

1 2

What Makes a Specification "Good"?

- Software development is a business
 - Not just about coding business processes
 - Processes enable better code development
- Complex projects need multi-person teams
 - Lone programmers do simple contract work
 - Teams must have people working separately
- Processes are about how to break-up the work
 - What pieces to give each team member?
 - How can we fit these pieces back together?

Preconditions are a Promise

• If precondition true

Function must work

>>> to_centigrade(32.0)

- Function must work

0.0 >>> to_centigrade('32')

- If precondition false
 - Function might work
 - Traceback (most recent call last):

 TK

 File "<stdin>" line 1 in <module
 - Function might not
- File "<stdin>", line 1, in <module>
- Assigns responsibility
- File "temperature.py", line 19 ...
- Assigns responsibility
- $\begin{tabular}{ll} Type Error: unsupported operand type (s) \\ for : 'str' and 'int' \\ \hline \end{tabular}$
- How to tell fault?

Precondition violated

3

Testing Software

- You are responsible for your function definition
 - You must ensure it meets the specification
 - May even need to prove it to your boss
- Testing: Analyzing & running a program
 - Part of, but not the same as, debugging
 - Finds **bugs** (errors), but does not remove them
- To test your function, you create a test plan
 - A test plan is made up of several test cases

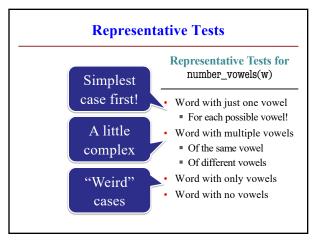
5

• Each is an input (argument), and its expected output

Representative Tests

- We cannot test all possible inputs
 - "Infinite" possibilities (strings arbritrary length)
 - Even if finite, way too many to test
- Limit to tests that are representative
 - Each test is a significantly different input
 - Every possible input is similar to one chosen
- This is an art, not a science
 - If easy, no one would ever have bugs
 - Learn with much practice (and why teach early)

1

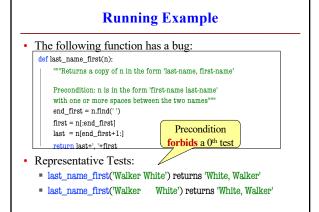


The Rule of Numbers

- When testing the numbers are 1, 2, and 0
- Number 1: The simplest test possible
 - If a complex test fails, what was the problem?
 - **Example**: Word with just one vowels
- Number 2: Add more than was expected
 - **Example**: Multiple vowels (all ways)
- Number 0: Make something missing
 - **Example:** Words with no vowels

7

8



Unit Test: An Automated Test Script

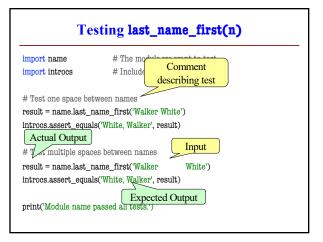
- A unit test is a script to test a single function
 - Imports the function module (so it can access it)
 - Imports the introcs module (for testing)
 - Implements one or more test cases
 - A representative input
 - The expected output
- The test cases use the introcs function

def assert_equals(expected,received):

"""Quit program if expected and received differ"""

9

10



11 12

2