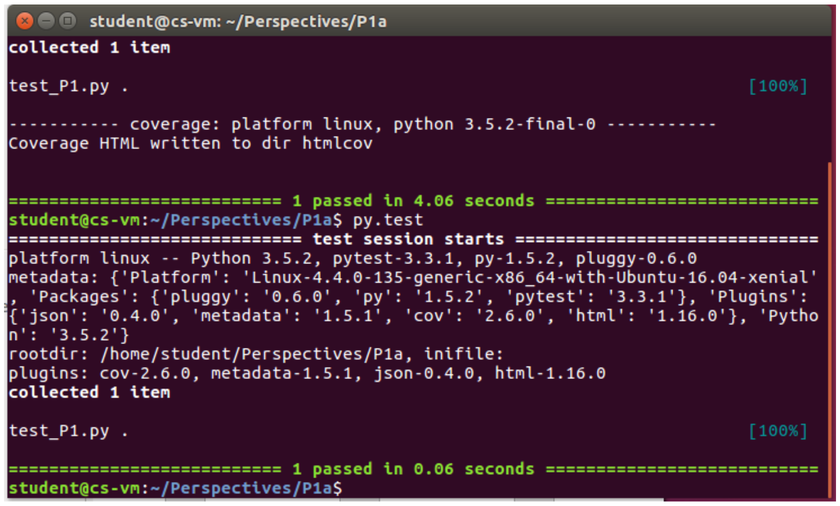
Dhruval Bhatt
Assignment 7
Perspective Analysis

Part 1: Problems 1 – 3 from Unit Testing Chapter

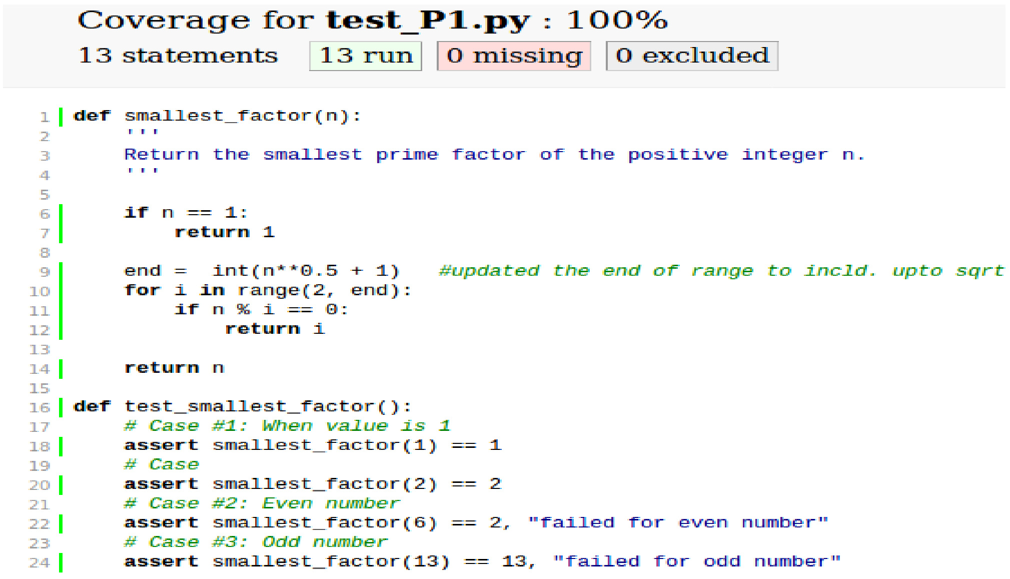
All python files are included in the folders as specified. A few screenshots are included to show important output.

Problem 1 – Folder name – P1a

In this problem, a test function is written to check for different cases such as 1, 2, even number and odd number. Upon running the test, it was evident that the program failed for numbers other than 1 and 2 which indicated an issue with the range. After fixing the range over which the loop is iterated, the test passes. A coverage check indicates that the test does cover all aspects of the program.



