

# Ch19-Unittest

October 30, 2020

## 1 19 Unit Test

### 1.1 Topics

- unit testing
- test-driven development
- doctest

### 1.2 19.1 Test

- coders/software engineers spend as much time debugging and testing their codes as they spend on developing/writing the codes
- “To err is human, but to preserve in error is diabolic.” - anonymous
  - this more than two thousand-years old proverb fits with coding
- “Program testing can be used to show the presence of bugs, but never to show their absence!” - Edsger W. Dijkstra
- code coverage is used to measure the degree to which the source code of a program is executed when a particular test suite runs
- various coverage criteria:
  - function coverage
  - statement coverage
  - branch coverage
  - condition coverage
  - loop coverage
  - data-flow coverage
  - function entry/exit coverage
- full path coverage is usually impractical or impossible
  - module with a succession of  $n$  decision in it can have up to  $2^n$  paths
  - loops can result in an infinite number of paths
- see this repository for more examples of unit testing:  
<https://github.com/rambasnet/KattisDemos>

### 1.3 19.2 Test-driven development

- write test cases before or simultaneously along with the code implementation
- 3 simple ways to test your code natively at the functional level
  1. assert statements - natively supported in many languages; as we’ve used before
  2. doctest - simple, interactive technique, but makes the source code look messy, as they’re defined inside the same function and module

3. unittest - preferred; separates the code with test

#### 1.4 19.3 doctest

- <https://docs.python.org/3/library/doctest.html>
- to check that a module's docstrings are up-to-date by verifying that all interactive examples still work as documented.
- to perform regression testing by verifying that interactive examples from a test file or a test object work as expected.
- to write tutorial documentation for a package, liberally illustrated with input-output examples. Depending on whether the examples or the expository text are emphasized, this has the flavor of “literate testing” or “executable documentation”.

##### 1.4.1 see factorial.py in utility package for doctest example

#### 1.5 19.4 unittest

- <https://docs.python.org/dev/library/unittest.html#module-unittest>

##### 1.5.1 see fib\_\_unittest.py in utility package for unittest example

#### 1.6 19.5 Exercise

1. Write unittest to test functions in utility/factorial.py module.

[ ]: