Software-testing (CS258)

Lesson 1

In testing, we are interested in finding failures in software and fixing them. Here we have the test input, the software and the output (i.e. what do we expect to happen). When we run software testing, we have two possibilities:

- The output is "okay" -- we haven't learned a lot
- The output is "bad" -- we have found a bug in our software. It can be caused by:
    - Bug in S.U.T. (Software Under Test)
    - Bug in acceptability test
    - Bugs in specification
    - Bugs in OS, compiler, libraries, hardware

Equivalent tests
- A test mapping a single input to an output
    - Input should represent a "class of inputs", i.e. if the code executes correct for one input, it should work for all input in the class

Creating testable software
- Clean code
- Refactor
- Should always be able to describe what a module does and how it interacts with other code
- No swamp of global variables
- No pointer "soup"
- Modules should have unit tests

Assertions
Check for a property that must be true. Assertions are useful because:

- Make code self-checking, leading to more effective testing
- Make code fail early, i.e. closer to the bug
- Assign blame
- Document assumptions, preconditions, postconditions, invariants

Assertions are common in many large project, for example GCC has ~900 assertions (1 assertion per 110 lines of code). One can disabling assertions in production code, e.g. this is done if we use the optimization flag in Python. However, it is common to keep the assertions in production code. Note that assertions are not a substitute for error handling.

Domains and ranges
Domain is the set of possible inputs. We should tests programs with values sampled from their domain. Range is the set of possible outputs.

Types of testing

- White box
  - Tester knows a lot about the software internals
- Black box
  - The opposite
- Unit testing
  - Testing only a small part of the software
  - Usually white box
- Integration testing
  - Testing multiple software modules (already tested) together
- System testing
  - Does system as a whole work?
  - Usually black box testing
- Differential testing
  - Take same test input and deliver it to two different implementations of SUT
- Stress testing
  - Test systems above their normal usage level
- Random testing
  - Feeding random data into the software