Simeon Patton

CS362 Oregon State Spring 2021

Homework 4

Question 1;

i) Submitted files;
cube_volume.py
test_cube_volume.py

can be found at;
https://github.com/pattons-OSU/CS362_Homework/tree/master/Week_5

ii) In my testing of the simple cube app, I chose to test the arithmetic, input of a negative number, and input type. I chose these three because they seemed like the most relevant tests to run on a cube. Without knowing that these items are working, we would not know for sure that the program is giving us the correct values out. The arithmetic literally tests to make sure that the math is correct and not squaring instead of cubing or something of that nature. The test for a negative number is really in two parts. There is an inline test in the original program that looks for a value lower than 0 and then continues or doesn't and then in the unittest suite it builds upon that and looks for a raised value error (some value lower than 0). Last, the type test is making sure that the user input is a type of int. If this was not the case then the program would fail and could not perform work on a non-integer type.

Question 2;

i) Submitted files;
average_list.py
test_average_list.py

can be found at;
https://github.com/pattons-OSU/CS362_Homework/tree/master/Week_5

ii) With the average program I chose to perform tests that included making sure that the list is what was expected, making sure that the list was not empty/and full, and testing the arithmetic on the average function itself. First, a hard coded example of the list is written into the unittest module and is compared against the expected output of the average program. Next, in order to make sure that the list is full, we check the Boolean output of a list. If a list is "true" then it has elements within the list. The same is true for the reverse, if it is false then the list is empty. Last, I tested to make sure that the average function was calculating the expected output by comparison.

Question 3;

i) Submitted files;
name_gen.py
test_name_gen.py

can be found at;
https://github.com/pattons-OSU/CS362_Homework/tree/master/Week_5

ii) For the name gen program, I had a hard time thinking of things to test. I did end up coming up with a type test, and expected output, and a length test. First, the type test was there to make sure that the user inputted a string and not a different type of input. This made sure that the forthcoming string concatenation functioned correctly. Next, the expected output test made sure that the “full_name” function was correctly adding the two user inputs together. Because the assignment asked us to add two stings, we needed to make sure that is what was happening. Last, I thought that an easy way to test if the user even entered anything was to test the length of the string itself, making sure that the program is not adding empty names.

Question 4;

i) In the in-class activity 1, I chose to showcase Facebook as a social media platform. Found within Facebook is a pretty robust messaging system. This system allows users to send a chunk of text from one user to another through the system. The receiving user is greeted with a notification of an incoming message and the option to respond, ignore, or report as spam.

ii) To test the messaging system, I would say that the first thing that we would want to test would be to make sure that the correct users are getting the messages that are being sent. This would be accomplished by checking the receiving user against the intended receiving user. Next, I would want to test the notification system within the messaging. I believe this would be as simple as making sure (isTrue) there was a notification message that appeared when the message came in. Last, I would say that we would want to test deletion or reply of some messages. I do not know how to make sure that the messages are being replied, however, the deletion would be handled by checking to see if the message in question is still found within the messaging system.