FOCUS ON EXCELLENCE

Federal Institute
of Science and Technology

Unit Testing, Linting and Coding Standards

CS-308

Submitted To:
Dr. Prasad J C
Professor & HoD
Computer Science
Department

Submitted By :
Yedhin Kizhakkethara
FIT16CS125
S6-CSE-B
61

Contents

1 Intro

1.1 Objective

1. To know what exactly is unit testing, to know about its applications and to write some tests live(if time allows)
2. Introduction to what's linting and how you can extend it to your favourite text editor
3. Auto Formatters and naming conventions
4. All examples shown using **Python**

1.2 Libraries used

1. unittest (python inbuilt)
2. math (python inbuilt)
3. flake8 (linter)
4. yapf (auto formatter)

2 Unit Testing in Python

2.1 Start a project called areas

2.1.1 Creation of directories and files

```
mkdir -p areas/{areas,tests}
cd areas
touch areas/__init__.py areas/area.py tests/__init__.py
```

2.2 Create a function for finding area of circle

```
from math import pi
def circle_area(r):
    return pi * (r**r)
```

2.3 Run with sample values

```
radii = [0,1,0.1,2+3j,-7,True,'a']
for radius in radii:
    print(circle_area(radius))
```

2.4 Are all the radii valid?

2.4.1 Are negative values valid?

1. No. Since radii here are simply the value of distance from center of circle to its circumference. Thus it should be +ve.

2.4.2 Are data types other than int and float valid?

1. No. It doesn't make any sense to give radius as True or as String.

2.5 Solution : Proper Unit Testing

2.5.1 Start by importing the necessary modules

```
import unittest
from math import pi
from areas import area
```

2.5.2 Subclass unittest.TestCase and create appropriate tests

```
class TestCircleArea(unittest.TestCase):
```

2.5.3 Create a test to assert value of area function for different radii

Different Assert Methods

```
def test_radius(self):
    result = area.circle_area(1)
    self.assertEqual(result,pi)
```

2.5.4 Create a test to check negative values

We can raise ValueError on intercepting a negative value as radius

```
"""
in test file
"""
def on_negative_radii(self):
    self.assertRaises(ValueError,area.circle_area,-7)
```

```
"""
in area file, inside circle_area function, add the following
"""
if r<0:
   raise ValueError
```

2.5.5 Create a test to check for invalid data types

We can raise TypeError on intercepting a invalid datatype value

```
"""
in test file
"""
def on_other_types(self):
    self.assertRaises(TypeError,area.circle_area,"Hello")
```

```
"""
in area file, inside circle_area function, add the following
"""
if type(r) not in [int,float]:
   raise TypeError
```

2.6 Workout Problem

2.6.1 Create an unittest, which can be used with all programs done using Python Sockets, in Computer Networks Lab

1. Example : Most people manually set the IP Address of the system while creating a socket. So when changing the system, the IP will also change, and will have to be manually changed again. Write a test case to make sure the IP Address is always same as that of the system.

3 Linting

3.1 What's it?

1. It's the process of automatically checking the program code for errors

3.2 How can it be done?

1. via your editor(live linting)

2. via your build process

3. via pre-commit hooks in version control

3.3 Types of Linting

1. Syntax

   (a) Refers to the anti-patterns and missing of keywords etc in code

2. Code Style

   (a) It deals with enforcing proper naming convention and standards

3.4 Popular linters

Language	Linter
C	Clangd
C++	Clangd/CppCheck
Python	Flake8/Pylint
Java	CheckStyle
Shell Script	ShellCheck

4 Auto Formatting

4.1 What's it?

1. It's the process of automatically formatting the code to follow a particular style guide and coding convention

4.2 Why do we need to use it?

1. Enforcing common style guide is valuable while in a project with team

2. Code gets formatted with click of a button or command

3. No need to discuss style in code review

4. Saves time and energy

4.3 Popular formatters

Language	Formatter
C	Clang-Format
C++	Clang-Format
Python	Yapf/Black
Java	Google-Java-Format

5 Coding Standards and Style Guide

5.1 Visit pep8 org