Whose Test Is It Anyway?

Maaret Pyhäjärvi

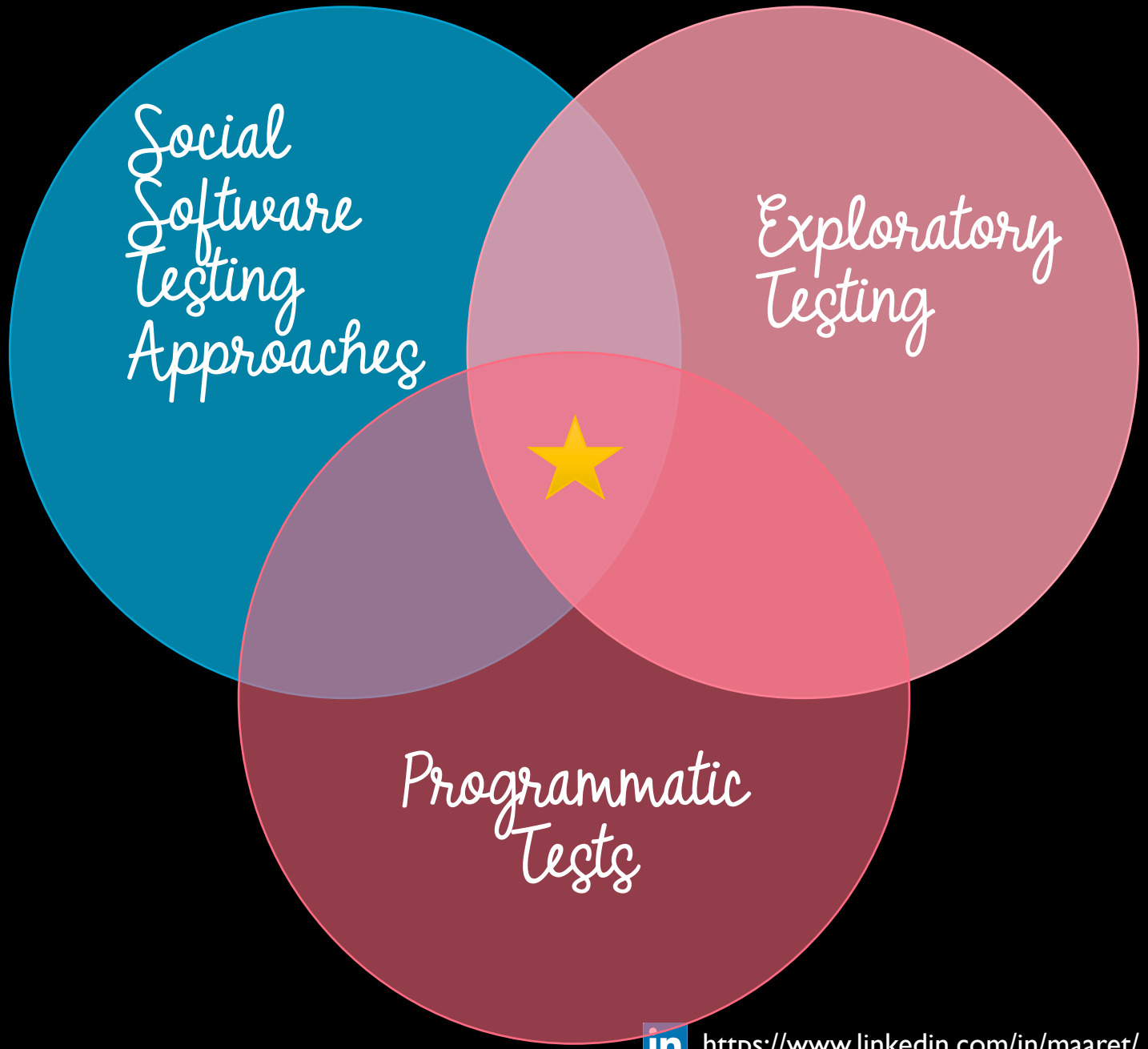
https://www.linkedin.com/in/maaret/ @maaretp@mas.to

Core message

Developers can test.

Testers need to test better than we do at large today.

https://www.linkedin.com/in/maaret/ @maaretp@mas.to



https://www.linkedin.com/in/maaret/ @maaretp@mas.to

important_program.py 1

important_program.py > ...

```
1    def test_error():
2        with pytest.raises(TypeError):
3            int_to_roman(1+2j)
4
5
6
7
8
9
10
11
12
```

This is a test.

