Simeon Patton

CS362 Oregon State Spring 2021

Homework 4

Question 1;

1. Submitted files;

cube\_volume.py

test\_cube\_volume.py

can be found at;

<https://github.com/pattons-OSU/CS362_Homework/tree/master/Week_5>

1. 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;

1. Submitted files;

average\_list.py

test\_average\_list.py

can be found at;

<https://github.com/pattons-OSU/CS362_Homework/tree/master/Week_5>

1. 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;

1. Submitted files;

name\_gen.py

test\_name\_gen.py

can be found at;

<https://github.com/pattons-OSU/CS362_Homework/tree/master/Week_5>

1. 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.