Python course 2019

TDD <=> unit tests ## Define the project - Requirements - all minimal - YAGNI (you aren't gonna need it): Think twice before adding features. In doubt leave it out. - what are the minimal natural features the program should have - first: command line program

Preparation

- isolation: virtual environments -> you should use them!
  - `python3 -m venv pr-py36`
  - `source pr-py36/bin/activate`
  - deactivate with `deactivate`
- version control systems (VCS)

Implementation

- Make it work: Tests
- Make it right: Clean code. Code is read more often than it is written!
- Make it fast (if you need to)

TDD

- London school double loop TDD approach
- unit tests
- functional tests
- 3 laws of TDD
  - don't write production code until you have a failing unit test
  - don't write more unit tests than is sufficient to fail
  - don't write more production code than is sufficient to pass the failing test
- TDD requires discipline
- unofficial mascot: Testing goat
  - single minded
  - one step at a time goat

Generally

- avoid complexity