5321 Homework 2

Summer 2020

Guidance for all problems:

1. Develop the minimum set of test cases needed to adequately test all actions and boundary values.
2. List all inputs and expected outputs using the test case table depicted in the previous HW 1 solution. Only show essential inputs and outputs - do not show intermediate values in your test case tables.
3. Develop the test needed for all partitions with two test cases per partition.
4. Assume that currency is truncated to the cent.
5. State any assumptions made, but do not change the function of the problem.
6. Values shown must have thousands separators and reflect the units being depicted. Significance shown must reflect the units being depicted.

The following are customer descriptions of software scenarios. For each problem state any assumptions made - but **do** **not** **change** the function being described.

**Problem 1.** Develop the test case table and the sequence enumeration table for the Binocular system in HW 1 Problem 5

1. Test case table
   1. Capture all inputs, outputs and states as described for the state diagram above.
      1. Each input will have a column under inputs and each output will have a column under expected outputs.
      2. Input values will be either 0 or 1
      3. For outputs show the specific values of outputs, e.g. X=2 will show in the table column X in expected outputs with a 2 in the cell for that test case.
      4. Test Case 1 is the start state - current state=Start and next state=OFF
      5. Each row only sets one input to a 1 value at a time. Where "D, G" is a possible transition use two rows with each set to a 1 value at a time.
   2. Test cases table will have the following columns: Test case number, Current state, Next state, Inputs, Expected Outputs.
2. Develop the sequence enumeration.
3. Submit the Sequence enumeration table
4. List the canonical states - in your pdf or word solution, not the table

Sequence enumeration table.

1. Use sequence enumeration to develop the canonical sequences. Show all sequences from length 0 to N.
2. Capture these in the attached table
3. For the "**Carry to next level**" column in the spreadsheet - use "Yes" or leave blank (for no).
4. Show all outputs for each - there will be no null responses
5. Mark each non-equivalence with a "-" enter in Excel as '-'

**Hint**: equivalences should only be either "idle", "-", or some previous **sequence** (e.g., PZ). **Please do not use a state as an equivalence** (e.g., X5).

**Submission checklist**

Test case table template follows



Sequence enumeration table template follows:



Include the following in your word or PDF file.

Canonical sequences are: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Problem 2**. You have an 3 input parameters to a method that is used to compute the call charge for an Mobile phone billing center (**Time of day**, **Call length**, **call type**).

**Time of day** is specified as the following 12-hour value - 12:00:00 am and is significant to the second.

**Call length** is in minutes truncated to the whole number (integer).

**Call type** is essentially enumeration of {Domestic, International, Conference}.

Additionally, the unit test plan requires the following values to be tested for call length (minutes): 1, 5, 10, 15, 61.

So, putting this together, the unit test plan calls for the 3 inputs have the following values (as shown below).

Calls are billed based on their time of day as follows:

|  |  |
| --- | --- |
| **Rate Table** | **Rate/min** |
| >=Midnight & <8am | $0.08 |
| 8am-7pm | $0.25 |
| >7pm & <10pm | $0.15 |
| >=10 pm & <Midnight | $0.10 |

The call charge (method return value) is calculated as follows:

Call charge = Rate/min \* call\_length \* call\_type\_multiplier

Call\_type\_multiplier is computed as the following:

|  |  |
| --- | --- |
| **Call Type Multiplier** | **Rate/min** |
| Domestic | 1 |
| International | 3 |
| Conference | 5 |

Develop the test to fully test all pair-wise combinations of Time of day, Call length, call type testing all boundary conditions.