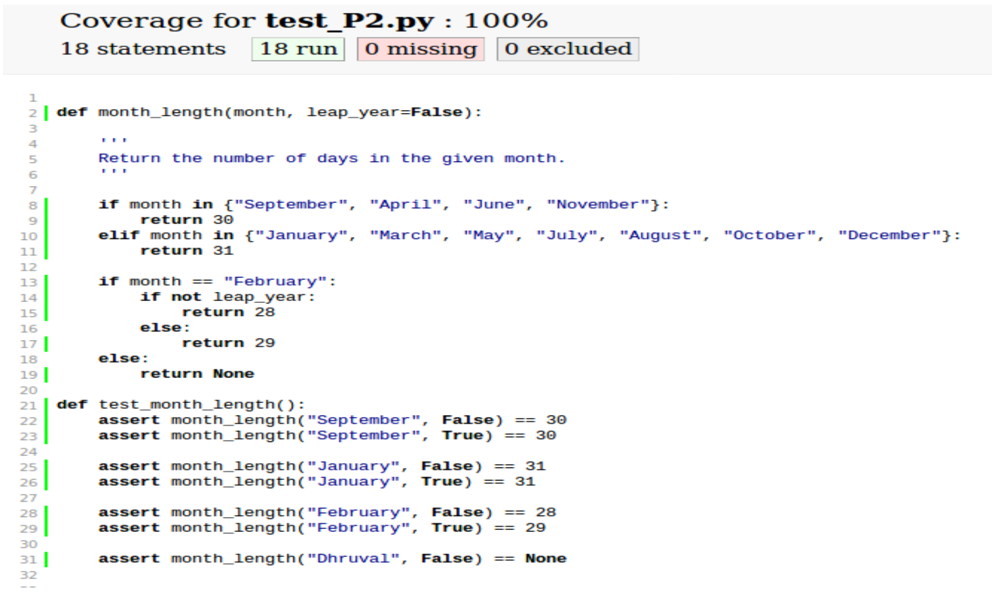
```
student@cs-vm: ~/Perspectives/P1a
collected 1 item

test_P1.py .                                                            [100%]

----------- coverage: platform linux, python 3.5.2-final-0 -----------
Coverage HTML written to dir htmlcov


=========================== 1 passed in 4.06 seconds ===========================
student@cs-vm:~/Perspectives/P1a$ py.test
============================= test session starts ==============================
platform linux -- Python 3.5.2, pytest-3.3.1, py-1.5.2, pluggy-0.6.0
metadata: {'Platform': 'Linux-4.4.0-135-generic-x86_64-with-Ubuntu-16.04-xenial'
, 'Packages': {'pluggy': '0.6.0', 'py': '1.5.2', 'pytest': '3.3.1'}, 'Plugins':
{'json': '0.4.0', 'metadata': '1.5.1', 'cov': '2.6.0', 'html': '1.16.0'}, 'Pytho
n': '3.5.2'}
rootdir: /home/student/Perspectives/P1a, inifile:
plugins: cov-2.6.0, metadata-1.5.1, json-0.4.0, html-1.16.0
collected 1 item

test_P1.py .                                                            [100%]

=========================== 1 passed in 0.06 seconds ===========================
student@cs-vm:~/Perspectives/P1a$
```



```
Coverage for test_P1.py : 100%
13 statements   13 run   0 missing   0 excluded

def smallest_factor(n):
    '''
    Return the smallest prime factor of the positive integer n.
    '''

    if n == 1:
        return 1

    end =  int(n**0.5 + 1)    #updated the end of range to incld. upto sqrt
    for i in range(2, end):
        if n % i == 0:
            return i

    return n

def test_smallest_factor():
    # Case #1: When value is 1
    assert smallest_factor(1) == 1
    # Case
    assert smallest_factor(2) == 2
    # Case #2: Even number
    assert smallest_factor(6) == 2, "failed for even number"
    # Case #3: Odd number
    assert smallest_factor(13) == 13, "failed for odd number"
```

Problem 2 – Folder name – P1b

This problem had the correct code and needed a good test to cover all cases of if-else-statement. By asserting the four major cases of the program, the unit test written could be verified to be 100% complete coverage.



