Annika Singh

Elvina Shaimukhametova

Julia Zurek

CIS206 W01 Applied Programming

A programming language is a set of rules used to write programs that tell a computer what to do. It helps humans communicate instructions to computers. Every programming language has its own set of coding standards. In the case of Python, PEP 8 (Python Enhancement Proposal 8) provides rules for writing cleaner, more readable, predictable, and maintainable code. Following these standards benefits everyone involved in the project, as well as the end users of the product. Below are some key points from PEP 8.

1. Indentation. Users should use 4 spaces per indentation level and avoid using tabs for indentation. For example:

*# Aligned with opening delimiter.*

foo **=** long\_function\_name**(**var\_one**,** var\_two**,**

var\_three**,** var\_four**)** (Tamang)

2. Line Length. Limit all lines to a maximum of 79 characters. For long expressions over multiple lines, users should use parentheses for readability. Blank lines can be used in functions, sparingly, to indicate logical sections.

3. Imports. Imports should usually be on separate lines and are always put at the top of the file. For example:

|  |  |  |
| --- | --- | --- |
| # Correct:  import os  import sys | or | # Correct:  from subprocess import Popen, PIPE |

(Tamang)

4. Whitespaces. Users should avoid extra space and trailing whitespace anywhere. Users should use a single space before and after operators.

5. Comments. Comments that contradict the code are worse than no comments. Use comments to explain why.

6. Function and Variable Names. Type variables introduced in **PEP 484** should normally use CapWords, preferring short names like T, AnyStr, or Num. When declaring covariant or contravariant behavior, add the suffixes \_co or \_contra, respectively:

**from** typing **import** TypeVar

VT\_co **=** TypeVar**(**'VT\_co'**,** covariant**=True)**

KT\_contra **=** TypeVar**(**'KT\_contra'**,** contravariant**=True) (**Tamang)

Function and variables names should be lowercase, with words separated by underscores (snake\_case). This enhances readability, especially for longer names. For example,

def calculate\_sum(a, b).

In addition, avoid using one-letter variable names. Users should use descriptive names instead, even for simple functions.

Instead of: def calc(a, b):

Use: def calculate\_sum(a, b): (Tamang)

Moreover, users should avoid using Python keywords or built-in function names as variable names.

If a variable is intended for internal use within a class or module, prefix it with a single underscore (\_) to indicate that it is "private." For example,

\_internal\_data, \_temp\_value

Constants should be written in all uppercase.

In conclusion, by adhering these standards, users will not only improve the quality of code but also enhance development experience.

# References

Guido van Rossum <guido at python.org>, Barry Warsaw <barry at python.org>, Alyssa Coghlan <ncoghlan at gmail.com>. *https://peps.python.org/pep-0008/#introduction*. 05 07 2001.

Tamang, Ram. *https://ran-ramtamang.medium.com/python-coding-standards-f39571dfc872*. 8 August 2023.