EE4001/IM2001 Software Engineering
Tutorial 7 and Sample Answer

1. Describe the key differences between black-box testing and white-box testing.

Q1 Answer

The key differences:

(1) Black-box testing tests the functionality and external characteristics of software but white-box testing tests the internal characteristics of software.

(2) Black-box testing is based on requirements and specification to test software but white-box testing is based on the logic structure and internal details of code to test software.

2. Draw a control flow graph for the program shown in Figure 1 (specified in pseudocode), identify a basis set of linearly independent paths through the program and use the basis path testing technique to design a set of test cases. Note that the input to this program is x and y.

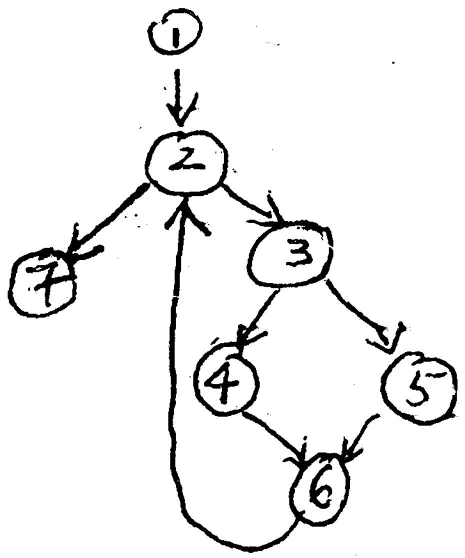
```
begin
    read (x);
    read (y);
    while x ≠ y loop
        if x > y then
            x := x – y;
        else
            y := y – x;
        end if;
     end loop;
     gcd := x;
     print (gcd)
end;
```

Figure 1. The pseudocode of a program

Q2 Answer

The control flow graph

```
(1) begin
        read (x);
        read (y);
    (2) while x ≠ y loop
        (3) if x > y then
        (4)     x := x - y;
            else
        (5)     y := y - x;
            end if;
    (6) end loop;
(7)     gcd := x;
    end; print(gcd);
```



As V(G) = E – N + 2 = 8 – 7 + 2 = 3 for this control flow graph, there are 3 paths in a basis set of linearly independent paths.

A basis set of linearly independent paths is as follows:

{(1, 2, 7)
  (1, 2, 3, 5, 6, 2, 7),
  (1, 2, 3, 4, 6, 2, 7)}

A set of test cases for Basis Path Testing is as follows:

Case 1: <x = 3, y = 3>, this is to force through the first path with expected output gcd = 3.
Case 2: <x = 3, y = 6>, this is to force through the second path with expected output gcd = 3.
Case 3: <x = 6, y = 3>, this is to force through the third path with expected output gcd = 3.

3. For the program shown in Figure 1 (specified in pseudocode), use both the statement and branch coverage testing techniques to design a set of test cases each. For a test suite that has only one test case, predict the possible of branch coverage in term of percentages.

Q3 Answer

I)
A set of paths that pass through all the nodes in the CFG:

{(1, 2, 3, 5, 6, 2, 7), (1, 2, 3, 4, 6, 2, 7)}

Hence, the following is a test suite for Statement Coverage Testing:

Case 1: Input: <x = 3, y = 6>; Expected Result: gcd = 3.
Case 2: Input: <x = 6, y = 3>; Expected Result: gcd = 3.

II)
The above set of paths also passes through all the branches in the CFG:
Altogether, there are four branches (2, 3), (2, 7), (3, 4) and (3, 5) in the CFG.
The first path passes through the branch from node 3 to node 5 and all the branches at node 2.
The second path passes through the branch from node 3 to node 4 and all the branches at node 2.
Hence, the set of path passes through all the branches in the CFG.

Hence, the above test suite is also a test suite for Branch Coverage Testing.

III)
Altogether, there are **4** branches in the CFG of the program shown in Figure 1.
A test suite that has only one test case exercises **only one path**.
All the paths through the CFG can be classified into the following types according to the number of branches that they pass through with the branch coverage shown:

a) Passing through only one branch: For example, the path (1, 2, 7). In this case, the branch coverage = 1/4 = 25%

b) Passing through three branches: For example, the path (1, 2, 3, 4, 6, 2, 7), (1, 2, 3, 4, 6, 2, 3, 4, 6, 2, 7), etc…. In this case, the branch coverage = 3/4 = 75%.

c) Passing through all the four branches: For example, (1, 2, 3, 4, 6, 2, 3, 5, 6, 2, 7), (1, 2, 3, 4, 6, 2, 3, 5, 6, 2, 3, 4, 6, 2, 7), etc… In this case, the branch coverage = 4/4 = 100% .

Therefore, the possible branch coverage for a test suite that has only one test case must be 25%, 75% or 100%.

4. (a) Draw a control flow graph for the program specified in Figure 2 (in pseudocode). And, use the basis path testing technique to design test cases to test the program.

```
read (i, y, g)
a := 1
b := 1
while (i < 100) do
        c := i + 1
        z := f()
        if (y < 0) then
           a := 2
           b := 2
        else
           a := 3
           b := 3
        endif
        if (g = 0) then
          a := a + 1
          b := b + 1
        endif
        i  := i + 1
        y := y + c
endwhile
print(g, a, b)
```

Figure 2. The pseudocode of a program

(b) In the control flow graph drawn by a student, the following first six statements in Figure 2 are represented by one node:

```
read (i, y, g)
a := 1
b := 1
while (i < 100) do
        c := i + 1
        z := f()
```

Is the control flow graph correct? Please justify your answer clearly.

