Writing tests that write themselves

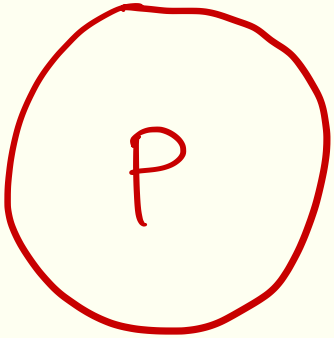
David Kua PyCon Canada 2017 @davidkua

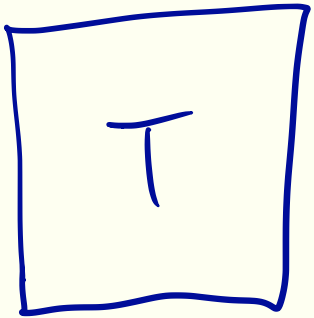
We're hiring!

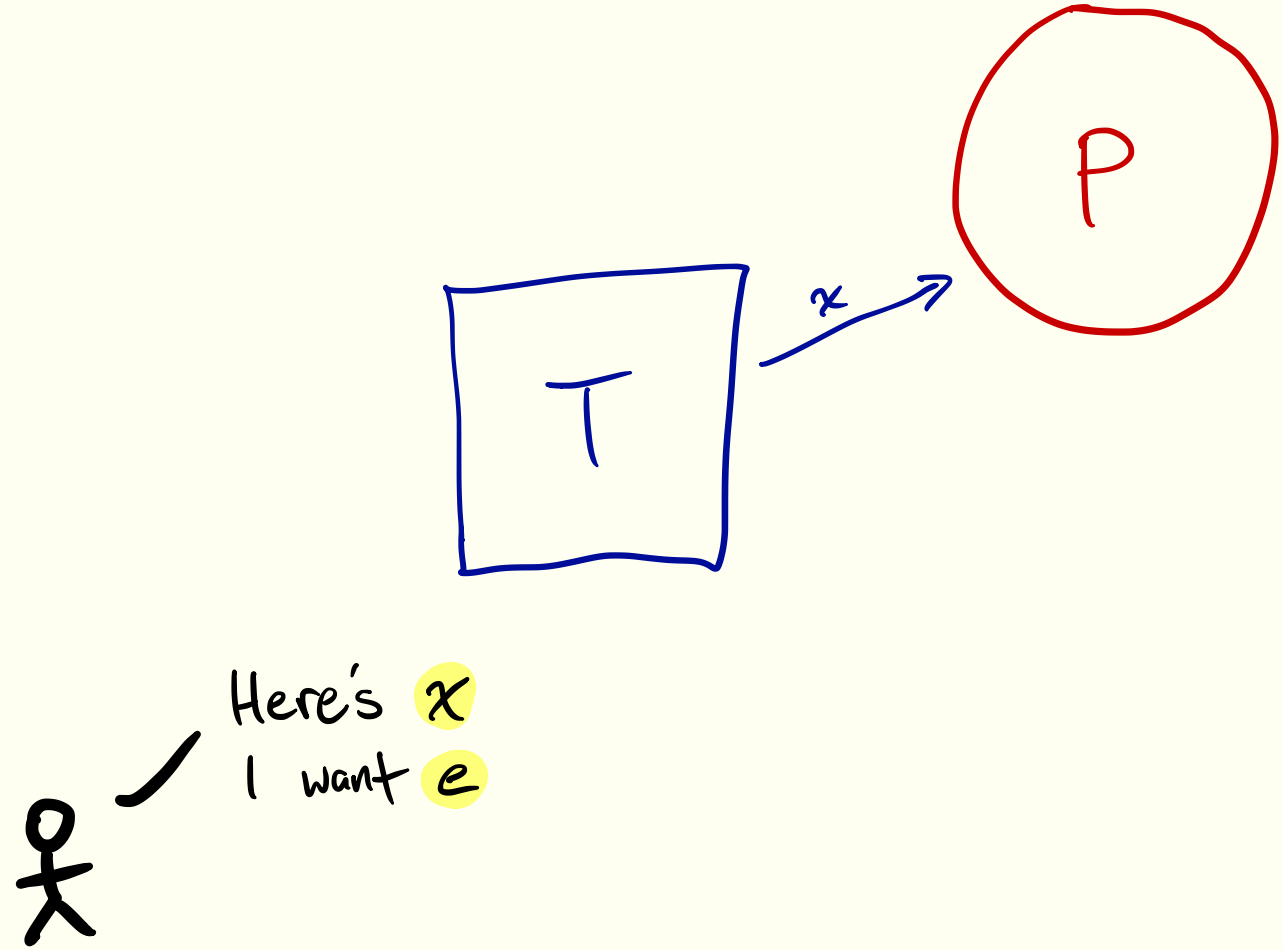
https://www.pagerduty.com/careers

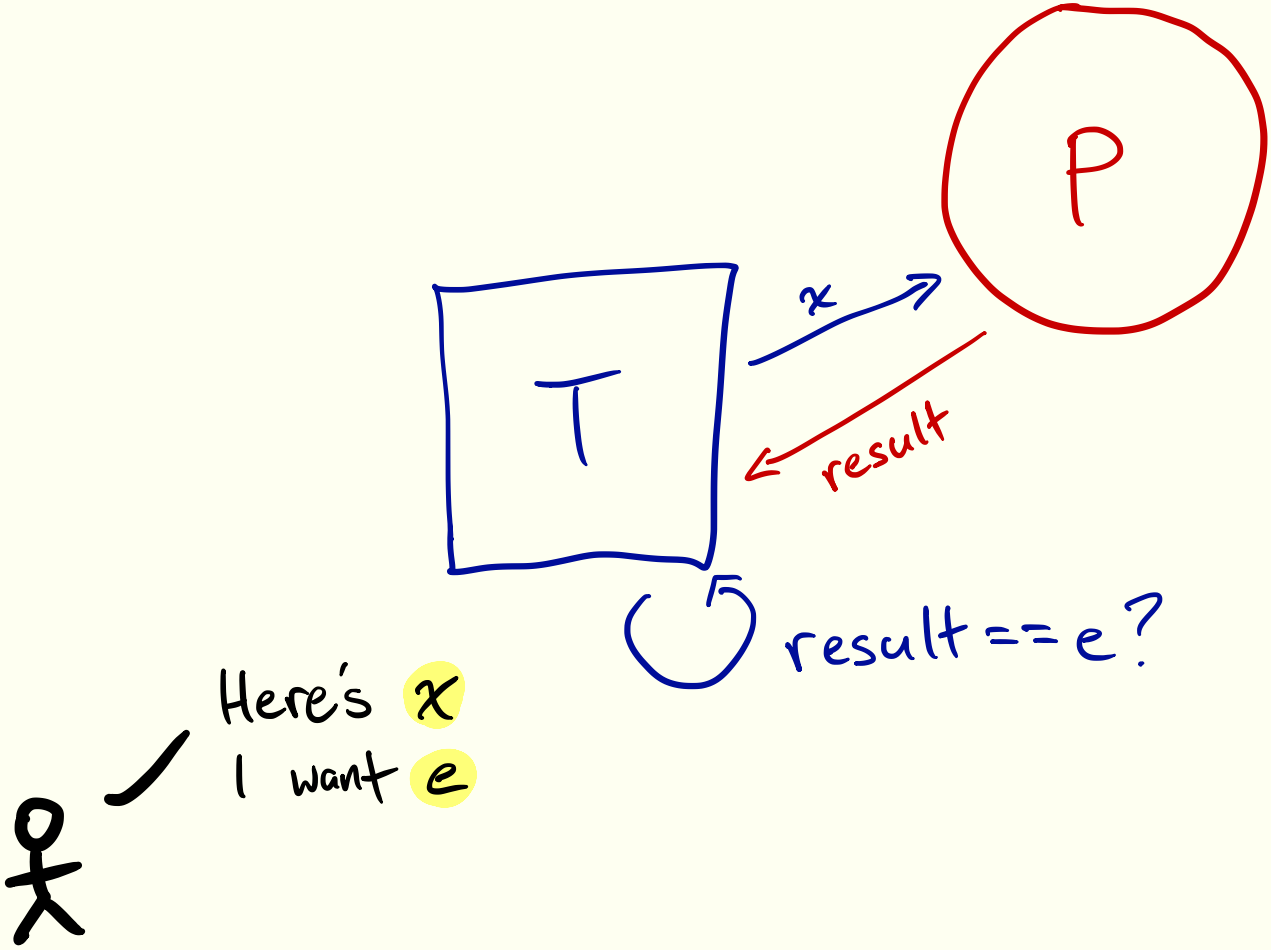
