# Hypothesis
# Simpler powerful test cases

Eli Gur

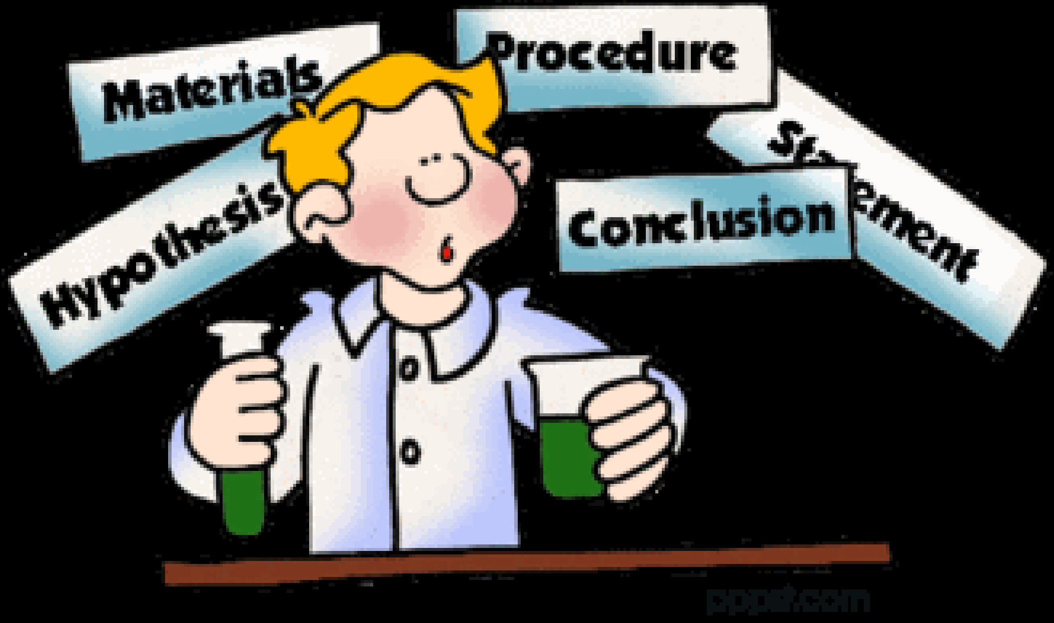
July 4th, 2016

# Agenda

- About Hypothesis
- “Normal” test cases
- Specifying what we want
- Simplifying values
- Remembering failures
- Examples
- Guarantying code

# “Normal” test cases

- Set up some data
- Perform some operations on the data
- Assert something about the result