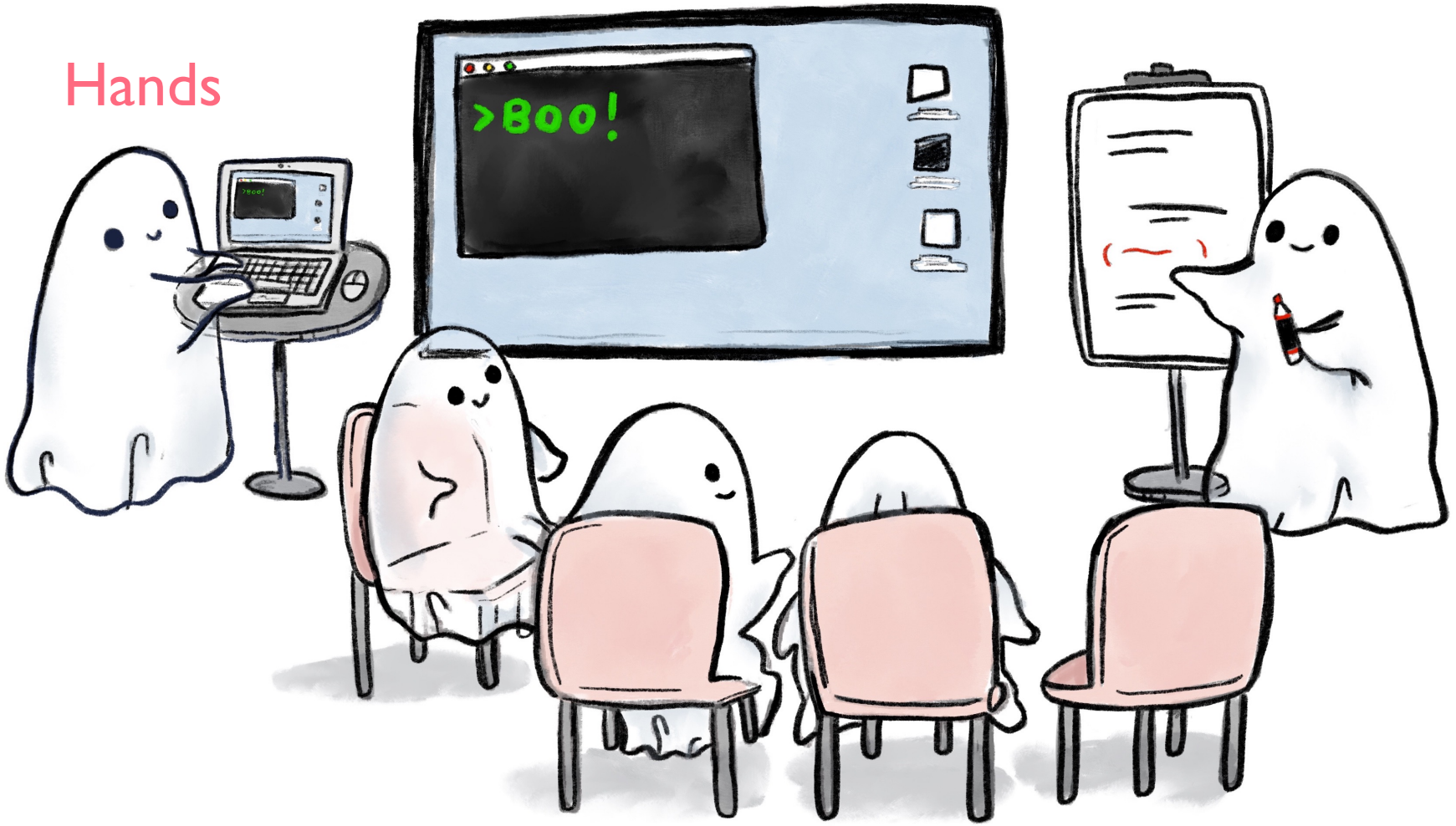
Whose test is it?

0 1 Live Share Watch Prettier


Improv
performance

Drive

Hands



Brains

ated

tor (Voice)

https://www.linkedin.com/in/maaret/ @maaretp@mas.to

Developers.

~~Developers~~ Team members with emphasis on programming.

Testers.

~~Testers~~ Team members with emphasis on testing (and programming on testing problems).

https://www.linkedin.com/in/maaret/ @maaretp@mas.to

"Find (some of) What Others May Have Missed"

Stakeholders happy,
even delighted
–Quality Information

Surprise!

Good Team's Output
–Quality Information

Results Gap

Pick up the pizza boxes…

Less than Good
Team's Output
–Quality Information

Results Gap
on a Team that thinks
Testers == Testing

https://www.linkedin.com/in/maaret/ @maaretp@mas.to

important_program.py 1

important_program.py

```python
# from int to roman converter
def int_to_roman(input):
    if not isinstance(input, type(1)):
        raise TypeError, "expected integer, got %s" % type(input)
    if not 0 < input < 4000:
        raise ValueError, "Argument must be between 1 and 3999"
    ints = (1000, 900,  500, 400, 100,  90, 50,  40, 10,  9,   5,  4,   1)
    nums = ('M',  'CM', 'D', 'CD','C', 'XC','L','XL','X','IX','V','IV','I')
    result = []
    for i in range(len(ints)):
        count = int(input / ints[i])
        result.append(nums[i] * count)
        input -= ints[i] * count
    return ''.join(result)
```

Computer Assisted Software Authorship

https://github.com/features/copilot/

WE are *accountable*

1. Legal / Ethical
2. Intent / Implementation
3. Domain for the Layman
4. Domain for the Expert
5. Reference Implementation
6. People Filtering
7. Interesting side effects

https://www.linkedin.com/in/maaret/ @maaretp@mas.to

roman.py — New Important Project

roman.py

```
1    # Integer to Roman
2
```

Guessing with power to accept

For Mood:

https://aboutmonica.com/blog/how-to-make-your-vs-code-sparkle/

0 0 Live Share Watch Prettier

roman.py — New Important Project

roman.py

```
1    # Integer to Roman
2    # By: Shahzeb Jadoon
3
```

Guessing with power to accept

Who Am I?

What am I testing again? My program or the tool?

