

Cheat Sheet

Shell commands

Running code

```
# Open the python interpreter
python3
```

```
# Run filename.py
python3 filename.py
```

```
# Open an interpreter with
  filename.py loaded
python3 -i filename.py
```

pytest

```
# Run tests in all files ending
  with '_test'
py.test -3
```

```
# Run tests in filename.py
py.test -3 filename.py
```

pylint

```
# Check filename.py with pylint
pylint3 filename.py
```

Coverage checking

```
# Measure the branch coverage of
  all python files in the current
  directory when running pytest
python3-coverage run --branch --
  source=. -m pytest
```

```
# View a simple report of the
  coverage measurements
python3-coverage report
```

```
# Generate a complete HTML report
  in 'htmlcov'
python3-coverage html
```

Python

Control Flow

```
while condition:
    statements

for var in some_iterable:
    statements

break # Exit the loop

continue # Next iteration

if condition1:
    statements
elif condition2:
    statements
else:
    statements

def func_or_method(args):
    statements

try:
    statements
except SomeException:
    statements
```

Classes

```
# Define a class
class someclass:
    def __init__(self, ...):
        statements

    def some_method(self, ...):
        statements
```

Strings

```
# concatenate two strings
string_a + string_b

# string formed by concatenating
# some_list with s as separator
s.join(some_list)
```

Lists

```
# Check if a exists in some_list
if a in some_list:
    statements

# Append a to some_list
some_list.append(a)

# Create a sorted list
sorted(some_list)

# Create a sorted list using the
# function foo(elem) to determine
# how it should be sorted
sorted(some_list, foo)

# Create a reversed list
reversed(some_list)
```

Pytest

```
import pytest

# function starts with test_
# picked up automatically by pytest
def test_something():
    assert boolean_expression

# Test will pass if SomeException is raised by the statements
def test_exception():
    with pytest.raises(SomeException):
        statements
```

Hypothesis

```
from hypothesis import given, strategies

# Test is run multiple times with values supplied from 'some_strategy'
@given(some_strategy)
def test_property(value):
    statements
```