SE 4367, HW #2 Coverage

T = {t_1=<4>, t_2=<25>, t_3=<-1>}
or T = {t_1=<A=4>, t_2=<A=25>, t_3=<A=-1>}

1) **What is the statement domain for P1? Express as line numbers. Exclude syntactical markers, such as {, }, else, and end.**
2) **What is the statement coverage of T for P1? Express as a fraction.**
3) **If the statement coverage of T for P1 is less than 100%, what test cases do you need to add to get 100% statement coverage? (Many possible correct answers.)**
4) **What is the decision domain for P1? Express as "line number) decision".**
5) **What is the decision coverage of T for P1? Express as a fraction.**
6) **If the decision coverage of T for P1 is less than 100%, what test cases do you need to add to get 100% decision coverage? (Many possible correct answers.)**
7) **What is the condition domain for P1? Express as "line number) condition".**
8) **What is the condition coverage of T for P1? Express as a fraction.**
9) **If the condition coverage of T for P1 is less than 100%, what test cases do you need to add to get 100% condition coverage? (Many possible correct answers.)**

```
 Program P1
1) integer A, B;
2) input (A);
3) while (A > 0)
4) {
5)       A = 2 * A;
6)       if (A < 20 or A > 30)
7)            B = A * 2;
8)       else
9)            B = A + 2;
10)      output (A, B);
11)      input (A);
12)}
13)output (“Program ends.”);
14)end;
```

2.1 Statement Domain

If you count physical lines of code, there are 14 LOC.

Excluding syntactical markers {, }, else, end (as required)…

D_S = {1, 2, 3, 5, 6, 7, 9, 10, 11, 13}

Also… $|D_S|$ = 10

```
 Program P1
1) integer A, B;
2) input (A);
3) while (A > 0)
4) {
5)       A = 2 * A;
6)       if (A < 20 or A > 30)
7)            B = A * 2;
8)       else
9)            B = A + 2;
10)      output (A, B);
11)      input (A);
12)}
13)output ("Program ends.");
14)end;
```

2.2 Statement Coverage

T = {t_1=<4>, t_2=<25>, t_3=<-1>}
D_S = {1, 2, 3, 5, 6, 7, 9, 10, 11, 13}

t_1 covers statements 1, 2, 3, 5, 6, 7, 10, 11

t_2 covers statements 11, 3, 5, 6, 7, 10, 11

t_3 covers statements 11, 3, 13, exit

Line 9 is not covered by T

Coverage is 9 / 10 = 90%

- **t_1: {1, 2, 3, 5, 6, 7, 9, 10, 11, 13}**
- **t_2: {1, 2, 3, 5, 6, 7, 9, 10, 11, 13}**
- **t_3: {1, 2, 3, 5, 6, 7, 9, 10, 11, 13}**

2.3 Completing Statement Coverage

For 100% statement coverage, need line 6 to be false for a test case.

Implies 2*A<20 is false and 2*A>30 is false.

t_4: A=12 → 2*12<20 is false AND 2*12>30 is false.

Adding t_4: A=12 will provide 100% statement coverage… as will any other input that meets the conditions above.

- **Note that there are many possible correct answers for what test cases can be added to provide adequate test coverage against the given criterion (statement, decision, or condition coverage).**

Note on T

Note that for

T^S = {t_1=<4>, t_2=<25>, t_4=<12>, t_3=<-1>}

it matters where you put t_4.

If you put t_4 after t_3, the program exits without ever reading in the 12.

If you just said t_4=<12>, we will assume you put it in the right place (the question was what test case would you add).

If you listed the test set explicitly, you need to put it in the right sequence (anywhere before t_3).