0 0 Live Share Watch Prettier

Note:

…pillaging digital content without consent, compensation and attribution

Even if *legal* not *ethical*.

https://www.linkedin.com/in/maaret/ @maaretp@mas.to

important_program.py 1

important_program.py

```
#roman numeral converter
#by: jake
def
```

Code Review

CTRL+enter for alternatives

GitHub Copilot

```
Synthesizing 10/10 solutions

=======

Accept Solution
def roman_numeral_converter(number):
    if number == 1:
        return "I"
    elif number == 2:
        return "II"
    elif number == 3:
        return "III"
    elif number == 4:
        return "IV"
    elif number == 5:
        return "V"
    elif number == 6:
        return "VI"
    elif number == 7:
        return "VII"
    elif number == 8:
        return "VIII"
    elif number == 9:
        return "IX"
    elif number == 10:
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL JUPYTER zsh

Say We Have This...

roman.py

```python
def roman ( num ):
    """
    Convert an integer to a Roman numeral.

    >>> [roman(n) for n in range(1, 6)]
    ['I', 'II', 'III', 'IV', 'V']
    """
    if not ( 0 < num < 4000 ):
        raise ValueError ( 'number out of range (must be 1..3999)' )
    if not isinstance ( num , int ):
        raise TypeError ( 'non-integers can not be converted' )
    ints = ( 1000 , 900 , 500 , 400 , 100 , 90 , 50 , 40 , 10 , 9 , 5 , 4 , 1 )
    nums = ( 'M' , 'CM' , 'D' , 'CD' , 'C' , 'XC' , 'L' , 'XL' , 'X' , 'IX' , 'V' , 'IV' , 'I' )
    result = []
    for i in range ( len ( ints )):
        count = int ( num / ints [ i ])
        result . append ( nums [ i ] * count )
        num -= ints [ i ] * count
    return ''. join ( result )


```

Done?

https://www.linkedin.com/in/maaret/ @maaretp@mas.to

```python
import pytest
@pytest.mark.parametrize('number, expected', [(3, 'III'), (4, 'IV'),
    (9, 'IX'), (10, 'X'), (40, 'XL'), (50, 'L'), (90, 'XC'), (100, 'C'),
    (400, 'CD'), (500, 'D'), (900, 'CM'), (1000, 'M'), (12, 'XII'), (45, 'XLV'),
    (1992, 'MCMXCII'), (2022, 'MMXXII'), (490, 'CDXC'), (3333, 'MMMCCCXXXIII')])
def test_first(number, expected):
    assert convert(number) == expected

@pytest.mark.parametrize('input', [(1.5), (1.6), ('moi'), (None)])
def test_second(input):
    with pytest.raises(ValueError):
        convert(input)

def numbers_list(num):
    num_list = []
    for i in range(1, num + 1):
        num_list.append(i)
    return num_list

from approvaltests.combination_approvals import verify_all_combinations

def test_all_cases():
    verify_all_combinations(convert, [
        numbers_list(100)])

```

Some Tests

Done?

Shapes of Approaches

Asserts

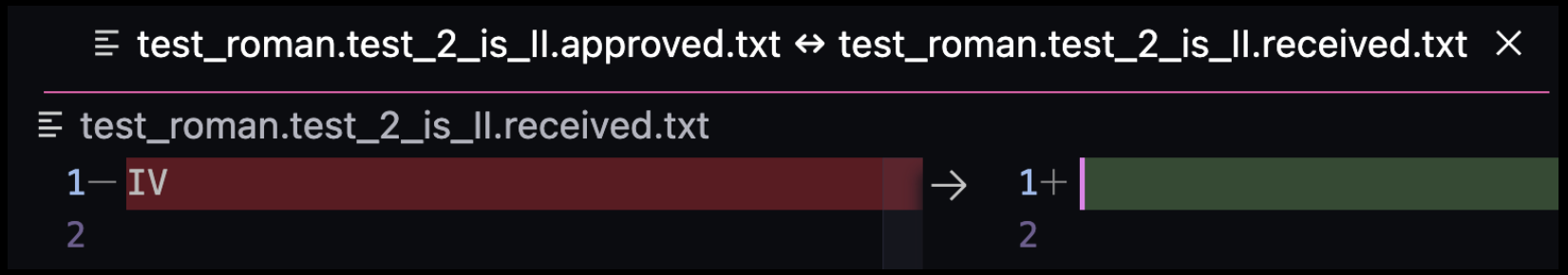
```
def test_4_is_IV():
    assert roman(4) == 'IV'
```

```
from assertpy import soft_assertions, assert_that

def test_four_is_IV():
    with soft_assertions():
        assert_that(roman(4)).is_equal_to('IIII')
        assert_that(roman(4)).is_equal_to('IV')
```

Approvals

