Style guidelines: Python

Code style is based on the PEP-8 standards, available here for reference:

https://www.python.org/dev/peps/pep-0008/

With the following exception:

* indentation will be with a single **tab** character. Ensure your editors are configured correctly, as some replace tabs by a variable number of spaces

Linter will be used to assess how well each file conforms to these standards, which will be available in the Utilities folder on github. Most IDEs (Integrated Development Environments) can also be configured to automatically look for most of these conventions and mark them as the code is written. Talk to a senior member of the software department if you would like help setting this up.

Code should be well commented, but make sure that comments are always up-to-date and useful. Do not include comments that simply restate the code:

* *Bad*: x += 1 # increment x
* *Good*: x += 1 # account for picture border

*Note to self: update linter to use a .txt file for input*

Naming conventions are as follows:

* **Files and folders** are all lowercase, with words separated by underscores
* **Classes** use the CapWords convention (note the capital on the first word as well)
* **Functions** are lowercase, with words separated by underscores.
* **Variables** use the capWords. One letter variables are okay as long as their function is obvious (ex: i and j for outer and inner loop counters, y and x for coordinates)
* **Global variables** use the “functions” naming convention

Every file must have a set of unit tests, present either as the “main” method or as a separate file. These must cover each method, going through as many logic branches as possible.

Every function must have a docstring. Very simple methods (like “setter” methods) can use a single line docstring, but otherwise they should be formatted as below, clearly indicating the parameters and returns:

*"""*

*Computes and displays the discrete NxN cosine transform of an image*

*Parameters*

*----------*

*img : image*

*the input image*

*N : int*

*the size of each block (NxN) computed*

*displaying=true : boolean*

*if true, displays the image and waits for a keystroke before continuing*

*Returns*

*-------*

*result : image*

*the image after the cosine transform*

*"""*

For general docstring guidelines, see PEP-257:

https://www.python.org/dev/peps/pep-0257/

Style guidelines: Arduino

For consistency, code style and naming conventions should generally match the Python guidelines above. However, there are a few concerns specific to arduino programming:

* All pin numbers should be defined at the top of the program. These names should be short and descriptive, in all caps with words separated by underscores
  + Digital names will be just the name (ex: LED\_PWR). Analog names will start with “A\_” (ex: A\_TEMP1).
* All pieces that may need to be replicated (i.e. motor drivers) will be created as objects. The constructors of these modules will take pin numbers as arguments

*Note to self: build a sample arduino light class and program*

Style Guidelines: C++

For consistency, code style and naming conventions should generally match the Python guidelines above. However, there are a few concerns specific to C++ programming:

* Create a header (.h) file for each source (.cpp) file.
* All headers must be protected against multiple inclusions using #ifndef guards:

#ifndef TEST\_FILE\_H

#define TEST\_FILE\_H

...

#endif

* The namespace “cv” is commonly used in files relating to image recognition. Other namespaces (including “std”) should be used with discretion.