2.4 Decision Domain

Decision domain D_D =

{

3) while (A>0)

6) if (A<20 or A>30)

}

2 decisions

$|D_D| = 2$

2.5 Decision Coverage

3) while (A>0)

6) if (A<20 or A>30)

$|D_D| = 2$

t_1 → 4>0… 3 is true
→ 8<20 or 8>30… 6 is true

t_2 → 25>0… 3 is true
→ 50<20 or 50>30… 6 is true

t_3 → -1>0… 3 is false

3 is covered (t_1,t_2 + t_3)
6 is not covered (t_1,t_2 + ?)

Decision coverage is 1 / 2 = 50%

2.6 Completing Decision Coverage

For 100% decision coverage, need the decision at line 6 to be false for a test case.

2*A<20 is false and 2*A>30 is false.

- **A≥10 and A≤15**

t_4: A=12 → 2*12<20 is false → 2*12>30 is false.

Adding t_4: A=12 will provide 100% decision coverage… as will any other input that meets the conditions above.

Note on Statement vs Decision Coverage

Does statement coverage subsume decision coverage?

Consider the code fragment

```
z=x;
if (x<0)
        y=z;
z=2;
```

Does decision coverage subsume statement coverage?

- **Yes and we'll talk about this later…**

2.7 Condition Domain

Condition domain D_C =

{

3) A>0

6) A<20

6) A>30

}

3 conditions

$|D_C| = 3$

2.8 Condition Coverage

3) A>0

6) A<20

6) A>30

$|D_C| = 3$

t_1 → 4>0… 3)A>0 is true
→ 8<20… 6)A<20 is true
→ 8>30… 6)A>30 is false

t_2 → 25>0… 3)A>0 is true
→ 50<20… 6)A<20 is false
→ 50>30… 6)A>30 is true

t_3 → -1>0… 3)A>0 is false

3) A>0 is covered (t_1,t_2 + t_3)
6) A<20 is covered (t_1 + t_2)
6) A>30 is covered (t_2 + t_1)

Condition coverage is 3 / 3 = 100%

2.9 Completing Condition Coverage

100% condition coverage therefore no need to add test cases to the test set.

This assignment demonstrates having 100% condition coverage without having 100% decision coverage.

It is also easy to see how to have100% decision coverage without having 100% condition coverage.

Notes

Note that you have to decide whether to execute program P1 for each test case or use the input loop for test cases after the first input.

- **Sometimes it makes a difference, e.g., when there are initialization or dependencies between input sequences – we will talk about this later.**
- **Using the input loop is usually the right thing to do unless you have a reason not to.**

Note that we left out some complicating factors in these coverage criteria – like dead code, infeasible decisions, and short-circuit evaluation.

Short-Circuit Evaluation

If someone assumed that t_1 would not traverse A>30 because of short-circuit evaluation, only take 5 points off

- **some languages, such as C, require the compiler to do short-circuit evaluation**
  - **this is language & compiler specific**
- **in such a case, A<20 has to be false before A>30 is considered in OR short-circuit evaluation**
- **for part 2.9, t_4=<12> would provide f+f coverage to go with t_2 f+t coverage of A<20 OR A>30**
  - **if you assume short-circuit evaluation, you need to create test cases appropriately…**

Grading Rubric

Parts 1, 4, and 7 are worth 5 points each.

Parts 2, 5, and 8 are worth 20 points each.

Parts 3, 6, and 9 are worth 8 points each.

Coverage is usually measured as a percent, but a ratio (fraction) is easier to calculate and grade.

- **The denominator is a count of the number of statements, decisions, or conditions within the domain. It is worth 10 points.**
- **The numerator addresses the domain elements covered. It is worth 10 points.**

In parts 1, 4, and 7, you're deciding what the denominator for parts 2, 5, and 8 should be…

Formatting Submissions

In the file name, include:

- **class**
- **assignment identifier**
- **your name (or team's name)**
  - **e.g., se4367a01jdoe**

In the file (or hardcopy) submitted, include the class, assignment, and name information at the top.

Minus 5 points per violation. Potentially 30 points off for formatting mistakes!