Week 6 exercises – model answers

1. Is Multiple Condition Coverage appropriate?

Several possible very thorough answers, worth 5/5:

No, it is not. In this case, the subject under test is the whole system, which consists of ten secret doors, and we will assume for simplicity 4 clauses in the predicates that control them, for a total of 40 clauses. If we aim to do thorough system testing, and each is to be set to "true" or "false", that gives 2^{40} = on the order of 10^{12} different tests – far too many to be realistic. Even executing a thousand tests a second, this would take over 34 years to run a system test.

Some form of *Active Clause Coverage* is probably a more reasonable criterion. If we again assume 40 clauses in the system in total, then for each of these, we will have to find values of *other* clauses which make the clause we are considering "active"; and we will need a positive and negative test case. So this gives us around 80 tests to write. (Which is quite a few, but not unreasonable for a whole system; besides which, the contract is probably lucrative, and the client likely to be especially unhappy with failures, so it would be a good idea to be thorough).

If we thought there were some parts of the system that were especially high risk, we could perhaps test those more thoroughly – perhaps using Multiple Condition Coverage just for those predicates, giving 2^4 or 16 tests per predicate.

Or, alternatively (though the assumptions are less plausible):

If we can assume that the different doors are guaranteed to be completely isolated, then we could apply combinatorial testing to each one individually, and count that as our "system test". A 3-clause predicate would mean 8 tests, a 4-clause predicate 16 tests, and a 5-clause predicate 32.

By way of example, let's assume 3 of each and one more 3-clause: that gives us 4 3-clause predicates $\times 8$ + 3 4-clause predicates $\times 16$ and 3 5-clause predicates $\times 32$ $= 32 + 48 + 92 = 176$ tests. In the worst case, if all were 5-clause predicates, we'd have 320 tests.

[assumption 1 for an answer]:
If we can assume that a sufficient system test can be done just by setting particular hardware signals to true or false (rather than someone manually checking each one), then 320 tests for a whole system (and a possibly irascible client) doesn't seem unreasonable.

[alternative, and probably better, assumption for an answer]:
If we can assume that a thorough system test involves manually operating equipment

for each door – which seems reasonable – then 320 tests is quite a few, but might still be worthwhile, for a sufficiently lucrative contract.

[Either way, you can then argue]:
However, it would be a more efficient use of time to use Active Clause Coverage, since some of the tests in combinatorial coverage won't "make a difference" – e.g. if we have a predicate of the form $A \wedge B \wedge C$), and A is false, then it doesn't matter what we set B and C to, and those tests are likely redundant.

The following answer doesn't use all the relevant facts/assumptions (size of system, type of industry/client), and doesn't bother to work out how many tests are needed. (Since every justification in software engineering is ultimately in terms of what correctness, or cost, or effort, if we have no idea how large those are, how can we say we've justified our result?)
-1 penalty to relevance, and -1 for justification, since the argument doesn't logically follow, for 3/5. (How can you say one unknown number is "infeasible", but another is "fine"?)
(An *efficiency* argument would be better – even if we don't know the numbers, MCC will always produce tests that are likely redundant.)

Multiple Condition Coverage would lead to a large number of tests, as the number increases exponentially with the number of clauses. On the other hand, Active Clause Coverage requires far fewer tests. So we should use Active Clause Coverage.

Marking:

Relevance:

- Identifies and estimates number of tests as being relevant, MCC and ACC as being relevant coverage criteria: 2m relevance.

  If mention is made of relevant facts like industry, project, client, the fact that hardware may need manual testing: even better, but still just 2m.
- Identifies MCC and ACC as relevant, quotes what the slides say about how MCC increase exponentially, but doesn't bother to calculate (or even estimate) numbers: 1m

Justification:

- Solid argument like one of the above: 2m.
- Fails to identify import assumptions, or makes some errors: 1m. e.g. just assumes implicitly that each door is guaranteed independent of all others. (Not a great assumption, hardware often affects other hardware in unpredictable ways, but if you make the assumption explicit, at least that shows some thought.)
- Poor logic in argument/calculation: 1m. e.g. Arguing that 320 tests is "utterly infeasible", yet 80 is perfectly fine. Why?
- Little or no evidence of logical reasoning: 0m. e.g. Suggests that MCC is fine, but that also (without justification) we should use ACC.

Clarity:

- 1m unless little or no evidence of ability to structure a clear answer.

2. Logic tests

Model answer for (i):

Let
i = inkwell is turned clockwise,
l = desk lamp is turned on,
d = top drawer is opened.

Then the logical expression is $i \vee (l \wedge d)$.

Possible answers for (ii).

If using MCC:

Two possible tests cases are:

Test description	**Test inputs**	**Expected output**
Test when all conditions true	$i = true$, $l = true$, $d = true$	Door opens
Test when all conditions false	$i = false$, $l = false$, $d = false$	Door does not open

OR, if using ACC:

Two possible tests cases are:

Test description	**Test inputs**	**Expected output**
i is active and made true	$i = true$, $l = false$, $d = false$	Door opens
i is active and made false	$i = false$, $l = false$, $d = false$	Door does not open

Marking:

- If more than two test cases were provided, I just looked at the first two.
- Relevance: Identifies relevant predicates and connectives - 2m.
  Little or no evidence of ability to identify relevant predicates: 0m. A 1 out of 2 mark isn't really relevant here.
- Justification/reasoning: Correctly gives two tests (inputs and expected results) - 2m.
  Some issues with tests = -1m penalty.
  Little or no evidence of ability to state a test: 0m.
- Clarity: 1m unless little evidence of ability to write a cogent answer.