**Writing clean code: Meaningful names**

Use meaningful names.

* *Be descriptive and imply type*: For booleans, you can prefix with is\_ or has\_ to make it clear it is a condition. You can also use parts of speech to imply types, like using verbs for functions and nouns for variables.
* B*e consistent but clearly differentiate*: age\_list and age is easier to differentiate than ages and age.
* *Avoid abbreviations and single letters*: You can determine when to make these exceptions based on the audience for your code. If you work with other data scientists, certain variables may be common knowledge. While if you work with full stack engineers, it might be necessary to provide more descriptive names in these cases as well. (Exceptions include counters and common math variables.)
* *Long names aren't the same as descriptive names*: You should be descriptive, but only with relevant information. For example, good function names describe what they do well without including details about implementation or highly specific uses.

Try testing how effective your names are by asking a fellow programmer to guess the purpose of a function or variable based on its name, without looking at your code. Coming up with meaningful names often requires effort to get right.

**Writing clean code: Nice whitespace**

Use whitespace properly.

* Organize your code with consistent indentation: the standard is to use four spaces for each indent. You can make this a default in your text editor.
* Separate sections with blank lines to keep your code well organized and readable.
* Try to limit your lines to around 79 characters, which is the guideline given in the PEP 8 style guide. In many good text editors, there is a setting to display a subtle line that indicates where the 79 character limit is.

For more guidelines, check out the code layout section of PEP 8 in the following notes.

**References**

[PEP 8 guidelines for code layout](https://www.python.org/dev/peps/pep-0008/?#code-lay-out)