This is why I commit to Python and rely on it to help me succeed.

Firstly, we both have used Python the most and feel the most comfortable with the syntax and execution. We will use PyCharm as our primary IDE of choice. Furthermore, we feel that Python in particular would be appropriate for the assignments on hand due to its powerful dynamic-casting abilities as well as its vast collection of libraries for coding use.

Python allows for execution on the level of the Unix command line. A widely used methodology is to run Python code through an interactive session, which can be opened on the command-line by typing in "python", or "python3" depending on installation, and then hitting enter. Furthermore, with Python's sys module, Python is capable of reading from STDIN and writing at STDOUT by providing us with the appropriate file object. Through the use of the JSON module, Python is able to decode/encode (built-in functions) the JSON file into Python's pre-existing data structures, allowing for reading, printing, and processing. As Python's socket module provides an interface to the Berkeley sockets API, TCP/IP sockets functionality is allowed by Python as well. Furthermore, through the importing module ability outlined above, Python allows for dynamic loading through the modules (which have built-in functions/objects coded inside the module, along with potentially other modules).

For unit testing, Python has a built in package called "unittest" which supports many different features of unit testing including test automation and aggregation. These commands can even be run from the command line. As for random testing facilities, Python also has built-in pseudo-random number generating libraries which can provide randomness for testing. Furthermore, Python also contains third-party libraries such as "pytest-randomly" which can be used to shuffle the order of test items like modules, test classes, and doctests.

Discussed previously, Python has numerous IDE's, of which we will be using Pycharm. Pycharm has great features for code completion, error highlighting, refactoring, navigation, and much more. Because both my partner and I are on different OS's, the cross-platform nature of it is also very useful.

Python has an REPL which can be accessed either through the IDLE or on the command line, and it also has built in ways of importing and downloading packages through installers like pip.