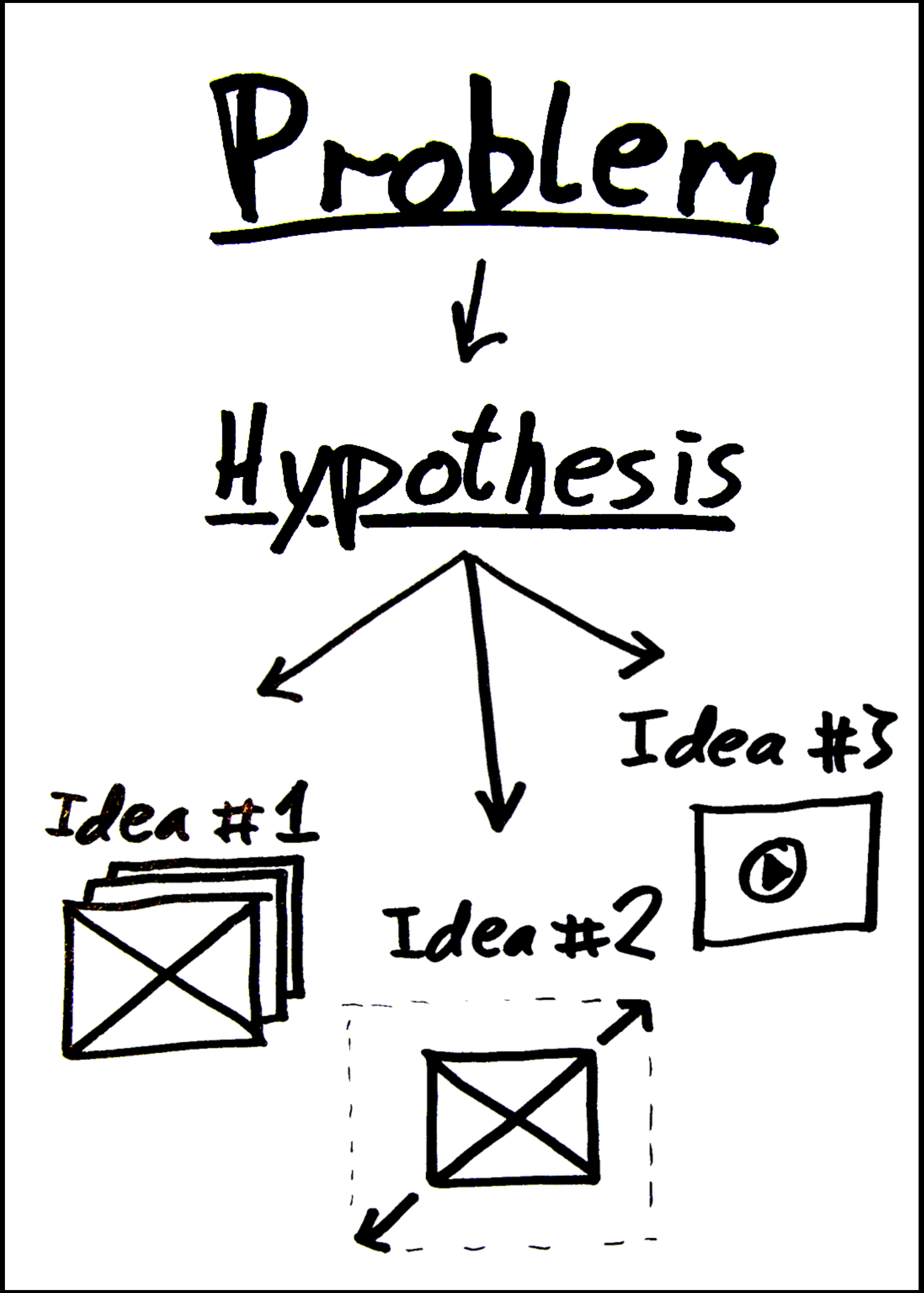
# Hypothesis test cases

- Specify what we want
- For all data matching some specification:
  - Perform some operations on the data
  - Assert something about the result

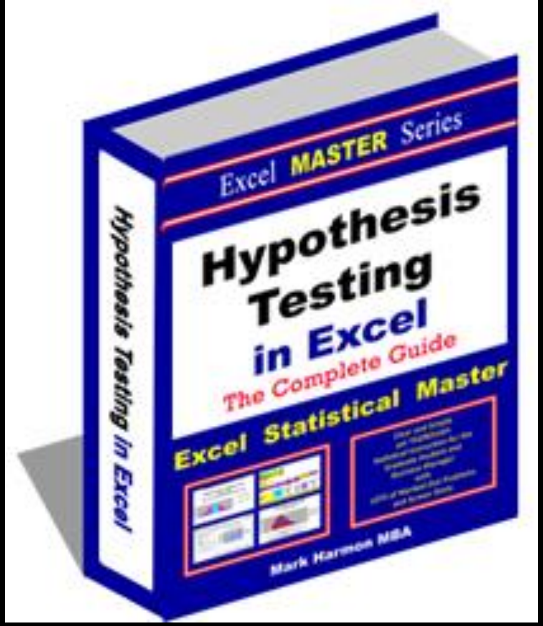
- Also called “Property based testing”

# Behind the scenes

- Hypothesis generates random data:
  - Matching specification
  - Checking guarantee hold
- When an example doesn't:
  - Cut it down to size
  - Simplify it to find smaller one that still causes the problem
  - Save the example for later



# How to write Hypothesis test case



- Decide on guarantees
  - Properties that should always hold true
  - Regardless of what the world throws at you
- Examples for guaranties:

```
assert deserialize(serialize(x)) == x
```

  - No exception or a particular type of exception
  - Deleted objects are no longer visible

# Generating test cases

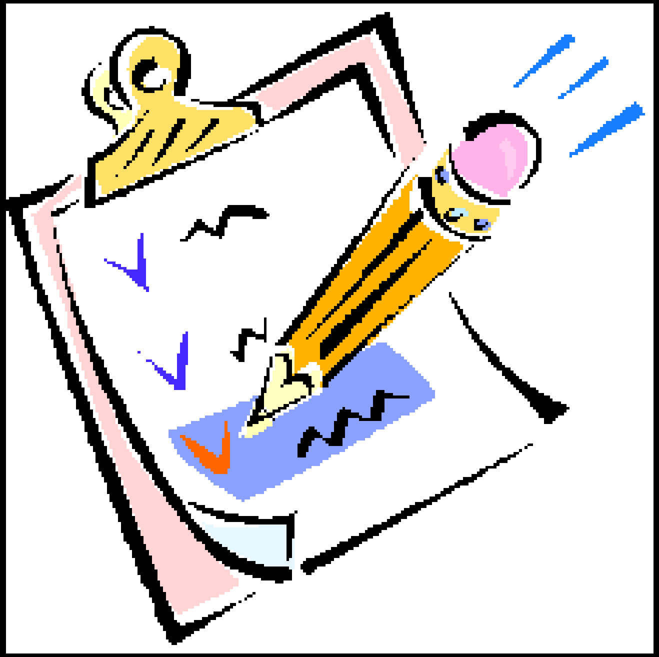
- given()
- example()
- assume()

# Example with float and int

```
from hypothesis import given, assume, example
import hypothesis.strategies as st
@given(x=st.floats(), y=st.integers())
@example(-0.0, 0)
def test_add(x: float, y: int):
    assume(y >= 0)
    assert x + y == y + x
```

# Example with list

```
from hypothesis import given
import hypothesis.strategies as st
@given(h_list=st.lists(st.integers()))
def test_reverse2(h_list: list):
    t_list = list(h_list)
    t_list.reverse()
    t_list.reverse()
    assert t_list == h_list
```



# Example with string

```
from hypothesis import given
import hypothesis.strategies as st
@given(txt=st.text())
def test_txt(txt: str):
    assert isinstance(txt, str)
```

# Example with date

```
from hypothesis import given
import hypothesis.extra.datetime as st
date1=st.dates()
for _ in range(4):
    print(date1.example())
```

# Advanced strategies

- lists() - supports min_size, average_size, max_size, unique
- tuples()
- build() - supports building test objects
- Trees, Nodes
- State machines
- Control amount of examples and iterations
- Interactive drawing of examples

# Available Integrations

- py.test, unittest, nose
- Django
- numpy
- fake_factory
- CI
- Timeout
- Profile and settings
- Python 2.7, 3.3+ and PyPy

# Thank You!