Agenda

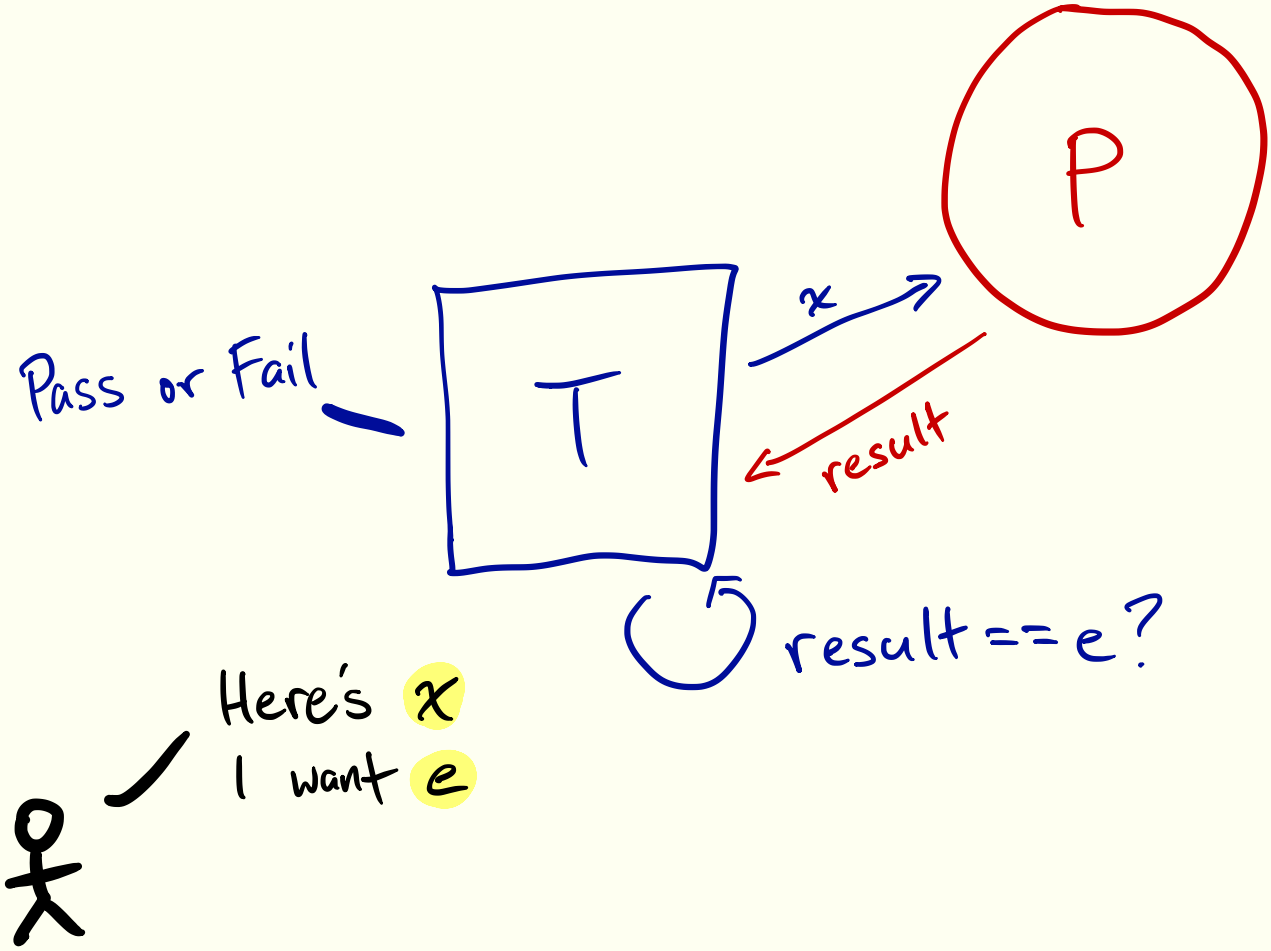
1. Overview of property-based testing (High level)
2. Exercises (As many as we can!)