Q4 Answer

(a)

With the basic blocks labelled below, the control flow graph is as follows:

```
(1) read (i, y, g)
      a := 1
      b := 1
(2) while (i < 100) do
(3) ——      c := i + 1
            z := f()
            if (y < 0)
(4) ———        a := 2
               b := 2
            else
(5) ———        a := 3
               b := 3
            endif
(6) ——      if(g ≠ 0)
(7) ———       a := a + 1
              b := b + 1
            endif
(8) ——      i  := i + 1
            y := y + c
      endwhile
(9) print(g, a, b)
```

$$\begin{tikzcd}
 & \textcircled{1} \arrow[d] & \\
 & \textcircled{2} \arrow[d] \arrow[ddddddd, bend right=60] & \\
 & \textcircled{3} \arrow[dl] \arrow[dr] & \\
\textcircled{4} \arrow[dr] & & \textcircled{5} \arrow[dl] \\
 & \textcircled{6} \arrow[d] \arrow[dd, bend right] & \\
 & \textcircled{7} \arrow[d] & \\
 & \textcircled{8} \arrow[uuuuuu, bend right=70] & \\
 & \textcircled{9} &
\end{tikzcd}$$

As V(G) = E – N + 2 = 11 – 9 + 2 = 4 for this control flow graph, there are 4 paths in a basis set of linearly independent paths. These paths and their respective test cases are as follows:

Path 1

(1, 2, 9)
Test case: i =101, y = 0, g = 0.
Expected result: g = 0, a = 1, b = 1.

Path 2

(1, 2, 3, 4, 6, 7, 8, 2, 9)
Test case: i = 99, y = -1, g = 0.
Expected result: g = 0, a = 3, b = 3.

Path 3

(1, 2, 3, 5, 6, 7, 8, 2, 9)
Test case: i = 99, y = 0, g = 0.
Expected result: g = 0, a = 4, b = 4.

Path 4

(1, 2, 3, 5, 6, 8, 2, 9)
Test case: i = 99, y = 0, g = 1.
Expected result: g = 1, a = 3, b = 3.

(b) The control flow graph must be incorrect. This is due to the fact during program execution, the given sequence of statements may execute from the first statement or the while statement. Likewise, it may end at the while statement or the last statement in the sequence. As such, it does not always execute from the first statement to the last statement. Therefore, this sequence of statements cannot be combined together and represented as a node.

5. Figure 3 shows a control flow graph (CFG) of a program without showing its code. Based on this CFG, compute the following:

i) The total number of paths through the CFG for the following three cases:
Case 1: do not go through the loop in the CFG at all.
Case 2: do not repeat the loop (i.e., go through the loop at most one time).
Case 3: repeat the loop at most one time (i.e., go through the loop at most two times).

ii) The possible branch coverage for a test suite that has only one test case.

iii) Without considering path feasibility, the number of test cases in a minimal test suite designed from statement coverage testing technique.

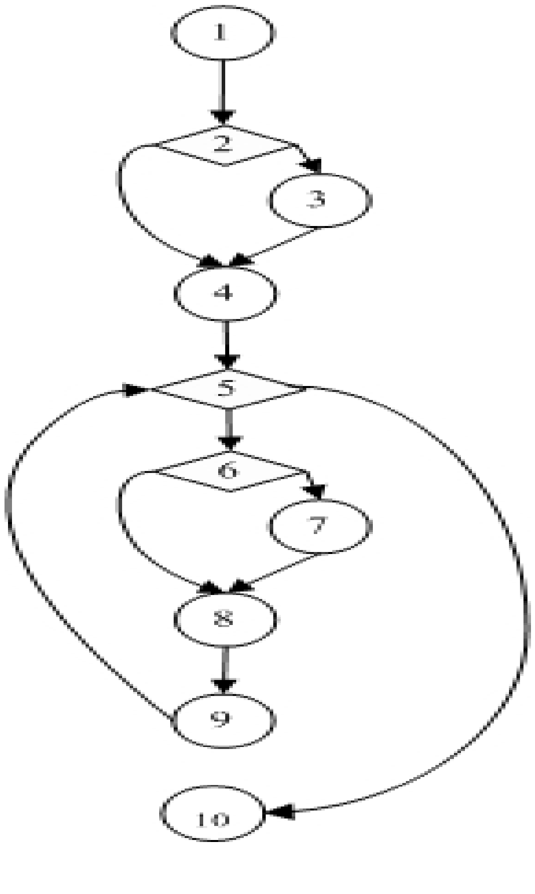


Figure 3. A control flow graph

Q5 Answer

(i)

Total no of paths in case 1 = 2.

Total no paths in case 2 = total no of paths in case 1 + 2 * 2= 2 + 2 * 2 = 6

Total no of paths in case 3 = total no of paths in case 2 + 2 * 2 * 2 = 6 + 8 = 14.

(ii)

The paths in the CFG can be classified into the following types in term of the branches they pass through:

Type 1: pass through one branch at node 2 and exit the loop at node 5 without going through the loop.
Type 2: pass through one branch at node 2, go through the loop at node 5 and pass through one branch at node 6.
Type 3: pass through one branch at node 2, go through the loop at node 5 and pass through both branches at node 6.

Branch coverage for each type are as follows:

Branch coverage for type 1= 2/6 = 33.33%
Branch coverage for type 1= 4/6 = 66.66%
Branch coverage for type 1= 5/6 = 83.33%

(iii)

The minimum no of paths in a set of paths through the CFG to pass through all the nodes is, 1, such as the set {(1, 2, 3, 4. 5, 6, 7, 8, 9, 5, 10)} of paths.
Therefore, without considering path feasibility, the number of test cases in a minimal test suite designed from statement coverage testing technique = 1.