COMP1730/COMP6730
Programming for Scientists

Code Quality

Lecture outline

- Why should we care about code quality?
- Aspects of code quality.
- Standards and guidelines.

Why should we care about code quality?

- Code quality is primarily for people.
- Example 1: the interpreter doesn't care what you call your variables, as long as they are valid names, but giving them useful names makes your code much easier to read and understand.
- Example 2: python ignores comments, but they can also make code much easier for a human to understand.

(Extreme) example

- What does the following code do?

```
def AbC(ABc):
    ABC = len(ABc)
    ABc = ABc[ABC-1:-ABC-1:-1]
    if ABC == 0:
        return 0
    abC = AbC(ABc[-ABC:ABC-1:])
    if ABc[-ABC] < 0:
        abC += ABc[len(ABc)-ABC]
    return abC
```

Australian National University

(Extreme) example - continued

- What does the following code do?

```
def sum_if_negative(input_list):
    '''Sums up all the negative
    elements in input_list.'''

    total = 0
    for number in input_list:
        if number < 0:
            total = total + number
    return total
```

What are the aspects of code quality?

Australian National University

Aspects of code quality

* Commenting and documentation.
* Variable and function naming.
* Code organisation.
* Code efficiency (somewhat).

What makes a good comment?

* <u>How</u> or <u>why</u> not <u>what</u>.
* Parameters and assumptions - python is not a typed language.
* Kept up-to-date and next to the relevant piece of code.
* More important when learning to program or working with other people.

Docstrings

* A way to formally document functions, classes and modules.
* A triple quoted string as the first item inside a function definition, module, or class definition.
* Used to document the purpose, as well as any assumptions that are required for the code to work correctly.
* Can be read using the <u>help</u> function in python.

Function docstrings

- Are a way of stating type requirements and purpose of function parameters and return values in python.

```
def get_rows(sudoku_state):
    '''Gets the rows of the sudoku
    :param sudoku_state:  the state
    of the sudoku - a list of lists.
    return:  a list of the rows
    each element is a list.'''

    ...
```

What makes a bad comment?

- Stating the obvious.
  ```
  x = 5 # Sets x to 5.
  ```
- Used instead of good naming.
  ```
  x = 0 # Set the total to 0.
  ```
- Out-of-date, separate from the code it describes, or flat out wrong.
- More comments than code is (usually) a sign that your program needs to be reorganised.

Rules of thumb for commenting

- Do document functions, classes and modules using proper docstrings.
- Don’t use comments as a substitute for good practice in other areas of code quality (organisation, naming, etc.).
- A good starting point is one comment for three to four lines of code (very rough guide).

Variable naming rules (recap)

- Can only contain letters, numbers and underscores '_'.
- Must start with a letter or underscore.
- Are case sensitive.
- Cannot be a reserved word (`def`, `if`, `class`, etc.).

Variable naming good practice

- Should tell you what the variable is being used for.
- Should not <u>shadow</u> a name in a greater scope, i.e. don’t use `list` as a variable name.
- By convention, variables that start with an underscore, have a special purpose in python. Don’t use them unless you are adhering to this convention (private attributes of classes).

Variable naming good practice (cont.)

* Can be long (within reason). Most IDEs will autocomplete them for you anyway. `sum_of_negative_numbers` is perfectly fine, and much better than trying to abbreviate it to `sonn`.
* Should not be very similar to other variable names. `sum_of_all_negative_numbers` is OK, but shouldn't be used with the above, since it's not clear what the difference between them is.

Question

* Is it ever OK to use 1 character variable names, such as `i` or `x`?

Answer

- Yes, as long as their meaning and usage is clear.
- `i` is often used as a shorthand for `index` in `for` loops.
- `x`, `y` and `z` are often used as variables for coordinate systems.
- It should generally be avoided though if the variables have a broad scope or will be used in many different places.

Organisation

- We can organise code by separating it into functions and modules (files).
- You can import your own files in the same way you import built-in modules. `import homework_3` imports the python file named ‘homework_3’. Thereafter any functions in that file can be used by writing: `homework_3.function_name(...)`.
- Well organised code and well structured code is far easier to follow, maintain and extend.

Using functions to organise your code

- A good function does <u>one</u> thing.
- A good function works at one level of abstraction.
- Functions promote abstraction, i.e. they separate the <u>what</u> from the <u>how</u>.
- Functions make code easier to maintain. If there is no repeated code in your program (it’s all encapsulated in functions) you only need to make a change in a single place.

Modules

* You can organise your code by placing functions that are related to each other into modules.
* For example all your user input handling functions might go in one module, all your database interaction functions in another, your data analysis in a third, etc.
* Keeping related functions together helps users (and you) quickly determine where to look for certain types of functionality.

Australian National University

Tips for using modules

- Make sure you give the module a descriptive name.
- If there is code that should be run from within the module, (testing code, debugging code, etc.) make sure to put it in a:
  `if __name__ == '__main__'` suite. This allows the module to be imported without running it.
- If your program is very large or very complex, it's a good idea to include a readme file explaining how to use your program, or where to start.

Efficiency

* Modern computers will usually have enough power to solve most problems, even if the code is not perfectly efficient.
* Programmer time is usually far more expensive than computer time.
* This means that program readability and clarity is (usually) more important than optimisation.
* It is not worth spending hours of your time to shave a few seconds off the running time of a program that you will only run once!

Where efficiency is important

* If your program is going to be run frequently, or by lots of people, any performance improvements can add up over time.
* If your program is too slow to run at all. A poor choice of algorithm or data structure may prevent your program from ever finishing.
* Where the efficient approach is just as readible or maintainable as the inefficient one. For example - don’t read through a file one time for each attribute. Don’t convert a string to an integer and then to a float.

Australian National University

Standards and Guidelines

Why have standards and guidelines

- They are particularly useful for group software development.
- Some decisions can prevent code from running at all (i.e. different indentation).
- Make code more readible and maintainable.
- New people on the project can contribute much faster.

What do standards and guidelines cover

* Documentation and commenting styles.
* Variable naming conventions (`CamelCase`, `lower_with_underscores`, etc.)
* Code organisation and structure.
* Unit testing and debugging procedures.
* Many other aspects.

Standards in python

- Many python specific coding conventions are described in PEP (Python Enhancement Proposal) 0008-Style Guide for Python Code, which is based on the assumption that code is read more than it is written. The PEP can be found on the python website.
- The worst standard is no standard at all (or multiple different standards). Even if you don’t completely agree with the standard your team is using, it is better to follow it than to do something different.