P

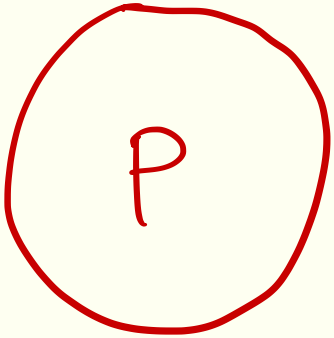

T

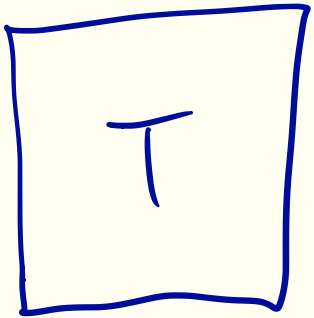

P
T
x
Here's x
I want e

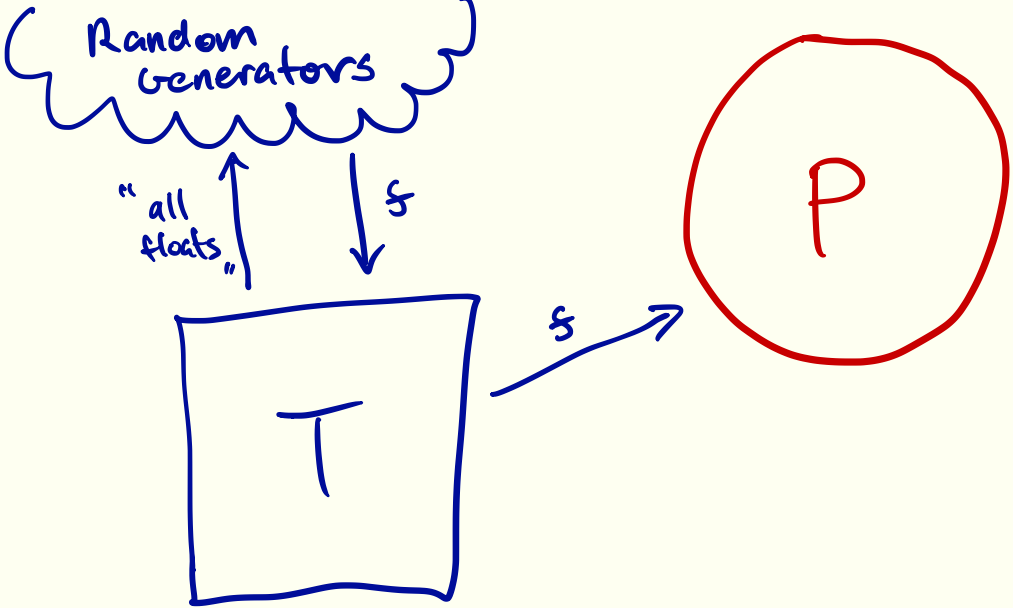

P
T
x
result
result == e?
Here's x
I want e

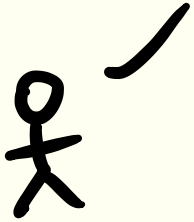

P
T
x
result
Pass or Fail
result == e?
Here's x
I want e


Random
Generators

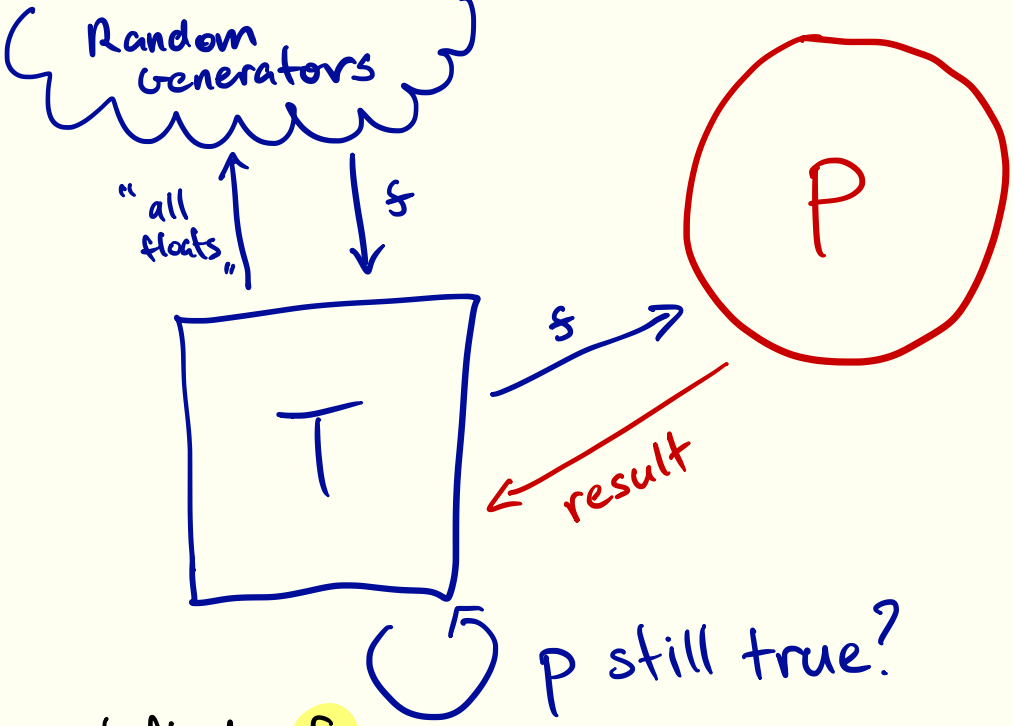

P


T

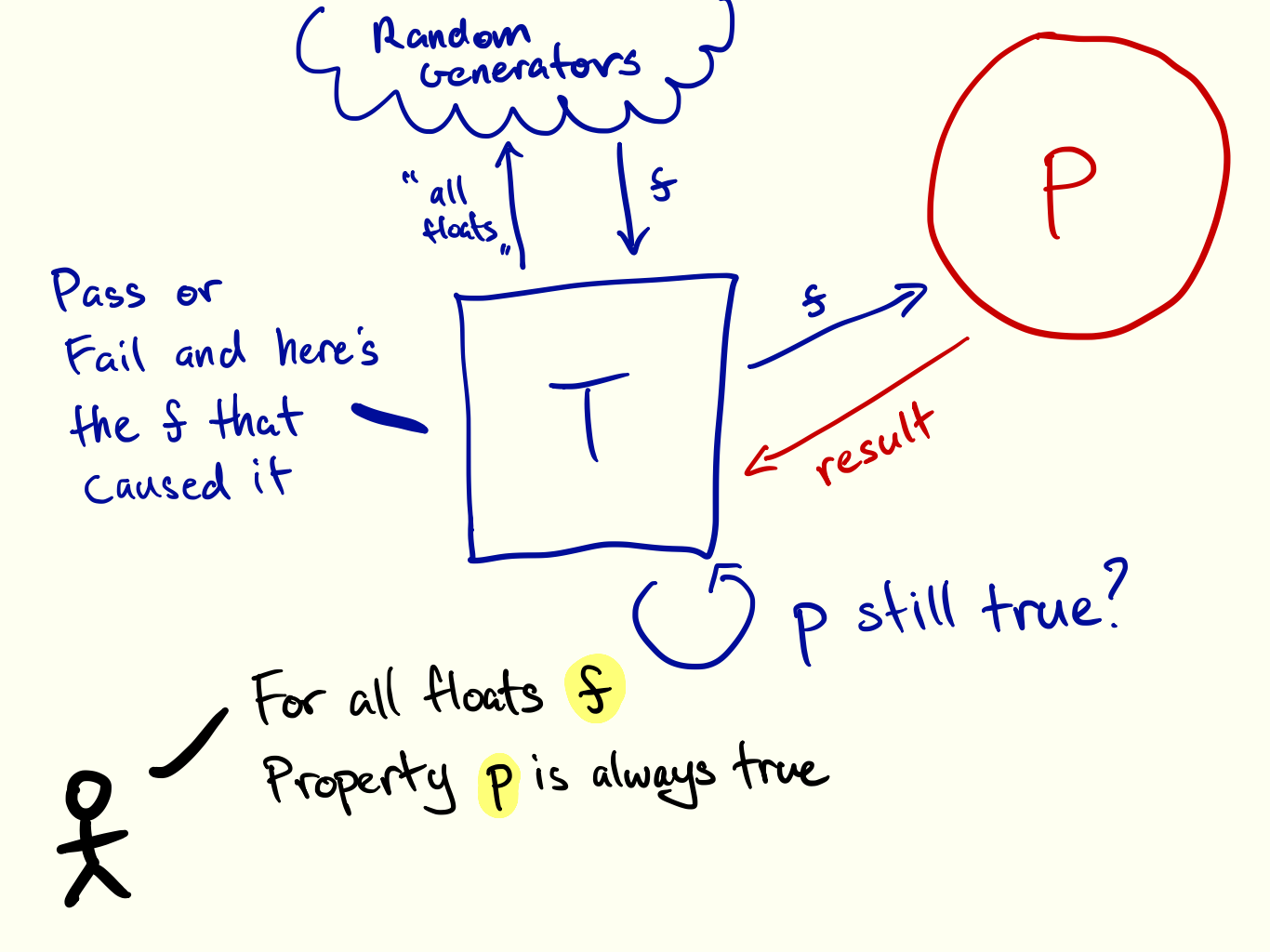




For all floats f
Property P is always true



For all floats f
Property p is always true


Random generators
"all floats"
f
P
Pass or Fail and here's the f that caused it
T
f
result
p still true?
For all floats f
Property p is always true

Unit Testing

Give an input. 2, 4, 57, 100, 999...

Give a result. true, true, false, true, false...

Property-based Testing

Describe the inputs. "All ints"

Describe condition(s) that must hold.

"Remainder is 0"

or

"Result is an int"

Pros

- # of tests only limited by power & time
- code overhead is minimized
- random generation more likely to find edge case

Cons

- harder & longer to write
- finding properties can be difficult
- random generation might miss an edge case

