,13,14,15,16 | 33 25 27 | 3 |
| 2 | P1 | 1,2,3,4,5,6,7,18 | 3 11 7 | 4 |
| P2 | 1,2,3,4,5,6,7,8,9,10,11,12 | 30 30 30 | 1 |
| P3 | 1,2,3,4,5,6,7,8,9,10, 13,14,17 | 16 11 16 | 2 |
| P4 | 1,2,3,4,5,6,7,8,9,10,13,14,15,16 | 4 7 9 | 3 |
| 3 | P1 | 1,2,3,4,5,6,7,8,9,18 | 23 17 41 | 4 |
| P2 | 1,2,3,4,5,6,7,8,9,10,11,12 | 37 37 37 | 1 |
| P3 | 1,2,3,4,5,6,7,8,9,10,11,13,17 | 45 45 56 | 2 |
| P4 | 1,2,3,4,5,6,7,8,9,10,13,14,15,16 | 22 9 18 | 3 |
| 4 | P1 | 1,2,3,4,5,18 | 35 17 6 | 4 |
| P2 | 1,2,3,4,5,6,7,8,9,10,11,12 | 23 23 23 | 1 |
| P3 | 1,2,3,4,5,6,7,8,9,10,13,14,15,17 | 17 30 30 | 2 |
| P4 | 1,2,3,4,5,6,7,8,9,10,13,14,15,16 | 89 78 52 | 3 |
| 5 | P1 | 1,2,3,4,5,6,7,18 | 13 37 20 | 4 |
| P2 | 1,2,3,4,5,6,7,8,9,10,11,12 | 15 15 15 | 1 |
| P3 | 1,2,3,4,5,6,7,8,9,10,13,14,17 | 22 10 22 | 2 |
| P4 | 1,2,3,4,5,6,7,8,9,10,13,14,15,16 | 23 45 31 | 3 |
| 6 | P1 | 1,2,3,4,5,6,7,8,9,18 | 12 17 39 | 4 |
| P2 | 1,2,3,4,5,6,7,8,9,10,11,12 | 43 43 43 | 1 |
| P3 | 1,2,3,4,5,6,7,8,9,10,13,14,15,17 | 22 31 31 | 2 |
| P4 | 1,2,3,4,5,6,7,8,9,10,13,14,15,16 | 6 7 8 | 3 |