



Programming Style & Documentation

What is programming style

- ▶ Equivalent to formatting when writing
- ▶ **Good formatting makes things easy to read**

- ▶ We wouldn't do this

Was **ETHAN really--WoRkInG** or **wAs hE onthebeach**

- ▶ We would do something like this

If he was on the beach then Ken might be p**sed



Programming style

- ▶ Some formatting is so important that it becomes the rule
- ▶ But other formatting is more a matter of choice

- ▶ Do you prefer

If we has at the beach he would have a tan, have sand in his hair and would seem relaxed.

- ▶ Or

If we has working he would be pale, have had to grade our assignments on Sunday, and would seem tired.



A sample from one of my 1000 word essays from high school

▶ How about

The is the first sentence of my first paragraph. This is the second sentence of my first paragraph.

The is the first sentence of my second paragraph. This is the second sentence of my second paragraph.

▶ Or

The is the first sentence of my first paragraph. This is the second sentence of my first paragraph.

The is the first sentence of my second paragraph. This is the second sentence of my second paragraph.



Consistency in formatting
makes code faster to read and
easier to understand



Style guides

- ▶ **Descriptions of what is accepted by a group as good style**
 - ▶ Can apply broadly or narrowly



Style guides

- ▶ **Descriptions of what is accepted by a group as good style**
 - ▶ Can apply broadly or narrowly
 - ▶ In programming these are typically language specific



Style guides

- ▶ **Descriptions of what is accepted by a group as good style**
 - ▶ Can apply broadly or narrowly
 - ▶ In programming these are typically language specific
- ▶ **Not often used by scientists**
 - ▶ Who wants to spend time writing and using a style guide when they could be doing science
 - ▶ Makes other people's code hard to read
- ▶ **Fortunately Python has a general style guide**



"A universal convention supplies... maintainability, clarity, consistency, and a foundation for good programming habits too. What it doesn't do is insist that you follow it against your will. That's Python!"

—Tim Peters



Variable and function names

- ▶ **Use descriptive variable/function names**
- ▶ **Python style guide**
 - ▶ Names should be lower case with underscores added for clarity
- ▶ Rules like this allow different kinds of objects to be quickly distinguished
 - ▶ Constants should be in all caps
 - ▶ `BOLTZMANN_CONST = 1.38 * 10 ** -23`



Indentation

- ▶ Considered good style because this:

```
for val in range(min_val, max_val + 1):
```

```
    total += val
```

```
    if total > min_total_for_calc_prop:
```

```
        proportion = val / total
```

```
        break
```

```
return proportion
```



Indentation

- ▶ Is easier to understand than this:

```
for val in range(min_val, max_val + 1){  
total += val  
if total > min_total_for_calc_prop{  
proportion = val / total  
Break}}  
return proportion
```

- ▶ Fortunately for us Python makes us do this
-



Line length

- ▶ Have you noticed the red line in Wing?
- ▶ It indicates a line of code that is 79 characters long
- ▶ Python style is to have lines that are no longer than this
 - ▶ A lot of programming is still done on small screens (laptops)
 - ▶ And other windows in IDEs (like Wing) take up a lot of space
 - ▶ And automated line wrapping can be difficult to read
 - ▶ So Python style keeps lines short



Line length

- ▶ There are two ways to continue long lines in Python
- ▶ The preferred method is implied line continuation
 - ▶ If there is an (but no) then you can simply hit enter and keep typing
 - ▶ It is OK to add parentheses for this purpose if it looks OK
- ▶ Alternatively a \ at the end of the line tells Python to continue reading onto the next line



Line length: implied continuation

```
def count_n_base_occurrences(sequence, n):  
    """Find the number of times n bases occur contiguously in a  
    DNA sequence"""  
    sequence = sequence.replace('\n', '')  
    number_of_occurrences = (string.count(sequence, 'a' * n) +  
                             string.count(sequence, 't' * n) +  
                             string.count(sequence, 'g' * n) +  
                             string.count(sequence, 'c' * n))  
    return number_of_occurrences
```



Line length: backslash

```
def count_n_base_occurrences(sequence, n):  
    """Find the number of times n bases occur contiguously in a  
    DNA sequence"""  
    sequence = sequence.replace('\n', '')  
    number_of_occurrences = string.count(sequence, 'a' * n) + \  
        string.count(sequence, 't' * n) + \  
        string.count(sequence, 'g' * n) + \  
        string.count(sequence, 'c' * n)  
    return number_of_occurrences
```



Basic Style

- ▶ Descriptive function/variable names
- ▶ Indenting
- ▶ Line length





Undocumented Code



Documenting your code

- ▶ Documentation helps other people use your code
 - ▶ And it helps you remember how to use it quickly
- ▶ **Easier to document as you go**
- ▶ Two major kinds of documentation
 - ▶ General
 - ▶ What the code does and how to use it
 - ▶ Detailed comments
 - ▶ Why the code is written the way it is



Documenting your code: General

- ▶ General description of how the code works at the beginning of the program or function
- ▶ Lets user determine what the code does and how to use it

range(...)

range([start,] stop[, step]) -> list of integers

Return a list containing an arithmetic progression of integers.

range(i, j) returns [i, i+1, i+2, ..., j-1]; start (!) defaults to 0.

When step is given, it specifies the increment (or decrement).

For example, range(4) returns [0, 1, 2, 3]. The end point is omitted!

These are exactly the valid indices for a list of 4 elements.



Documenting your code: General

- ▶ In Python this is done using docstrings
- ▶ At the beginning of every module and program
- ▶ At the beginning of every function

- ▶ “should be sufficient for a new user to use the command properly, as well as a complete quick reference to all options and arguments for the sophisticated user.”

- ▶ Always enclosed in triple double quotes
- ▶ `“““This is my one-line docstring”””`



Documenting your code: Detailed

- ▶ Describes difficult to understand parts of the program

`#removes newlines to find contiguous sequences across lines`

`sequence = sequence.replace('\n', '')`

- ▶ Don't comment the obvious
- ▶ If you had to think hard about the code or a bug was difficult to find/fix then a comment is likely needed
- ▶ Can be used to note changes to be made using things like `#TODO` or `#FIXME`