An extra tool in your toolkit

QuickCheck:
A Lightweight Tool for Random Testing of Haskell Programs

Koen Claessen & John Hughes

Frameworks

Haskell - QuickCheck

Erlang - Erlang QuickCheck, PropEr

Clojure - test.check

Scala - ScalaCheck

F# - FsCheck

PHP - Eris

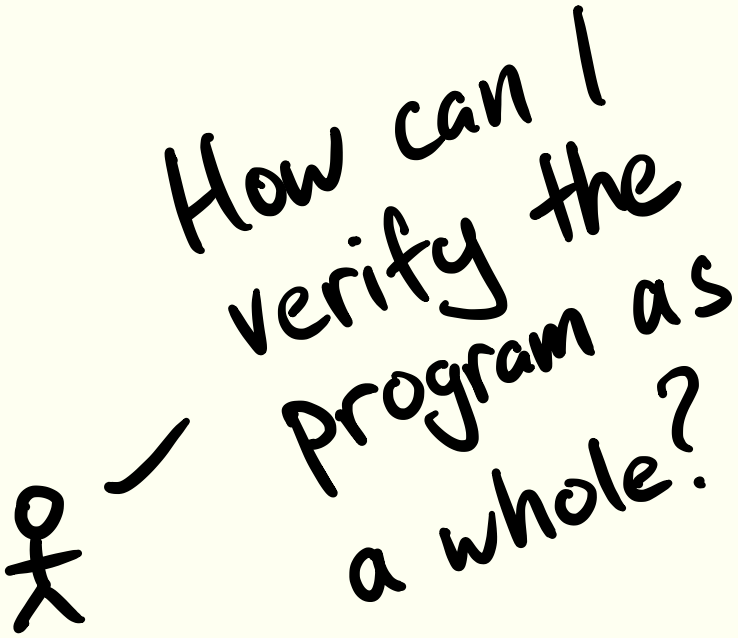
C - QuickCheck for C

... and more

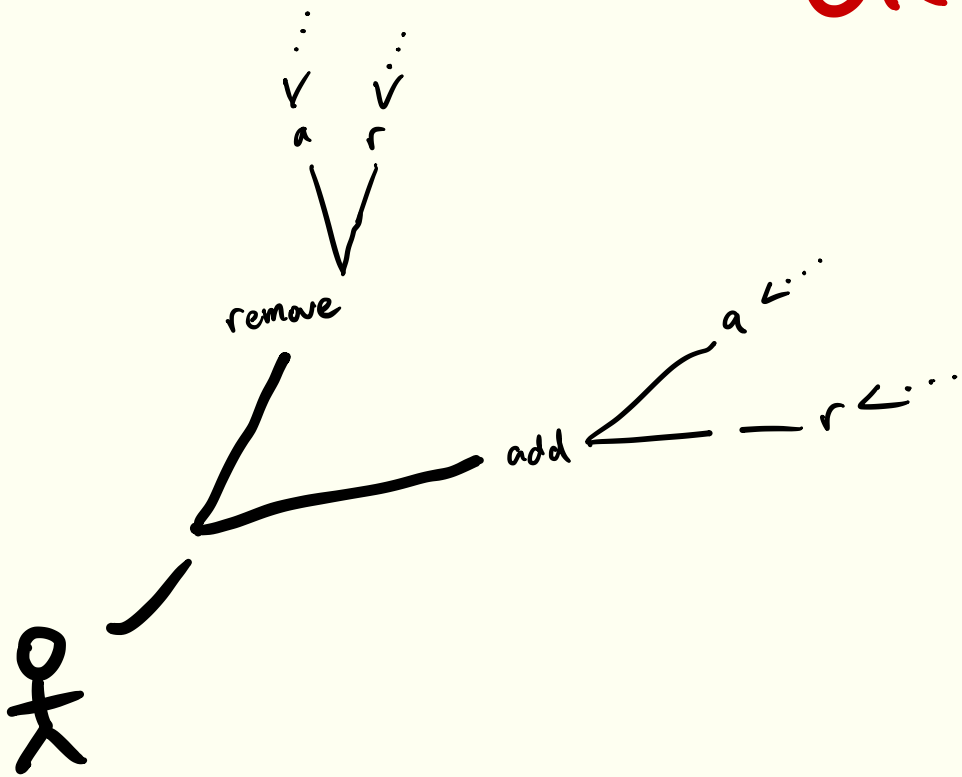
Hypothesis

- Python 2.7, 3.3+
- py.test, unittest, Nose (maybe)
- extra packages for: Django, datetime, NumPy
- super dope

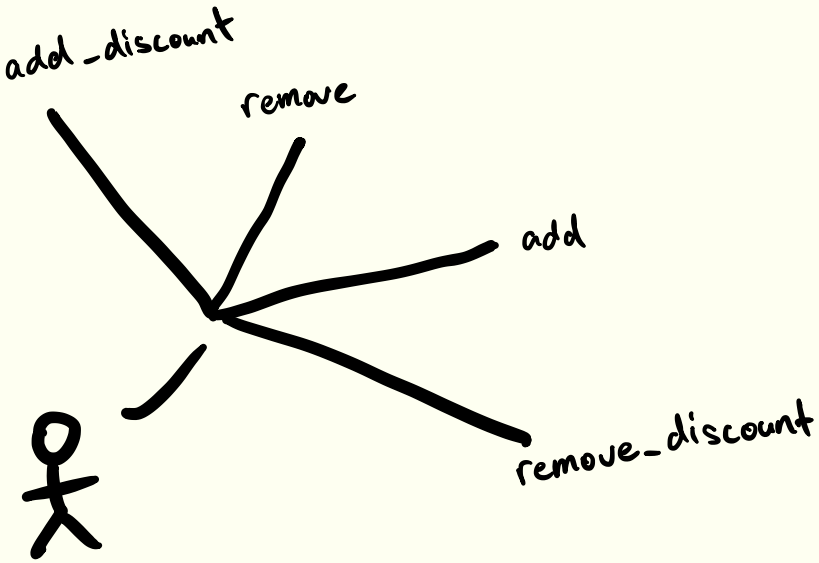
http://hypothesis.readthedocs.io/
https://github.com/HypothesisWorks/hypothesis-python/