```
from approvaltests import verify
def test_2_is_II():
    verify(roman(4))
```



https://www.linkedin.com/in/maaret/ @maaretp@mas.to

From One to Many

Asserts

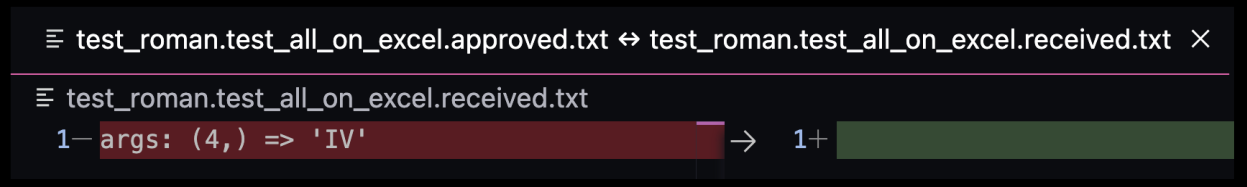
```python
@pytest.mark.parametrize("num, expected", [(4, 'IV'),])
def test_roman_converter(num, expected):
    assert roman(num) == expected
```

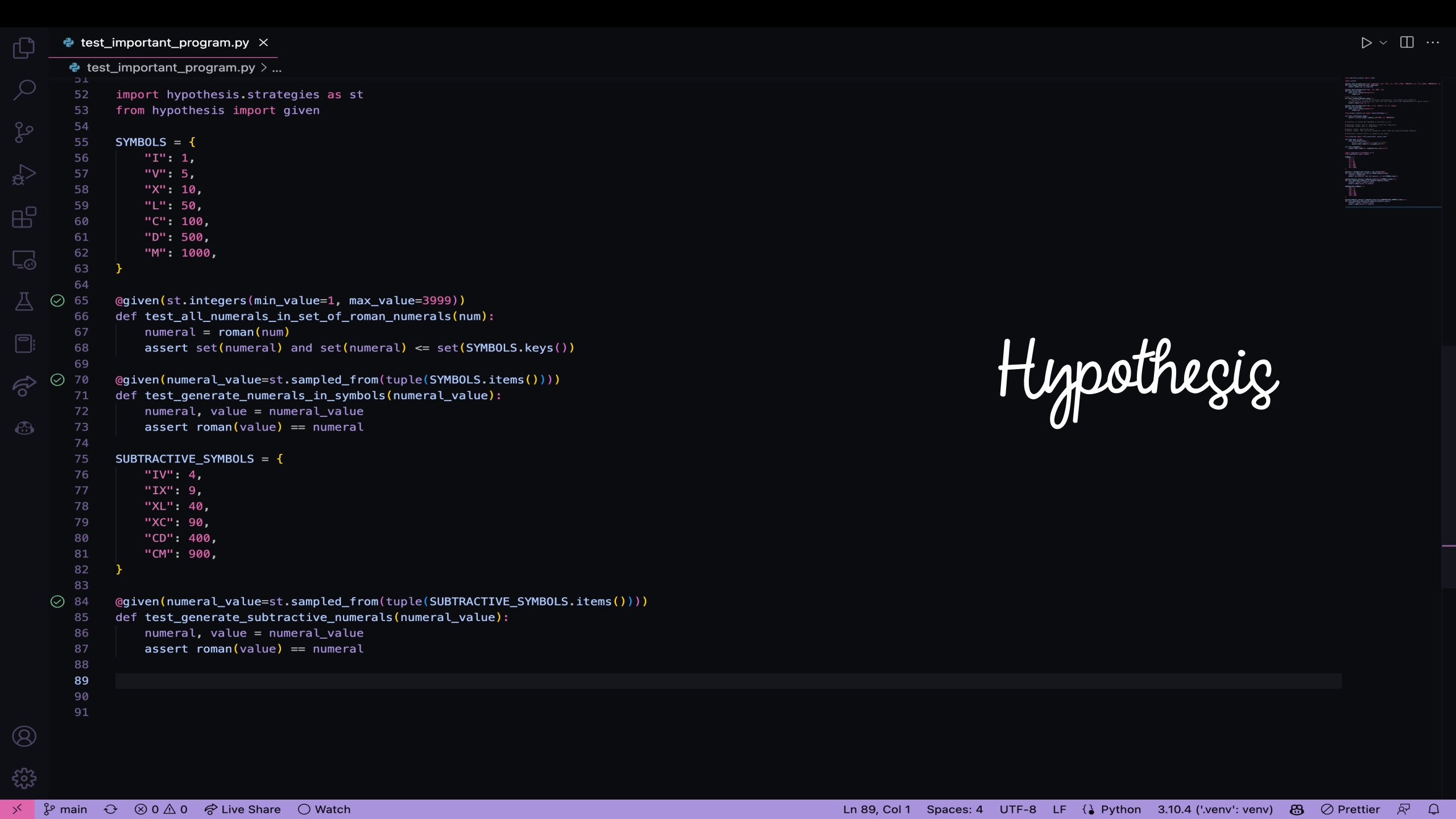
Approvals

```python
def numbers_list(start, stop):
    num_list = []
    for i in range(start, stop + 1):
        num_list.append(i)
    return num_list

