Unit Testing

Object Oriented System Design

Otago Polytechnic
Dunedin, New Zealand

We are bad coders and we should feel bad

- Here in the BIT there are two things that we are gnerally bad at:
  1. Documentation
  2. Testing
- I don't have any great ideas about documentation. You just have to suck it up.
- Testing is a different matter. We must get better at testing. The good news is that there are plenty of tools and methods to help us.

Types of testing

- Unit testing
- Integration testing
- Functional testing
- Performance testing
- Penetration testing
- Acceptance testing

Things we do instead of testing

- We write a lot of code in one go - maybe the whole assignment.
- When things don’t work, we stare at the code and try to figure out why.
- We throw in some print statements.
- We use a debugger.
- We write throwaway examples to test things.
- In short we waste time.

What we want to do.

- We will write tests that cover all our code.
- We will cleanly seperate test modules from application code.
- We will write tests that *document* our code.
- We will write tests that run automatically.
- We will run our tests whenever we make a change to be sure nothing was broken.

Unit tests

- Testing starts with *unit tests*.
- Unit tests are written and run by developers.
- They test code at the lowest level of functionality.
- Basically, every function has a set of tests.

Test Driven Development

Suppose you want to add a new feature to a class. If you are practicing TDD, you

1. Write tests that test the new feature. (They should fail.)
2. Write code that passes the tests.
3. *Refactor*, or clean up your code.

Unit testing tools

- There are heaps of libraries and frameworks for writing unit testing modules.
- There is a family of testing tools that are collectively called "xUnit" that are all based on Smalltalk's SUnit.
- This means that there is a common approach to unit testing that applies to many languages.
- Python's standard testing library, `unittest`, is part of the xUnit family. We will use it this week.

Example

https://gist.github.com/tclark/2c32445d65e3ed9a63cc

Running your tests

- You can run one test script using the command `python your-script-name.py`.
- You can run all the test scripts in a directory with the command `python -m unittest discover your-directory-name`.

More information

- Read the `unittest` section on the Python testing pdf on the I: drive.
- See `https://docs.python.org/2/library/unittest.html`

Read these before the next class.

Today's exercise

Write at least one test class that covers one of your blackjack classes completely. Be prepared to show it to the group at the and of the hour.