Notes and Pics for Lighting Round -- PyOhio 2018 -- Professor Reed -- verC

Notebook:	LightingRounds_PyOhio_2018_go		
Created:	7/27/2018 12:40 PM	**Updated:**	7/27/2018 2:11 PM
Author:	James Reed		
URL:	https://github.com/profjrr/python2018/blob/master/lighting_tests1_test2_and_test3		

Asserting better Programming--for now--and in the future

What has happened?

The state of affairs for computer programming is poor as illustrated by the Sad Face near the "current interest" in Computer Programming (see pic)

- Misunderstanding of what computer programming is and is not
- Is not -- WEB page design
- Is not -- gaming "per se" or playing on computer devices
- Is not -- WEB browsing or thumbing on mobile devices (see pic)
- It Is "so-called" heads down coding in a
  time tested computer language (C, C++, C# and Python)

Writing correct programs from the very start

How?

Assert everything from the very beginning
(follow NASA's lead!)

(Python has one of the best assertions statements in computer programming)
(Eiffel and Ada) to name a few others

Our assertions SHOULD NEVER EXECUTE!

(later we will assert x is greater than or equal to zero)

Current estimated error rates by NASA

- 99.99999% of errors times (2.5 million lines estd to 3.5m of

C code has a predicted error rate of less than 1 error
(reported as only 500,000 lines of C code -- see 7 mins of Terror)

And NASA is still looking for it (that one potential error)
of the successfully completed MARS Curiosity Landing

(again, see Seven Minutes of Terror)

Our theoretical business project of similar scope
and size has a predicted error rate of ***5k to 8k errors***
written in Python (see pic)

WHY assert?

NASA requires them!

We start out wrong and in a hurry to meet a schedule
or to keep our Boss happy etc

Asserting everything from the beginning makes us
think correctly from the project's earliest beginning and
forces us to test continuously before we have even
written the first line of substantial code (Python).

What NASA did was write the required C code for the Mars Curiosity Lander
and tested it with over 25,000 lines of Python Test Coding!
(NASA rule: once you include a test you can NEVER remove a test)
Asserts are required approximately for every 10 lines of C code at NASA

So how we doing?

Not very well -- so let's now look at my demonstration example in Python
to **solve x is greater than or equal to zero** **no matter what!**

AND NOW THE PROGRAMMING CODE IN PYTHON

```
# -*- coding: utf-8 -*-
"""
Created on Mon Jul 23 13:57:38 2018

@author: jrree
"""

## Starting out right with Assertions in Python
##asserts 2018 verC
##
##
### Python asserts presented by Professor Reed
### last updated (27-JUL-2018)
### for Python 3.4+
###Lighting Talks -- PyOhio 2018 -- ProfJRR
##
##
### NOW WITH ASSERTS!!! ###
import pytest

#float(i = 1)
#print("int(123) is:", int(123))

def mainasertions(x):
# x=1
# x=-1
# print("int(123) is:", int(123))
assert(x>=0),("### x is not greater than or equal to zero")
print("asserting x is greater than or equal to zero")
print("x is: ",x)
print()

## determines x is greater than a negative value")
#
##
### --- Professor Reed's test driver area --- ###
##
def testMyassertions(x):
### Part II ###
## int(x)
print()
print("=== Testing My Assertions ===")
print()
x=1
print("with a value of 1 for test #1")
mainasertions(x)
x=0
print("with a value of 0 for test #2")
mainasertions(x)
x=-1
print("xxx should never execute or occur xxx")
print("with a value of -1 for test #3")
mainasertions(x)
# final test or otherwise ###
```

```
print()

### Testing values and area follow: ##
##
#print("[negative value test1:]")
#x = -1
##x = 0
##x = 999
##x = 1011
testMyassertions(x)
##
##
### note: the test driver area above is a replacement for main()
##
### Professor Reed uses test cases instead of main()!
##if __name__ == '__main__':
## main()
```

Our Testing Results

```
=== Testing My Assertions ===

  Testing with an x value of 1

with a positive value of 1 for test #1
asserting x is greater than or equal to zero

x is:  1



  Testing with an x value of 0

with a value of 0 for test #2
asserting x is greater than or equal to zero

x is:  0



  xxx Testing with a negative x value of -1 -- never xxx

xxx should never execute or occur xxx
with a value of -1 for test #3

Traceback (most recent call last):
  File "<ipython-input-33-633d60e6e452>", line 1, in <module>
    runfile('C:/Users/jrree/.spyder-py3/lighting_code_verC.py', wdir='C:/Users/jrree/.spyder-py3')

  File "C:\ProgramData\Anaconda3\lib\site-packages\spyder\utils\site\sitecustomize.py", line 705, in runfile
    execfile(filename, namespace)

  File "C:\ProgramData\Anaconda3\lib\site-packages\spyder\utils\site\sitecustomize.py", line 102, in execfile
    exec(compile(f.read(), filename, 'exec'), namespace)

  File "C:/Users/jrree/.spyder-py3/lighting_code_verC.py", line 73, in <module>
    testMyassertions(x)

  File "C:/Users/jrree/.spyder-py3/lighting_code_verC.py", line 66, in testMyassertions
    mainasertions(x)

  File "C:/Users/jrree/.spyder-py3/lighting_code_verC.py", line 31, in mainasertions
    assert(x>=0),("### x is not greater than or equal to zero")

AssertionError: ### x is not greater than or equal to zero
```

THE BIG PICTURE

European Mars Lander Crashed Due to Data Glitch, ESA Concludes
Destroyed 19 October 2016

Current interest In Computer Programming ???

A bug in American Airlines' pilot scheduling software has left the US airline woefully understaffed for the coming month, particularly around Christmas week.

Program Correctness from the very beginning

NASA "mission critical" success rate and lack of errors

Curiosity Landed Successfully on **MARS**
August 6, 2012, 05:17:57 UTC
99.99999% error free
Program coding (.5 to 2.5m lines of code)
2.5m x 99.99999% ≈ 1 error
.5m x 99.99999% ≈ 1 error

Business errors best guess?
5K to 8k errors for a "commercial endeavor" of a project of similar scope

We're writing 5k or more errors for every 1 error that NASA produces

The "thumbers"!