from approvaltests.combination_approvals import verify_all_combinations

def test_all_on_excel():
    verify_all_combinations(roman, [
        numbers_list(4, 4)])
```




test_important_program.py
test_important_program.py > ...
import hypothesis.strategies as st
from hypothesis import given
SYMBOLS = {
"I": 1,
"V": 5,
"X": 10,
"L": 50,
"C": 100,
"D": 500,
"M": 1000,
}
@given(st.integers(min_value=1, max_value=3999))
def test_all_numerals_in_set_of_roman_numerals(num):
numeral = roman(num)
assert set(numeral) and set(numeral) <= set(SYMBOLS.keys())
@given(numeral_value=st.sampled_from(tuple(SYMBOLS.items())))
def test_generate_numerals_in_symbols(numeral_value):
numeral, value = numeral_value
assert roman(value) == numeral
SUBTRACTIVE_SYMBOLS = {
"IV": 4,
"IX": 9,
"XL": 40,
"XC": 90,
"CD": 400,
"CM": 900,
}
@given(numeral_value=st.sampled_from(tuple(SUBTRACTIVE_SYMBOLS.items())))
def test_generate_subtractive_numerals(numeral_value):
numeral, value = numeral_value
assert roman(value) == numeral
Hypothesis
main
0 0
Live Share
Watch
Ln 89, Col 1
Spaces: 4
UTF-8
LF
Python
3.10.4 ('.venv': venv)
Prettier

Note:

…developer intent. It might not do what you wanted it to do.

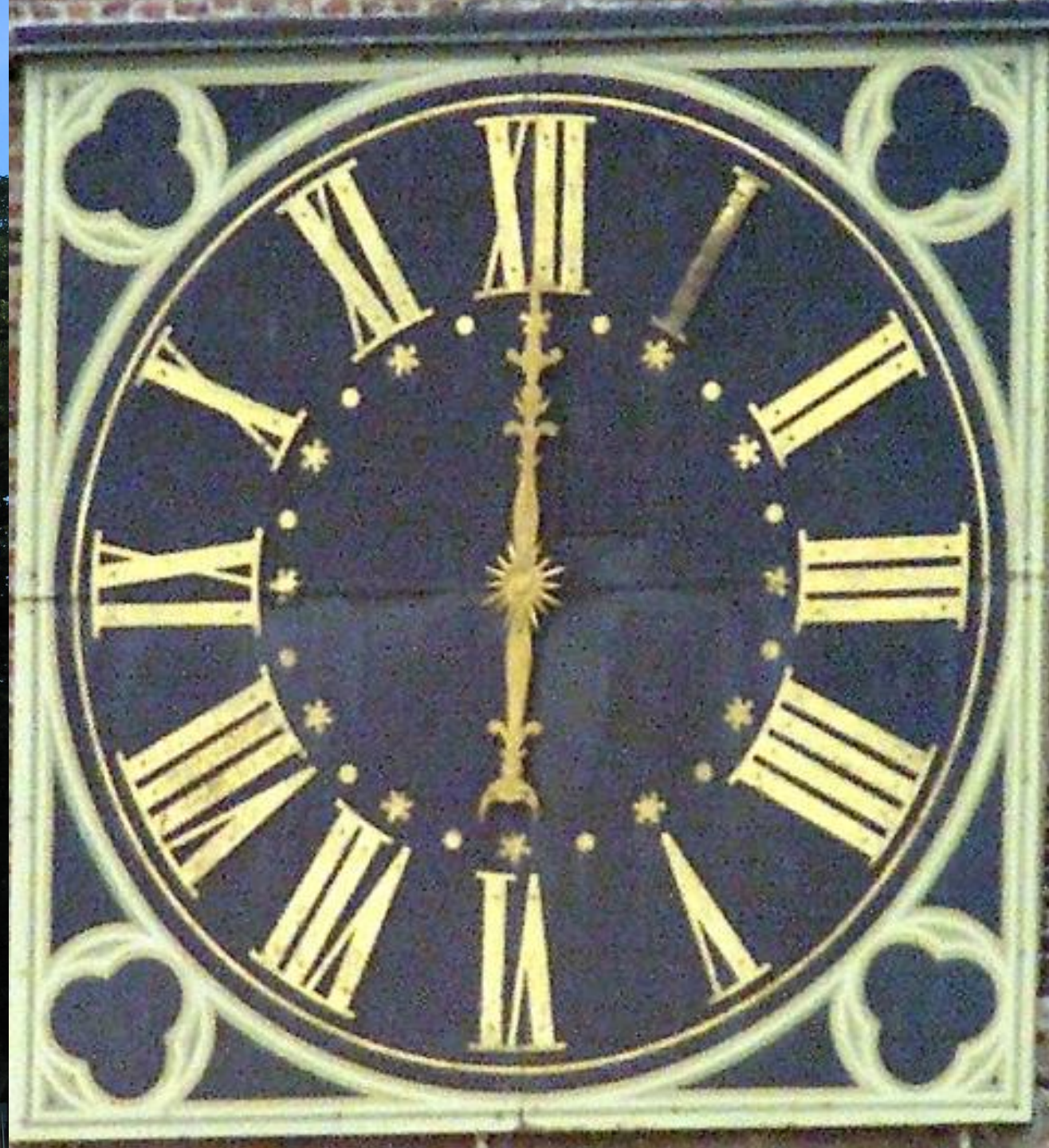
Developer intent

Part I. Review for correctness and conciseness
Part II. Input → Output
Part III. Rules of behavior boundaries
Part IIII. Coverage
Part IIIII. Sampling vs wide nets (approvals)
Part VI. Properties

https://www.linkedin.com/in/maaret/ @maaretp@mas.to

https://www.amalgamsab.com/iiii-or-is-it-iv.html

Domain rules IV → IIII in clock design as per orders of King Louis XIV of France



Roman numerals

From Wikipedia, the free encyclopedia

For a description of numeric words in Latin, see Latin numerals.

Roman numerals are a numeral system that originated in ancient Rome and remained the usual way of writing numbers throughout Europe well into the Late Middle Ages. Numbers in this system are represented by combinations of letters from the Latin alphabet. Modern style uses seven symbols, each with a fixed integer value:[1]

Symbol	I	V	X	L	C	D	M
Value	1	5	10	50	100	500	1000

The use of Roman numerals continued long after the decline of the Roman Empire. From the 14th century on, Roman numerals began to be replaced by Arabic numerals; however, this process was gradual, and the use of Roman numerals persists in some applications to this day.

One place they are often seen is on clock faces. For instance, on the clock of Big Ben (designed in 1852), the hours from 1 to 12 are written as:

I, II, III, IV, V, VI, VII, VIII, IX, X, XI, XII

The notations IV and IX can be read as "one less than five" (4) and "one less than ten" (9), although there is a tradition favouring representation of "4" as "IIII" on Roman numeral clocks.[2]

Other common uses include year numbers on monuments and buildings and copyright dates on the title screens of movies and television programs. MCM, signifying "a thousand, and a hundred less than another thousand", means 1900, so 1912 is written MCMXII. For the years of this century, MM indicates 2000. The current year is MMXXII (2022).

Contents [hide]

1 Description
1.1 Standard form
1.2 Variant forms
1.2.1 Additive notation
1.2.2 Irregular subtractive notation
1.2.3 Rare variants
1.2.4 Non-numeric combinations
1.3 Zero
1.4 Fractions
1.5 Large numbers
1.5.1 Apostrophus
1.5.2 Vinculum
2 Origin
2.1 Etruscan numerals
2.2 Early Roman numerals
2.3 Classical Roman numerals
3 Use in the Middle Ages and Renaissance
4 Modern use

This article contains special characters. Without proper rendering support, you may see question marks, boxes, or other symbols.

Roman numerals on stern of the ship *Cutty Sark* showing draught in feet. The numbers range from 13 to 22, from bottom to top.

Part of a series on

Numeral systems

Place-value notation [show]

Sign-value notation [hide]

Non-alphabetic

Attic · Brahmi · Chuvash · Egyptian · Etruscan · Kharosthi · Prehistoric counting · Proto-cuneiform · **Roman** · Tally marks

Alphabetic

Abjad · Armenian · Āryabhaṭa · Cyrillic · Ge'ez ·

Main page
Contents
Current events
Random article
About Wikipedia
Contact us
Donate

Contribute
Help
Learn to edit
Community portal
Recent changes
Upload file

Tools
What links here
Related changes
Special pages
Permanent link
Page information
Cite this page
Wikidata item

Print/export
Download as PDF
Printable version

In other projects
Wikimedia Commons

Languages
Deutsch
Eesti
Español
Français
Latina
Nederlands
Русский

Roman **Numerals**

- Intro to Roman Numerals
- Numerals Converter
- Numerals Chart (1-100)
- Video Guide
- The Basics
- Years and Dates
- Large Numbers
- Zeros and Fractions

Roman numerals use seven letters: I, V, X, L, C, D and M to represent the numbers 1, 5, 10, 50, 100, 500 and 1000. These seven letters make up thousands of numbers. Read our full guide below or use the converter and chart to quickly check a numeral.

Numerals Converter

Enter numeral or number...

Upper Boundary?

3999	MMMCMXCIX
3999999	M̄M̄M̄C̄M̄X̄C̄I̅X̅CMXCIX

https://www.linkedin.com/in/maaret/ @maaretp@mas.to

AutoSave OFF

Single-Roman

Home Insert Draw Page Layout Formulas Data Review View Tell me

Share Comments

Paste Cut Copy Format

Calibri (Body) 12

Wrap Text Merge & Centre

General

Conditional Formatting Format as Table Cell Styles

Insert Delete Format

Auto-sum Fill Clear

Sort & Filter Find & Select

Analyse Data

Sensitivity

G14

	A	B	C	D	E	F	G	H	I
1	Decimal	Classic Roman	More Concise 1	More Concise 2	More Concise 3	Simplified Roman			
2	499	CDXCIX	LDVLIV	XDIX	VDIV	ID			
3									
4									
5									
6									
7									
8									
9									
10									
11									
12									
13									
14									
15									
16									
17									
18									
19									
20									
21									

