**FWHITE BOX TESTING**

1. **Scenario:** The average cost of an insurance premium for drivers is €300, however, this premium can increase or decrease depending on three factors: Age, gender and marital status. Drivers that are below the age of 25, male and single face an additional premium increase of €1000. If a driver outside of this bracket is married or female their premium reduces by €100, and if there are aged between 46 and 65 inclusive, their premium goes down by another €150.

Use the following program fragment for questions a-c below:

**private static int perdetails(int age, String gender, boolean married)** {

int premimum = 300; **1**

if((age < 25) && (gender.equals("male")) && (!married)){

**2 3 4**

premimum = premimum + 1000; **5**

}

else{

if (married || gender.equals("female")){

**6 7**

premimum = premimum - 100; **8**

if((age >= 46) && (age <= 65)){

**9 10**

premimum = premimum - 150; **11**

}

}

}

System.out.println("premimum" + premimum); **12**

return premimum;

}

1. Draw the CFG for **perdetails**().
2. From the CFG, identify independent paths to reach 100% statement coverage.
3. Identify additional paths, if necessary, to reach 100% branch coverage.
4. Design test cases based on CFG. Calculate the minimum number of  
   test cases if the goal is:  
   • 100% Statement coverage  
   • 100% Branch coverage

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Test Path** | **Inputs** | **EO** | **RO** | **Note** |
| tc1 | 1;2(T);3(T);4(T);5;12 | age=12; gender=”male”; married = false; | premimum: 1300 |  |  |
| tc2 | 1;2(F);6(F);7(T);8;9(T);10(T);11;12 | age=50; gender=”female”; married = false; | premimum: 50 |  |  |

2) Use the following program fragment for questions a-c below:

int Test(int a, int b, int c) {

int t; **1**

switch (a) **2**{

case 2: **3**

if (b>9)**6**

t=1; **7**

else

t=8; **8**

break;

case 7: **4**

if (b<0 || b>10)

**9** **10**

t=0; **11**

else

if (c>=3) **12**

t=3; **13**

else

t=4; **14**

break;

default:**5**

t=5; **15**

}

return t; **16**

}

1. Draw the CFG for **Test**().
2. From the CFG, identify independent paths to reach 100% statement coverage.
3. Identify additional paths, if necessary, to reach 100% predicate coverage.
4. Design test cases based on CFG. Calculate the minimum number of  
   test cases if the goal is:  
   • 100% Statement coverage  
   • 100% Predicate coverage

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Test Path** | **Inputs** | **EO** | **RO** | **Note** |
| tc1 | 1;2;5;7;18 | a=1;b=2;c=3 | t=5 |  |  |
| tc2 | 1;2;3;6(T);7;9;18 | a=2;b=10;c=10 | t=1 |  |  |
| tc3 | 1;2;4;10(T);12;16;18 | a=7;b=-1;c=10 | t=0 |  |  |
| tc4 | 1;2;4;10(F);11(F);13;14;16;18 | a=7;b=5;c=4 | t=3 |  |  |

**3)** Use the following program fragment for questions a-c below:  
void soNT(int n) {

bool co=true **1**  
for(int i=2;i<n;i++){

**1** **2** **3**

if(n%i==0){ **4**

co=false; **5**

break;

}

}

if(n != 1 && co == true)

**6** **7**

Console.WriteLine("{0} là số nguyên tố ",n); **8**

else

Console.WriteLine("{0} không phải là nguyên tố",n); **9**

}

1. Draw the CFG for **soNT**().
2. From the CFG, identify independent paths to reach 100% statement coverage.
3. Identify additional paths, if necessary, to reach 100% branch coverage.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Test Path** | **Inputs** | **EO** | **RO** | **Note** |
| tc1 | 1;2(F);6(T);7(T);8 |  |  |  | tc2 |
| tc2 | 1;2(T);4(F);3;2(T);4(F);3;2(T);4(F);3;2(F);6(T);7(T);8 | n=5 | 5 là số nguyên tố |  |  |
| tc3 | 1;2(T);4(T);5;6(T);7(F);9 | n=4 | 4 không phải là số nguyên tố |  |  |
| tc4 | 1;2(F);6(F);9 | n=1 | 1 không phải là số nguyên tố |  |  |

1. Design test cases based on CFG. Calculate the minimum number of  
   test cases if the goal is:  
   • 100% Statement coverage  
   • 100% Branch coverage