SFWRENG 3S03 Software Testing – Sketch Solutions
Midterm Test, Winter 2022
50 minutes + 40 minutes extra for technical disasters

1. Consider the following specification. The program reads three integer values from a dialog box. The three values represent the lengths of the sides of a triangle. The program displays a message that states whether the triangle is scalene, isosceles, or equilateral.

(a) [3 marks] Write three important functional tests.

- *Test for a scalene triangle*
- *Test for an equilateral triangle*
- *Test for an isosceles triangle*

(There are other tests, e.g., related to the dialog box, or type checking the inputs, but these are less important than the key functions of this program.)

(b) [3 marks] Write three important boundary tests.

- *Test behaviour for one or more negative inputs*
- *Test behaviour for one or more inputs at MAX_INT*
- *Test behaviour for one or more inputs at MIN_INT*

(There are other tests, e.g., all zero inputs, which are acceptable, but the key boundaries for integers are zero, MIN_ and MAX_INT and I expected to see coverage of these boundaries.)

(c) [3 marks] Write three ***additional*** tests that you might construct if you were doing exploratory testing.

There are lots and lots of valid answers here, and I was very generous in marking any sensible answers, e.g., testing invalid inputs (strings, real numbers, too few inputs).

(d) [1 mark] State one improvement you would make to this specification.

Probably the best answer is to require non-negative inputs. Requiring a specification of what to do when encountering erroneous inputs would require very substantial changes to the specification (lots of cases to enumerate) and would get into implementation details too.

2. Suppose you were asked to the test the main menu functionality of a new game for the Playstation 5. List 3 questions you would want to try to answer with your testing. Justify why each question is important to try to answer, in the context of testing this functionality.

(Note: ignore the fact that it is next to impossible to find a PS5.)

The main challenge with this question is identifying questions (or tests) that focus on the menu. Some questions that were submitted were more holistic, testing the system as a whole rather than the menu. Once you restrict your focus to the menu then a number of questions arise.

- *Is the menu easy to navigate? (perhaps do an observational study)*
- *Does each menu option operate the correct feature?*
- *Is it easy to find the right operation?*
- *Is each menu screen properly designed?*
- *Is it easy to recall how to repeat functions?*

Generally, questions that focused on navigation, understandability and recall were more targeted at the menu. I accepted a range of answers that generally addressed these points.

3. Consider the following specification of a method used as part of a system for managing loyalty points for customers in an online store.

```
public Status giveDiscount(long bonusPoints, boolean goldCustomer)

// bonusPoints: the number of bonus points accumulated by the customer

// goldCustomer: true for a gold customer

// return type: enum Status { full-price, discount, error };

{
  // Return Values:
  //   full-price: if bonusPoints <=120 and not goldCustomer
  //   full-price: if bonusPoints<=80 and goldCustomer
  //   discount: if bonusPoints>120
  //   discount: if bonusPoints >80 and goldCustomer
  //   error: if any inputs are invalid
}
```

a) [9 marks] State all equivalence partitions that you would use for testing this method.

The partitions I came up with are as follows:

- *For the parameter bonusPoints:*

    - *MIN_LONG…0*
    - *1..80*
    - *81..120*
    - *121..MAX_LONG*
- *For the parameter goldCustomer:*
    - *True*
    - *False*
- *For the Return Value*
    - *FULLPRICE*
    - *DISCOUNT*
    - *ERROR*

b) [9 marks] State a plausible test for each equivalence partition from (a). You do not need to write the test out in jUnit format; just specify a property (e.g., x>=0 and y<=5).

- *For the parameter bonusPoints:*
    - *(MIN_LONG <= bonusPoints) && (bonusPoints<=0)*
    - *(1<=bonusPoints) && (bonusPoints<=80)*
    - *(81<=bonusPoints)&&(bonusPoints<=120)*
    - *(121<=bonusPoints)&&(bonusPoints<=MAX_LONG)*
- *For the parameter goldCustomer:*
    - *goldCustomer*
    - *!goldCustomer*
- *For the Return Value*
    - *Return Value==FULLPRICE*
    - *Return Value==DISCOUNT*
    - *Return Value==ERROR*

(Note, I also accepted concrete test case values.)

c) [2 marks] Name any two cats that you have met during Richard's lectures this term.

The most popular named cats were Asterix, Possum, Merry, Willoughby and Venetia. I do not have a cat named Java, but the suggestion did make me laugh so you got full marks ☺