```python
from helpers.numeral_ref import NumeralRefPage as r
from helpers.excel_ref import ExcelRef as e

def test_ref(browser_page):
    assert r(browser_page).numeral_ref(4) == 'IV'

def test_excel():
    assert e.extract_cell_classic(4) == 'IV'
    assert e.extract_cell_simplified(4) == 'IV'

def excel(num):
    return e.extract_cell_simplified(num)

def test_online_and_excel(browser_page):
    assert r(browser_page).numeral_ref(399) == e.extract_cell_classic(399)
```

References...

```python
from playwright.sync_api import Page

from conftest import open_to_url


class NumeralRefPage:

    def __init__(self, page: Page):
        self.page = page

    def numeral_ref(self, inputnumber):
        url = "https://www.knowtheromans.com/roman-numerals/"
        self.page.goto(url)
        self.page.fill("#convertnumeralinput", str(inputnumber))
        outputnumber = self.page.inner_html("#numeralsout").strip("\n")
        return str(outputnumber)

```

References...

```python
import xlrd

class ExcelRef:

    def extract_cell_classic(number):
        loc = ("ref.xls")
        wb = xlrd.open_workbook(loc)
        sheet = wb.sheet_by_index(0)
        return sheet.cell_value(number, 1)

    def extract_cell_simplified(number):
        loc = ("ref.xls")
        wb = xlrd.open_workbook(loc)
        sheet = wb.sheet_by_index(0)
        return sheet.cell_value(number, 5)
```

References...

