Programming Foundations
Unit 3 – Effective Python programming

Objectives

- What is PEP (Python Enhancement Proposal)?
- Whitespace
- Spacing and Formatting
- Naming
- Conditional Statements and Expressions
- Enumerate
- Loops
- List
- Dictionaries
- Functions

What is PEP (Python Enhancement Proposal)?

PEP 8: The Style Guide for Formatting Python Code

- Consistent style improves code readability and approachability.
- Though not mandatory, adhering to PEP 8 is recommended.
- PEP 8 provides a wealth of details about how to write clear Python code.
- It’s worth reading the whole guide online (https://www.python.org/dev/peps/pep-0008/).

Whitespace

PEP 8 Guidelines for Indentation and Line Length

- Use spaces instead of tabs for indentation
- Indent by 4 spaces for each level of syntactic significance
- Limit line length to 79 characters or less
- Continuations of long expressions should have an additional 4 space indentation.

Spacing and Formatting

PEP 8 Guidelines for Spacing and Formatting

- Separate functions and classes with two blank lines in a file
- In a dictionary, no space between key and colon, and one space before the value if it fits on the same line.
- Use one space before and after = operator in variable assignment
- In type annotations, no space between variable name and colon, and use one space before the type information.

```
# correct format x: int

# wrong format x :int
```

Naming

PEP 8 Guidelines for Naming Conventions

- Functions, variables, and attributes should be in lowercase_underscore format
- Protected instance attributes should be in _leading_underscore format
- Private instance attributes should be in __double_leading_underscore format
- Classes (including exceptions) should be in Capitalized word format.

Conditional Statements and Expressions

PEP 8 Guidelines for Conditional Statements and Expressions

- Use if not somelist instead of if len(somelist) == 0 for checking empty containers or sequences
- Use if somelist to check for non-empty containers or sequences
- Avoid single-line if statements, for and while loops, and except compound statements, spread them over multiple lines for clarity
- If an expression can't fit on one line, surround it with parentheses and add line breaks and indentation to make it easier to read
- Prefer surrounding multiline expressions with parentheses over using the \ line continuation character.

enumerate

PEP 8 guidelines for using enumerate

- enumerate provides a concise syntax for looping over an iterator and getting the index of each item as you go.
- Prefer using enumerate over looping over a range and indexing into a sequence.
- You can supply a second parameter to enumerate to specify the number from which to begin counting (the default is zero)

loops

PEP 8 guidelines for using else statement with loops

- Python has special syntax that allows else blocks to immediately follow for and while loop interior blocks.
- The else block after a loop runs only if the loop body did not encounter a break statement.
- Avoid using else blocks after loops because their behaviour may not be intuitive and can be confusing.

List and Dictionaries

- Keep slicing concise by leaving out unnecessary start and end indexes.
- Slicing also allows for easy access to the boundaries of a sequence using negative numbers or omitting the end index.
- Additionally, when assigning to a slice, the original sequence will be modified even if the lengths of the slice and the assigned value are different.

Lists

- Slicing a list returns a new list with references to the original list's objects.
- For more efficient code, avoid using unnecessary start and end indexes when slicing.
- Slicing also allows for easy handling of out-of-bounds indexes, making it simple to access the front or back of a sequence (such as a[:20] or a[-20:]).
- Assigning a value to a sliced list will replace the range in the original sequence, even if the lengths are different.

Dictionary

- The sort method of the list type can rearrange the contents of a list by its natural ordering.
- The sort method doesn't work for objects unless they define a natural ordering using special methods.
- The key parameter of the sort method can be used to supply a helper function for sorting.
- Returning a tuple from the key function allows combining multiple sorting criteria together.
- The unary minus operator can be used to reverse individual sort orders for types that allow it.
- Combining many sorting criteria together by calling the sort method multiple times with different key functions and reverse values in order of lowest rank to highest rank.

Dictionary

- In Python 3.7 and later, iterating a dict instance's contents will occur in the same order as the keys were added.
- Dictionaries-like classes may not preserve insertion order, so it's important to be careful when using them.
- Three ways to be careful when using dictionary-like classes:
- Write code that doesn't rely on insertion ordering.
- Explicitly check for the dictionary type at runtime.
- Use type annotations and static analysis to require dictionary values.

Functions

- Functions can return multiple values by using tuples and unpacking syntax
- Functions can also use catch-all starred expressions for unpacking
- Avoid unpacking into 4 or more variables, instead use small class or named tuple
- Avoid using None to indicate special situations, use exceptions and proper documentation
- Use type annotations to indicate a function will never return None.

Functions

- Functions can accept a variable number of positional arguments by using *args in the def statement.
- You can use the items from a sequence as the positional arguments for a function with the * operator.
- Using the * operator with a generator may cause a program to run out of memory and crash.
- Adding new positional parameters to functions that accept *args can introduce hard-to-detect bugs.

Functions

- Function arguments can be specified by position or by keyword.
- Keywords make it clear what the purpose of each argument is when it would be confusing with only positional arguments.
- Keyword arguments with default values make it easy to add new behaviors to a function without needing to migrate all existing callers.
- Optional keyword arguments should always be passed by keyword instead of by position.

References

- Effective Python,2nd edition by Brett Slatkin