Documentation

In this page, we'll outline the documentation guidelines that you're expected to follow in this course, based on both the Python standard PEP 257 and the function design recipe of CSC108. Keep in mind that we'll be marking your documentation on **assignments** and possibly exams, but not on your **exercises**. However, we strongly encourage you to get in the habit of writing good documentation early so that becomes second nature to you.

Classes

Every class should have a docstring with the following format:

- A one line summary of the class.
- A longer description containing notes and brief implementation details.
- A list of all attributes of the class, their types, and their purpose.

For maximum readability, each of these should be separated by one blank line. Even though it would be helpful, you are not required to list all of the public class methods explicitly. However, it is strongly recommended to mention the most important of them in the longer description.

Here's an example:

```
class Point:
    """A point in the 2-D plane.

    The class represents a point in rectangular (x,y) coordinates,
    and supports standard geometric transformations like translation and
    rotation.

    Attributes:
    - self.x (number): the x-coordinate of the point
    - self.y (number): the y-coordinate of the point
    """
```

Functions and Methods

Every function and method you write should have a docstring with the following format:

- A **type contract** describing the types of the parameters and return value.
    - If a parameter/return value could be any type, use `object`
    - If a parameter/return value could be any numeric type, use `number`; otherwise, use `int` or `float`
    - If a parameter/return value is a compound type (like list, dict, or tuple), try to specify the types of the components, if possible: `list of str`, `dict of number`, `tuple of (float, float)`. If the components could be any type, you can omit them and just write `list` or `dict`.
    - For methods, include the `self` parameter, whose type is the name of the class

    - If no value is returned, the appropriate return type is `NoneType`.

- A **description** of what the function does, written as *commands*: `Return blah` rather than `Returns blah`.
- A list of parameters and their purpose. `self` may be omitted here.

Again, for readability, separate the list of parameters by a blank line. Here's an example for a `distance` method in the `Point` class:

```
    def distance(self, other):
        """ (Point, Point) -> float
        Return the distance between self and other.

        Parameters:
        - other: the point to calculate the distance from
        """
```

Modules

This part is less important, because we'll generally be giving you starter code with the docstrings already written. But just for your reference, here are the guidelines that we're trying to follow:

- One line summary of module.
- List of all classes and functions provided by the module (including exceptions), with *very brief* description.

David Liu (liudavid at cs dot toronto dot edu)
Come find me in BA4260
Site design by Deyu Wang (dyw999 at gmail dot com)