```
17  def numbers_list(num):
18      num_list = []
19      for i in range(1, num + 1):
20          num_list.append(i)
21      return num_list
22
23  from approvaltests.combination_approvals import verify_all_combinations
24
25  def test_all_on_excel():
26      verify_all_combinations(excel, [
27          numbers_list(100)])
28
29  def test_all_on_browser(browser_page):
30      verify_all_combinations(r(browser_page).numeral_ref, [
31          numbers_list(100)])
```

ApprovalTests

Done?

Note:

…business rules. You think you know them and yet you don't.

https://www.linkedin.com/in/maaret/ @maaretp@mas.to

Domain

Part I. Be the resident expert. Ask around.
Part II. Rules. More rules.
Part III. Find better experts.
Part IIII. Disagreeing with boundaries.
Part IIIII. Oracles.
Part VI. Find better oracles.
Part VII. No user would do what users would do.

https://www.linkedin.com/in/maaret/ @maaretp@mas.to

Environment

Part I. Dependencies.

Part II. Interruptions. Both software and hardware.

Part III. People.

"People are not pure functions; they have all sorts of interesting side effects." - Engineering Management for the Rest of Us, Sarah Drasner

… nor are pure functions if you grow the boundary of what might fail.

https://www.linkedin.com/in/maaret/ @maaretp@mas.to

WE are *accountable*

1. Legal / Ethical
2. Intent / Implementation
3. Domain for the Layman
4. Domain for the Expert
5. Reference Implementation
6. People Filtering
7. Interesting side effects

https://www.linkedin.com/in/maaret/ @maaretp@mas.to

Applications are tester's external imagination.

* audience suggestions
* genAI suggestions

https://www.linkedin.com/in/maaret/ @maaretp@mas.to

A majority of the production failures (77%) can be reproduced by a unit test.

https://www.usenix.org/system/files/conference/osdi14/osdi14-paper-yuan.pdf
Through https://www.slideshare.net/Kevlin/the-error-of-our-ways

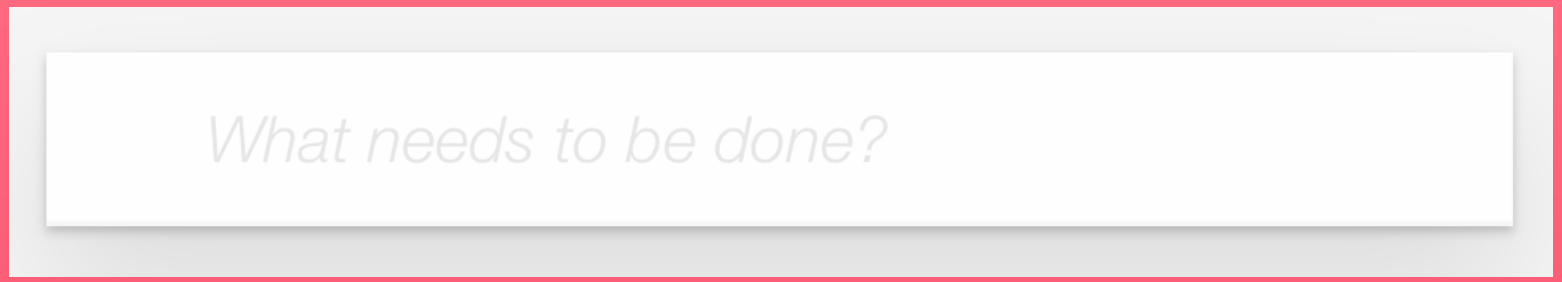
https://www.linkedin.com/in/maaret/ @maaretp@mas.to

Everything that does not need to be automated gets done while automating.

* Programming with thinking – quality of thinking matters

https://www.linkedin.com/in/maaret/ @maaretp@mas.to

How Would You Test This?



Visual ★ Functional ★ Purpose?

https://todomvc.com/examples/vanillajs/

No Todos
- ✓ should hide #main and #footer

New Todo
- ✓ should allow me to add todo items
- ✓ should clear text input field when an item is added
- ✓ should trim text input
- ✓ should show #main and #footer when items added

Mark all as completed