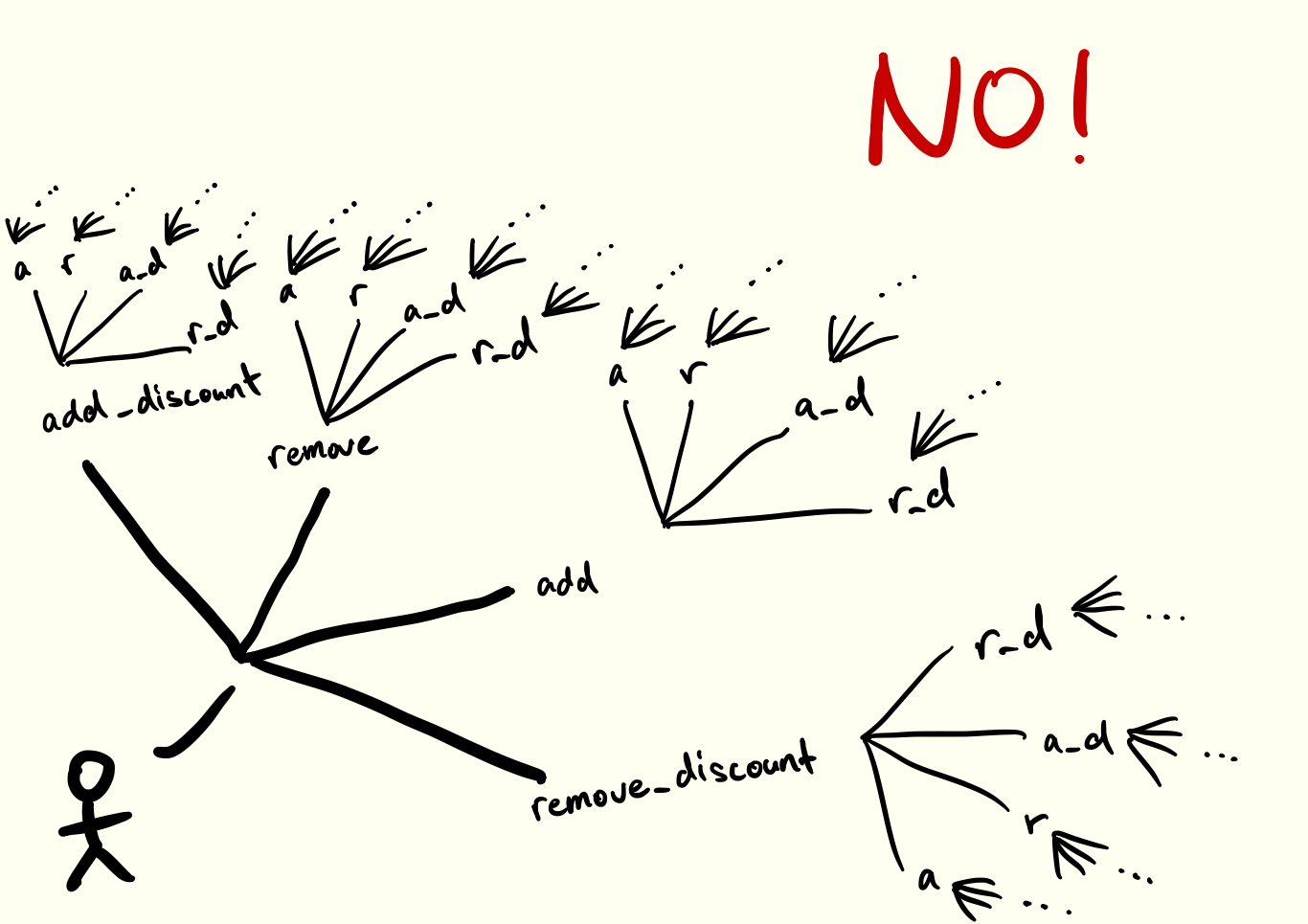

How can I verify the program as a whole?

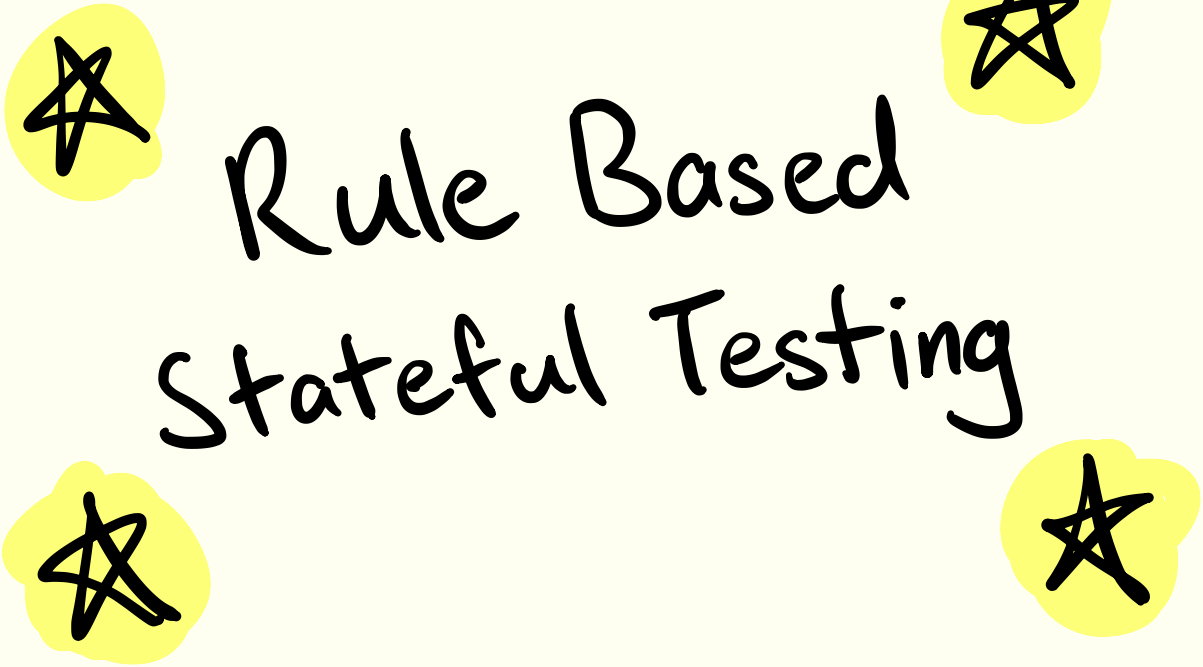
OK...