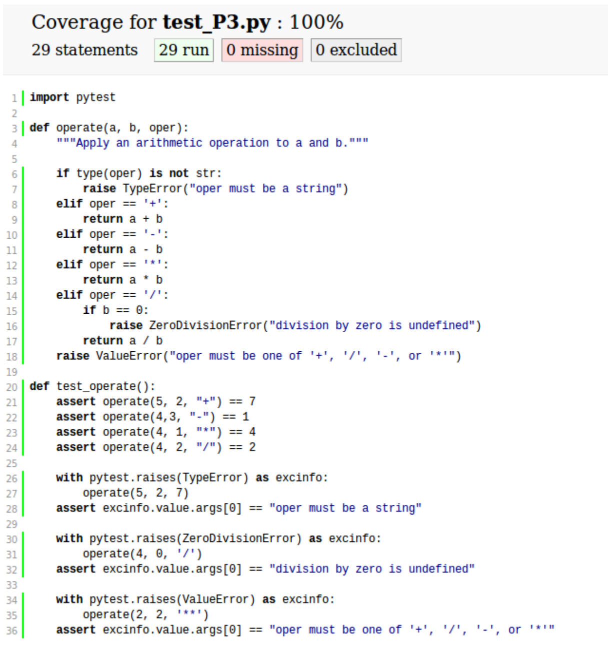
Coverage for **test_P2.py** : 100%

18 statements 18 run 0 missing 0 excluded

```python

def month_length(month, leap_year=False):

    '''
    Return the number of days in the given month.
    '''

    if month in {"September", "April", "June", "November"}:
        return 30
    elif month in {"January", "March", "May", "July", "August", "October", "December"}:
        return 31

    if month == "February":
        if not leap_year:
            return 28
        else:
            return 29
    else:
        return None

def test_month_length():
    assert month_length("September", False) == 30
    assert month_length("September", True) == 30

    assert month_length("January", False) == 31
    assert month_length("January", True) == 31

    assert month_length("February", False) == 28
    assert month_length("February", True) == 29

    assert month_length("Dhruval", False) == None

```

Problem 3 – Folder name – P1c

In this problem all the if-else cases are checked as well as all the exceptions. The test checks if an exception is raised any of the requirements are violated.



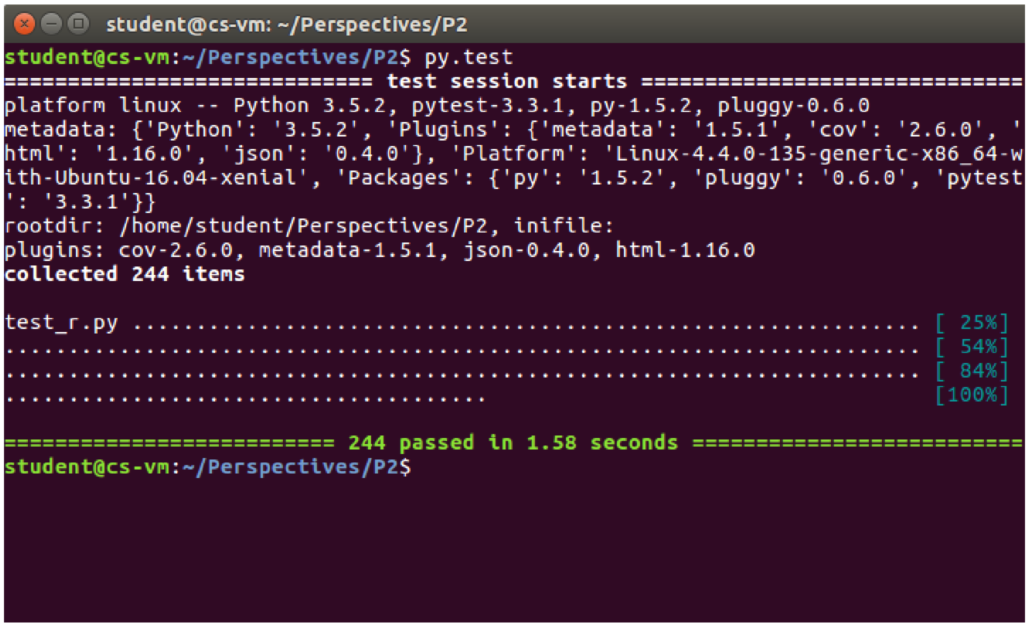
Coverage for **test_P3.py** : 100%

29 statements 29 run 0 missing 0 excluded

```python
import pytest

def operate(a, b, oper):
    """Apply an arithmetic operation to a and b."""

    if type(oper) is not str:
        raise TypeError("oper must be a string")
    elif oper == '+':
        return a + b
    elif oper == '-':
        return a - b
    elif oper == '*':
        return a * b
    elif oper == '/':
        if b == 0:
            raise ZeroDivisionError("division by zero is undefined")
        return a / b
    raise ValueError("oper must be one of '+', '/', '-', or '*'")

def test_operate():
    assert operate(5, 2, "+") == 7
    assert operate(4,3, "-") == 1
    assert operate(4, 1, "*") == 4
    assert operate(4, 2, "/") == 2

    with pytest.raises(TypeError) as excinfo:
        operate(5, 2, 7)
    assert excinfo.value.args[0] == "oper must be a string"

    with pytest.raises(ZeroDivisionError) as excinfo:
        operate(4, 0, '/')
    assert excinfo.value.args[0] == "division by zero is undefined"

    with pytest.raises(ValueError) as excinfo:
        operate(2, 2, '**')
    assert excinfo.value.args[0] == "oper must be one of '+', '/', '-', or '*'"
```

Part 2: Testing get_r()

