Comments: Explain why you did it that way. Code says what and comments say why

-can write a comment before each function to explain what will happen and what will be returned (if any).

-some IDEs can use special formatted comments for tooltips.

-Write comments as you develop/write the code because that’s when the code is fresh in your mind and you know what’s going on. You might forget why you did something if you when review your code down the road.

-Don’t over comment, write simple and easy to read code to reduce comments. Ex:



- If your code isn’t really self explanatory then write short one line comments to explain

Consistent style: Easier to read code when its all the same style

-Indentation:

-There is more than one way of indenting code:



-Name Scheme:

-popular options:

-camelCase: First letter of each word is capitalized, except the

except the first word

-underscores: Underscores between words, like: some\_dumb\_function()

if you are starting a project form scratch pick a style and stick with it

if you are working with other, agree on a style and stick with it

if you are working on someone else’s code, stay with there style

No best style. Just be consistent.

Code Grouping:

Certain tasks require more than one line of code. In this case it is a good idea to separate these blocks of code with some spaces. Adding a short one line comment at the beginning of each block of code can help visual the separation.

No DRY Code:

-DRY = Don’t Repeat Yourself, DIE= Duplication Is Evil

- the same chonk of code should not be repeated over and over again.

- if there is repetition in your code, the repeated code can most likely be put into a separate function that can the called where ever needed

No Deep Nesting:

-too many levels of nesting are make code hard to read.

-most of the time can rearrange code to reduce nesting.



Multiple Files:

-even though you can write a big project all in one file, it is a very bad idea to do so. The solution is to split the project up into separate files using a good file structure to organize them.

Naming:

-Make sure variable and function/method names are descriptive and meaningful. As mentioned in the section about comments, well named variables and functions reduce the need for comments

-one exception is Temporary variables. These can be seen more commonly in for and foreach loops. Even though these temp variables are usually named using are a single letter or two they should stay consistent. i.e if you use i in one forloop you should use it in all your for loops (unless you have a for loop nested in a for loop. In this case use another letter like j and use that for all other for loops nested in for loops).

-common letters used for some of these temp variables are:

i,j = loop index

k = keys, v = values