**As a hint**, since we need to test each boundary value the following table shows the possible sets of **values** for the three parameters (THIS IS NOT A TEST CASE TABLE - IT SIMPLY SHOWS THE POSSIBLE SETS OF VALUES FOR EACH).

|  |  |  |
| --- | --- | --- |
| **Time of Day** | **Call Length** | **Call Type** |
| 12:00:00 AM | 1 | International |
| 7:59:59 AM | 5 | Domestic |
| 8:00:00 AM | 10 | Conference |
| 7:00:00 PM | 15 |  |
| 7:00:01 PM | 61 |  |
| 9:59:59 PM |  |  |
| 10:00:00 PM |  |  |
| 11:59:59 PM |  |  |

Additional information:

Assume Midnight==12:00:00am

Your test case table is as follows.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case**  **No.** | **Inputs** | | | **Exp Output** |
| **Time** | **Call Length** | **Call type** | **Cost** |

Instructions

1. Download the allpairs tool from the following site <http://www.satisfice.com/tools/pairs.zip> and follow the instructions.
2. Use the previous table (test case table) to develop the test cases needed to test all pairs.
3. Supply item 2 as a single test case table in Excel. You do NOT need to show the output of the allpairs tool - just the test cases it generates and in the order it provides.

**PLEASE MAKE SURE TO SAVE A COPY OF THE TEST CASE TABLE ABOVE AS A TAB DELIMITED TXT FILE. In Excel -> Save As... -> tab delimited txt file. This will allow the GTAs to use WinMerge to compare your test case table with the output. 50% deduction if not supplied.**

**Problem 3.**

Minimize the following expressions using a K-map. Show all work including the K-map.

1. a'b'c' + abc + abc'd' + a'b'cd' + a'bc'd + a'bcd + ab'c'd
2. a'b'c' + a'cd' + ac'd' + ab'c + a'b'cd + abcd' + ab'c'd + a'bc'd'
3. cd + a'b'c'd' + a'b'c'd + ab'cd + a'bcd+ a'b'cd' + ab'c'd' + a'bc'd' + ab
4. a'b'c'd + abcd + a'b'cd + abc'd + a'bc'd + ab'cd + a'bcd + ab'c'd

**Problem 4.**

For each of the following expressions develop the following below. Make sure to reduce each to the minimum logical expression. Reduce all answers.

1. a+b'c'
2. a(c + d)
3. ab + c' + d
4. ab XOR (a + b)
5. The condition coverage, decision coverage, condition/decision coverage terms (one pair per coverage). Write solutions in terms of n-tuples - (FFF, FFT) as appropriate. Clearly indicate your answers for each. **FOR DECISION COVERAGE USE THE FIRST TERM AS FFF or FFFF**
6. The TOFs (Term Omission Faults) and TNFs (Term Negation Faults) for each. Separate each possible answer by a comma.

**Problem 5.**

1. Provide a UC MCDC solutions for each expression.
2. ab' + c
3. a' + b + c
4. a + bc + d'
5. Provide the following for each of the following expressions:

* A single condition/decision, condition, and decision pair for each
* All TNFs and TOFs for each

1. abc XOR (ab'c')'
2. ((a+b)(b'+c))'
3. (ab' + ac + c'd)'
4. Provide all UC MCDC solutions for each expression.
5. (a'b'c' + a'b'd')'
6. (a'c' + a'd' + a'b)'
7. (abd + acd)'

**Bonus Point questions**

Question b1 and b2 are worth 10 extra bonus points each

b1) Develop the MC/DC solutions for the following expressions - 2 UC solutions and 1 Masking solution (that is not a UC solution). Show which are Masking and which are Unique Cause.

1. ab' + c' + d'
2. a + bc + d'

b2) Use MC/DC logic and BV testing to determine the minimum test cases for each of the following requirements expressions. For each part, develop a test case table showing test case number, inputs, and expected outputs using the table as shown below.

1. a = (b< 0) || c
2. a = b && (c<5)
3. a = (b >5) && (c<8)
4. a = (b>5) && (b<10)

Express inputs in terms of numbers (for conditions with logical operators) and Booleans (for logical conditions) - e.g. the inputs are b (int) and/or c (int) when integer expressions are used, otherwise the inputs are Boolean.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Inputs** | | **Expected Outputs** |
| **Test Case** | **b** | **c** | **a** |
| 1 |  |  |  |
| 2 |  |  |  |
| 3 |  |  |  |