Test Plan

Testing process:

To test our methods, we will extract test values from a properly formatted test case JSON and then put the results to an HTML page. Our testing framework tests div, factorize, factorial, mod, and pow. Each of these methods originate from the functions.py section of the SugarLabs codebase. The computed values will be compared to the predetermined oracle value for validity.

Requirements traceability:

We will be testing to ensure that we cover a wide range of testable operations by selecting a diverse range of parameters. This range will include positive, negative, overflow, and varying data types.

Tested items:

We will be testing the div, factorize, factorial, mod, and pow methods. Within these methods we will test them for overflow, type errors, positive, and negative values. Each method has its own requirements to be tested that will be accounted for in each test case. For example, the div method can allow for a division by zero, so we have created a test case to account for that.

Testing schedule:

Testing will be done according to the schedule outlined within the deliverables. At each stage of testing, all previous tests will be performed to ensure the correct operation of each method. If testing should be come strenuous on any testing systems, then testing will be performed during off hours to lighten the load on the machines.

Test recording procedures:

Our testing results will be output to an HTML file complete with test case id, method tested, oracle, actual value, and whether the test case passed or failed.

Hardware and software requirements:

The framework is tested within Ubuntu 16.04. Python 2 or higher is required on the machine to operate the testing framework.

Constraints:

The framework is constrained by the existing test cases. Each new method requires a new driver and a new specification within the framework to run the new driver.

System Tests:

The system should test the functionality of the each of the selected methods by utilizing a range of input. Every test case specified in the test case fold should be tested in a random order, assuming they are properly formatted. When new test cases and drivers are added, all previous test cases are to be tested.