**Assignment 1: code review**

**For easy reference, the first four are repeated below – those metrics pertain to the more basic architectural setup of the code base, which in general will suffice.**

**Write short units of code**

**Write simple unites of code**

**Write code once**

**Keep unit iterfaces small**

**Before this more concrete analyses, make sure you give a thorough desciption of the code base you received: what is it supposed to do, how well is it organized and how would it scale and be maintainable?**

**Also, provide at least one suggestion for a not trivial refactor. Make sure you describe the code as it is and why you propose this refactor. In what way will this improve the code base in your opinion?**

I received two Python files (average\_month.py and average\_year.py) from my classmate, Jacob. I would like to add the following remarks summarized in the table below. I believe that the codes seem not to be completed and I did not receive other classes (e.g., Reader) that may be necessary.

|  |  |
| --- | --- |
| Faults | Suggestions |
| def get\_data: It is used in both classes. | It can be used in the class reader |
| def get\_averages: the average is used as the method, but it is re-writed in another method as def all\_in\_one | If there is an average method, just call it another method |
| column += 5 ?? In the average\_month he used 5 as a hardcode! The code cannot be changed based on the new data. | Replacing 5, could be an attribute in the \_\_init\_\_ and just calling that variable would be enough |
| In the \_\_init\_\_ of the classes, there is just reader. There could be other attributes that can decrease hard coding and repetition. | There were several options that can be in the init, for example, the average year/month can be in init, but None, and then update. |
| The classes have multi-tasks: reading the data, calculating the average, and plotting. | It is better to make classes that follow single responsibility principle |