CITS5501 Software Testing and Quality Assurance
Semester 1, 2020
Week 8 Workshop – Logic-based testing

Reading

It is strongly suggested you complete the recommended readings for weeks 1-7 *before* attempting this lab/workshop.

0. Notation

When writing logic expressions, we will normally use mathematical notation for “and”, “or”, and “not”:

- $\wedge$ – “and”
- $\vee$ – “or”
- $\neg$ – “not”

If writing actual Java code, however, we use the normal Java logical operators:

- `&&` – “and”
- `||` – “or”
- `!` – “not”

1. Identifying clauses

Ensure you understand the difference between *predicates* and *clauses*.
What are the *clauses* in the predicates below?

a. $((f \leq g) \wedge (x > 0)) \vee (M \wedge (e < d + c))$

b. $G \vee ((m > a) \vee (s \leq o + n)) \wedge U$

2. Making clauses active

Recall that to make a clause (call it c) *active*, we need to assign values to variables in a predicate such that the truth-value of the whole predicate depends on c.

So for instance, given the predicate $(x < 0) \vee P$; to make the first clause active, we must assign $P = true$.

For each of the clauses in the predicates below, identify test inputs which will make the clause *active* (that is: state what values need to be assigned to the variables in the predicate), explaining your reasoning.

a. $((f \leq g) \wedge (x > 0)) \vee (M \wedge (e < d + c))$
b. $G \vee ((m > a) \vee (s \leq o + n)) \wedge U$

3. Logic-based scenario – trap-doors

Suppose a component under test has the following requirement:

> If the lever is pulled and the chair is occupied, open the trap-door.
>
> If the button is pressed, open the trap-door.

Represent the component as a set of logic expressions. You should explain what each variable in your expressions mean. (For instance, something like "Let M represent whether the moon is full.")

You don't need to represent "open the trap-door"; that's not a logic expression, but rather, something that should happen when a logic expression is true.

4. Logic-based scenario – mice

Suppose we have the following requirement for some piece of software:

> "List the product codes for all wireless mice that either retail for more than \$100, or for which the store has more than 20 items. Also list non-wireless mice that retail for more than \$50."

This requirement contains a number of logic expressions – product codes will only be listed when particular conditions are satisfied.

For this scenario, it can be useful to define *functions* that operate on a product code. E.g., we might say, "Assume we have a function *price*(), which takes a product code as argument, and returns an integer representing the retail price for that product code in dollars."

Then we could represent one clause in the system as:

$$price(p) > 100$$

where p is some product code.

Write down an appropriate predicate representing the logical conditions in this requirement. Explain what variables or functions you're defining.