Python files are found in folder P2. A screenshot is included to show important output. For this problem, we utilize the fact that python works on vectors so that for both scalar and vector case, the formula remains the same. The function module, get_r() is modified with the formula given and the test file is run to check if it passes the instructor requirement. The screenshot below shows that function meets the requirements.



```
student@cs-vm: ~/Perspectives/P2
student@cs-vm:~/Perspectives/P2$ py.test
============================= test session starts ==============================
platform linux -- Python 3.5.2, pytest-3.3.1, py-1.5.2, pluggy-0.6.0
metadata: {'Python': '3.5.2', 'Plugins': {'metadata': '1.5.1', 'cov': '2.6.0', '
html': '1.16.0', 'json': '0.4.0'}, 'Platform': 'Linux-4.4.0-135-generic-x86_64-w
ith-Ubuntu-16.04-xenial', 'Packages': {'py': '1.5.2', 'pluggy': '0.6.0', 'pytest
': '3.3.1'}}
rootdir: /home/student/Perspectives/P2, inifile:
plugins: cov-2.6.0, metadata-1.5.1, json-0.4.0, html-1.16.0
collected 244 items

test_r.py ............................................................... [ 25%]
......................................................................... [ 54%]
......................................................................... [ 84%]
.....................................                                     [100%]

========================== 244 passed in 1.58 seconds ==========================
student@cs-vm:~/Perspectives/P2$
```

Part 3: Reading and Responding to Watts' Paper

Duncan J. Watt's paper titled "Common Sense and Sociological Explanations" challenges the common notion that sociologists strictly utilize theoretical approaches versus common sense. The author proposes and illustrates that in fact, sociologist also build upon the notions and assumptions of common sense to theorize social action. However, this approach could lead to diminished ability to predict and stand the test of scientifically validity. On the other hand, if a strictly scientific method with assumptions and model is used, it may lose the intuitive sense that is commonly known and accepted. The author suggests, that as the field of sociology and data and resources available is changing and increasing, more sociologist will have to tradeoff between scientific method and use of common, intuitive sense.

Watt uses "Rational Choice Theory" as an example of how certain sociological theories have tried to use a scientific approach, in such that many observations are evaluated based on certain accepted theory. Using the rational choice as the basis to create precise models. The issue that arouse with studies using this approach was that that many theories deviated significantly from empirical results. Rational choice could not account for the limitations of reason and unknowns of future. It started to be evident that it was unreasonable to think that people acted to only maximize utility and consistently prefer a certain way or that they consistently determined their own course of action. As a response to this understanding, researchers shifted from scientific approach with predictability to more common-sense evaluation.

The author provides additional examples of common-sense theories to show that is not a one-off phenomenon but something that needs to be understood carefully as it has unintended consequences. Using this retrospective, sense making can help explain the world but not provide the basis for prediction. This means sometimes a theory could appear true when it is in fact false or not complete. The author goes on to effectively summarize why rationalizable action fails to predict causality. Human beings tend to imagine or simulate situations for answers to why something occurs or has occurred and may not be able to discriminate the information provided by the unconscious mind. While in real life situations when this mental simulation is employed for prediction, continuous feedback from reality helps update the involuntary assumptions but in generalized theory that does not occur as quickly and therefore leads to repetitive correction of idea, losing the very basis for being called theory. In addition, theories based on common sense could fall prey to “indeterminacy problem”. Individuals with different attributes could collectively have a different characteristic. This once again highlights the pitfall of using common sense in research. A researcher might try to generalize a theory based on individual perception but will not lead to accurate prediction for a collective. Finally, the author shows that the problem of prediction based on past experiences will persist due to what he calls, “the outcome problem”. This is the difficulty that arises in categorizing and recollecting outcome of action done in the past to determining action for the future.

Watts proposes that one way to address the challenges raised above is to make the researcher aware of the difference in empathetic understanding and causal explanation. He also suggests that more experiments should be conducting in the field that account for differences in natural conditions. In addition, he is a proponent of using models that are applied to large datasets that is gained non-experimentally. This includes the massive digital data and traces that is now available to researchers. Finally, the author encourages the understanding and appropriate definition of prediction and use that as a condition for causal mechanisms.

In this paper, Watts makes a compelling case for using more field experimental driven analysis in field of social sciences due to the shortcomings described above. However, even theoretical approach with outlined assumptions and set process have its merits too. In a strictly experimental approach, a researcher is susceptible to making implicit assumptions that could go unverified. They are likely to fit a theory to data available and may miss the generalizable theory whereas in a methodical approach, the researcher can consciously think about different cases and attempt to test details that might not be evident experimentally.