Practical 7

Implementing coding practices in Python using PEP8.

An Introduction to PEP-8

The Python programming language has evolved over the past year as one of the most favourite programming languages. This language is relatively easy to learn than most of the programming languages. It is a multi-paradigm, it has lots of open source modules that add up the utility of the language and it is gaining popularity in data science and web development community.

However, you can use the benefits of Python only when you know how to express better with your code. Python was made with some goals in mind, these goals can be seen when you type “import this”

Indentation

When programming in Python, indentation is something that you will definitely use. However, you should be careful with it, as it can lead to syntax errors. The recommendation is therefore to use 4 spaces for indentation. For example, this statement uses 4 spaces of indentation:

```
if True:
print("If works")
```

And also this for loop with print statement is indented with 4 spaces:

```
for element in range(0, 5):
print(element)
```

When you write a big expression, it is best to keep the expression vertically aligned. When you do this, you'll create a "hanging indent".

```
1. value = square_of_numbers(num1, num2,

2. num3, num4)
3.
4. def square_of_number(
5. num1, num2, num3,
6. num4):
7. return num1**2, num2**2, num3**2, num4**2
```

```
8.
9. value = square_of_numbers(
10. num1, num2,
11. num3, num4)
12.
13. list_of_people = [
14. "Rama",
15. "John",
16. "Shiva"
17. ]
18.
19. dict_of_people_ages = {
20. "ram": 25,
21. "john": 29,
22. "shiva": 26
23. }
```

Maximum Line Length

Generally, it's good to aim for a line length of 79 characters in your Python code.

Following this target number has many advantages. A couple of them are the following:

- It is possible to open files side by side to compare;
- You can view the whole expression without scrolling horizontally which adds to better readability and understanding of the code.

Comments should have 72 characters of line length. You'll learn more about the most common conventions for comments later on in this tutorial!

In the end, it is up to you what coding conventions and style you like to follow if you are working in a small group and it is acceptable for most of the developers to divert from the maximum line length guideline. However, if you are making or contributing to an open source project, you'll probably want and/or need to comply with the maximum line length rule that is set out by PEP-8.

While using the + operator, you can better use a proper line break, which makes your code easier to understand:

You should use	You should avoid
total = (A + B + C)	**total = (A + B + C)**

Blank Lines

In Python scripts, top-level functions and classes are separated by two blank lines. Method definitions inside classes should be separated by one blank line. You can see this clearly in the following example:

```
class SwapTestSuite(unittest.TestCase):
"""

Swap Operation Test Case
"""
def setUp(self):
self.a = 1
self.b = 2

def test_swap_operations(self):
instance = Swap(self.a,self.b)
value1, value2 =instance.get_swap_values()
self.assertEqual(self.a, value2)
self.assertEqual(self.b, value1)

class OddOrEvenTestSuite(unittest.TestCase):
"""
This is the Odd or Even Test case Suite
"""
def setUp(self):
self.value1 = 1
self.value2 = 2

def test_odd_even_operations(self):
instance1 = OddOrEven(self.value1)
instance2 = OddOrEven(self.value2)
message1 = instance1.get_odd_or_even()
message2 = instance2.get_odd_or_even()
self.assertEqual(message1, 'Odd')
self.assertEqual(message2, 'Even')
```

The classes SwapTestSuite and OddOrEvenTestSuite are separated by two blank lines, whereas the method definitions, such as .setUp() and .test_swap_operations() only have one blank line to separate them.