COMP1531

8.2 - SDLC Testing - Property-based Testing

Testing: what's the problem?

- Good testing gives us strong guarantees about our software, but...
  - Tests take a lot of time to write
  - It's easy to miss edge cases (even with coverage checking)
  - These problems only get worse with larger and more complicated systems

Can the machines do it for us?

- To write a program that generates tests for us requires:
  - A means of generating test data
  - Knowing what behaviour is correct?
  - Some way for the computer to report test failures to us in an understandable way

Example: Bubblesort

```
1 def bubblesort(numbers):
2     numbers = numbers.copy()
3     for _ in range(len(numbers) - 1):
4         for i in range(len(numbers) - 1):
5             if numbers[i] > numbers[i+1]:
6                 numbers[i], numbers[i+1] = numbers[i+1], numbers[i]
7     return numbers
```

Property-based testing

- A method of testing where tests are defined as general properties (i.e. parameterised predicates)
- Test input is generated automatically by supplying a strategy for generating that input
- The testing framework runs the test many times to ensure the properties are true for each input
- In the event of a test failure, the framework will shrink the generated input to find the smallest value that still fails the test

Hypothesis

- Hypothesis is the name of property-based testing framework for python
- Can be installed via: **pip3 hypothesis**
- Parameterised tests are decorated with @given to supply strategies for generating test input
- See **bubblesort.py**

What properties to test?

- It's not always easy to find testable properties
- Can you think of one for the Zune bug example?
- Software designs with testable properties tend to be good designs...