Test Driven Development

Basic idea:

Test driven development is basically just testing everything that you write. The way to do it is by thinking of what you're going to write and making a test for it. This test must fail. Now that you have a failing test you write the actual code, you keep writing the code until it passes the test. One of the key parts of this technique is that you never write more than is needed. Once you have a failing test you go to the next step, you don't try to keep adding to it. Once your functional code passes the test you stop. After you having working code that passes a test you refactor it, make it nice and clean. Rinse and repeat until all coding has been done (in general) .

Benefits:

The benefit of writing code in this manner is the fact that you get code with minimal bugs (generally). With the short testing cycle you get positive feedback from every line of code, you know that it works properly. By also testing small sections of the code it is easier to see where the errors are. Someone who writes large chunks of code and then tests it later may have the problem of the code being inherently hard to test, in most cases TDD forces you to code in a way that is testable. Making changes are safer as the tests are already there and you will know if you have broken it, this also goes for the addition of newer code (it could cause older tests to fail), the sheer number of tests as the program grows helps to keep the code correct. With documentation of code the tests are pretty much examples of how the code should be used. This could be useful with our project as it will probably be split up into multiple parts, the tests will make it easier to use and understand everyone else's code.

Things that can be done in TDD:

There are two different types of TDD, The smaller developer test that handles the smaller code chunks and the larger acceptance test which tests a larger section (module) of the code. These two types of TDD can be used together with each part of code added to get the acceptance test to pass going through its own smaller developer test.

During the project the number of tests can be quite large and the idea is to keep the running of the tests quite fast to keep the pace of the programming up. If the number of tests becomes too large it can slow down the pace so having multiple levels of tests with older tests being run with larger intervals between can keep the pace up

Unit tests

PyUnit or unittest is a testing framework that is part of the python language. We should be able to use this during our project to achieve some level of TDD