- ✓ should allow me to mark all items as completed
- ✓ should allow me to clear the completion state of all items
- ✓ complete all checkbox should update state when items are completed

Item
- ✓ should allow me to mark items as complete
- ✓ should allow me to un-mark items as complete
- ✓ should allow me to edit an item
- ✓ should show the remove button on hover

Editing
- ✓ should hide other controls when editing
- ✓ should save edits on enter
- ✓ should save edits on blur
- ✓ should trim entered text
- ✓ should remove the item if an empty text string was entered
- ✓ should cancel edits on escape

Counter
- ✓ should display the current number of todo items

Clear completed button
- ✓ should display the number of completed items
- ✓ should remove completed items when clicked
- ✓ should be hidden when there are no items that are completed

Persistence
- ✓ should persist its data

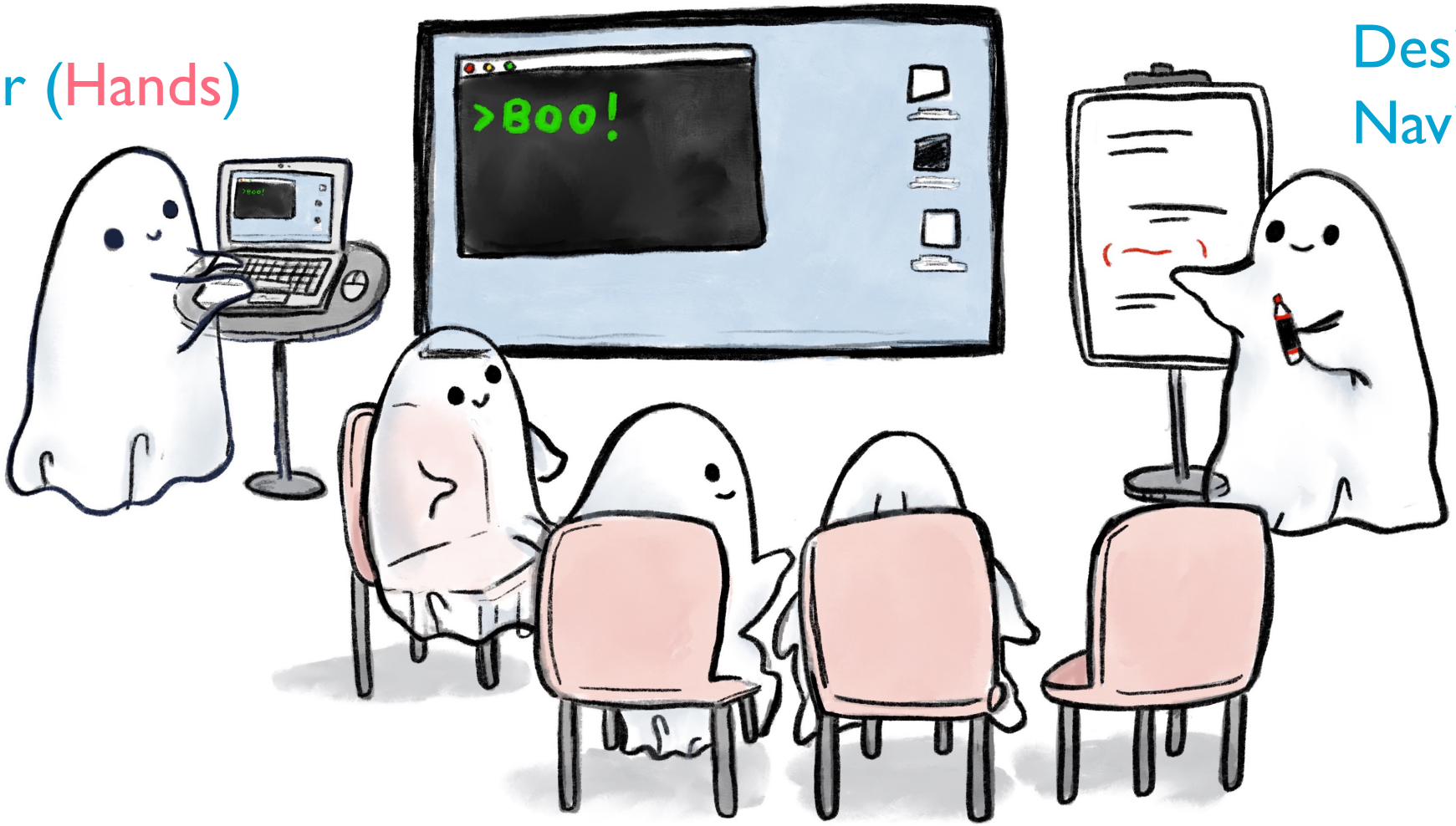
Routing
- ✓ should allow me to display active items
- ✓ should allow me to display completed items
- ✓ should allow me to display all items
- ✓ should highlight the currently applied filter

Developer Intent

https://www.linkedin.com/in/maaret/ @maaretp@mas.to

Driver (Hands)

Designated Navigator (Voice)



Navigators (Brains)

Agency. * sense of control

Pull. * timely for use

https://www.linkedin.com/in/maaret/ @maaretp@mas.to

Testing is too important to be left for testers. Testing is too important to be left without testers.

https://www.linkedin.com/in/maaret/ @maaretp@mas.to

1. **Observe**: notice more
2. **Ideate**: stop censuring
3. **Let go**: trust the process

4. **Be Impressed**: react
5. **Support others**: make them shine
6. **Do something**: experiment culture

7. **Yes, and**: combine

* Metaskills of Improv, Simo Routarinne, proimpro.fi

https://www.linkedin.com/in/maaret/ @maaretp@mas.to

Maaret Pyhäjärvi (from Finland)

EuroSTAR TESTING EXCELLENCE AWARD **2020**

AGILE TESTING DAYS **MIATPP** Most Influential Agile Testing Professional Person **2016**

tivi 100 ICT-influencers in Finland **2019 - 2023**

Email: maaret@iki.fi
Mastodon: @maaretp@mas.to
Web: maaretp.com
Blog: visible-quality.blogspot.fi
(please connect with me through Mastodon or LinkedIn)

#PayToSpeak #TechVoices
#EnsembleTesting #EnsembleProgramming #StrongStylePairing
#ExploratoryTesting #TestAutomation
#ModernAgile
#ContemporaryExploratoryTesting

Legacy maaretp

Exploratory Testing
https://exploratorytestingacademy.com

Ohjelmistotestaus ry

TECH VOICES
https://techvoices.org

Se TIVIA

https://www.linkedin.com/in/maaret/ @maaretp@mas.to