4...




NO!
a
r
a_d
r_d
add_discount
remove
add
remove_discount

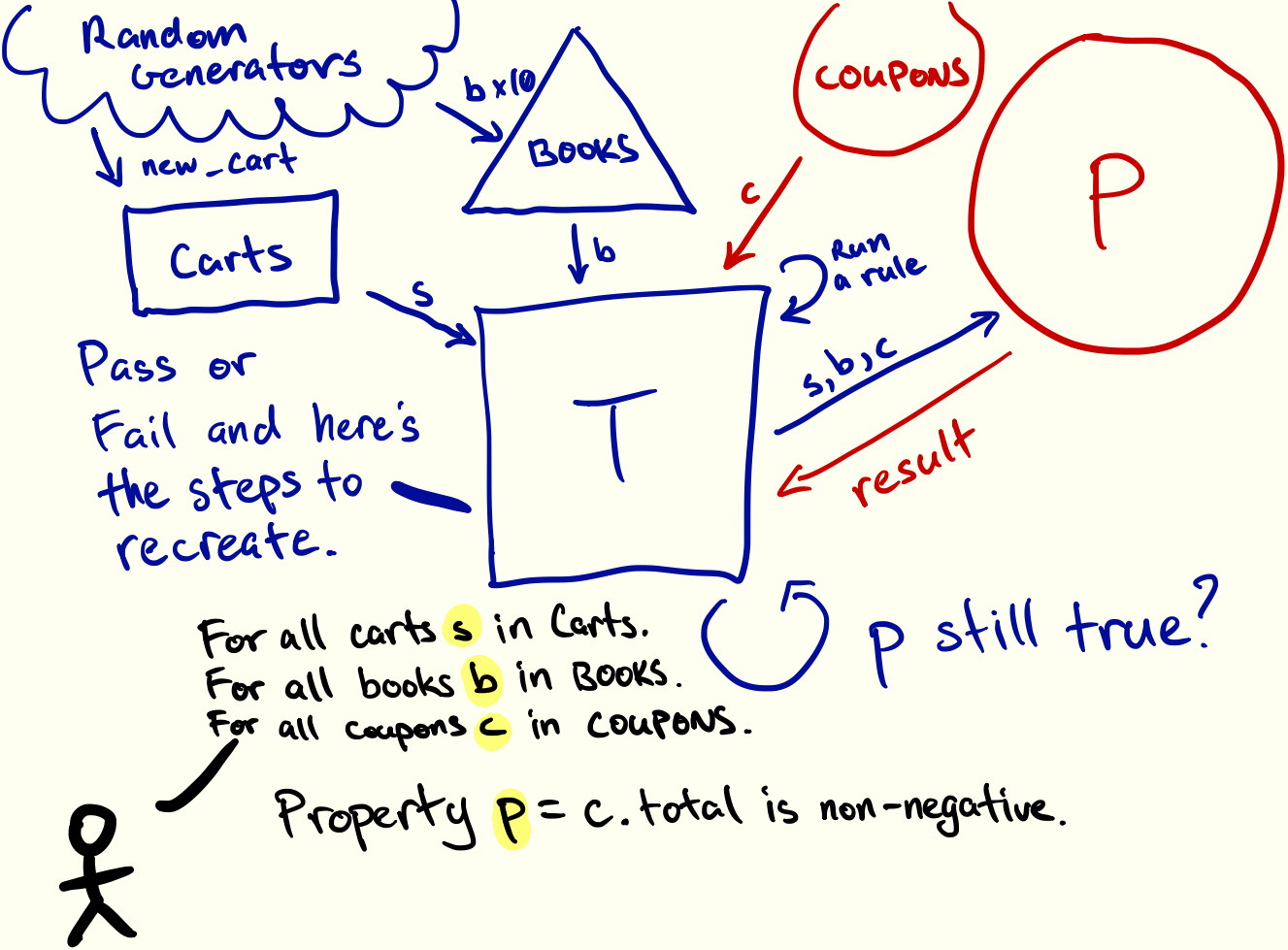

Rule Based
Stateful Testing

hypothesis.stateful

@rule

RuleBasedStateMachine

Bundle


Random Generators
new_cart
b x10
BOOKS
COUPONS
Carts
b
c
Run a rule
P
s
T
Pass or Fail and here's the steps to recreate.
s,b,c
result
For all carts s in Carts.
For all books b in BOOKS.
For all coupons c in COUPONS.
p still true?
Property p = c.total is non-negative.