Designing Good Test Cases

- Think about edge cases
- Think about "fundamentally different behavior"
- Test comprehensively -- your tests will only be as useful as you make them
- Your tests should cover the different features and behavior you expect

Code Coverage

- idea: metric that measures what percentage of code is run by tests (either lines, methods, etc.)
- not particularly great
- often included as an automated metric
- don't simply rely on it
- having 100% code coverage != having comprehensive tests of behavior

Mutation Testing

- mutant: a version of the source code with small change(s)
- at least one test should fail for non-equivalent mutants
- mutation score = killedmutants / (nummutants - numequivalent)