Justin Phillips

cs362 winter 2020

2/5/2020

Random Testing Assignment

I developed the random tester file that has two different functions the first function is used to test if the random generator created the appropriate random letter or punctuation. The first step was to call the inputChar function from the testme.py and set that to the variable tc. Now that I had the input, I need to test it. To test it I created an if statement to check if the input created is a letter by using the python function isalpha() and the result of this function call would be compared to True. If this was correct it would then verify it by an assertTrue call for tc.isalpha() == True. If the 1st argument wasn't equal it jumped to the else statement to check if the input was a punctuation. I created a temp variable and set it equal to " ", then ran a for loop checking that temp is in string.punctuation and if it was it got set to temp then run a assertEqual comparing that tc and temp were equal.

The other function I created was to test the inputstring function in the test me file. The inputstring function created a random string with possibility of having a space in the string. The first thing I did was call the inputstring function and set the results to variable called ts. I also create a variable called count which was just used to hold the number of times the loop iterated. I then ran a for loop through the string and compared each item in the string to isalpha if that was true or if the string item was equal to '\0' it would increment count and print the item int the string. If both failed it jumps to an else where it prints invalid input and breaks the loop. If we successfully run through the string without the error message, I ran an assertEqual comparing the count of the letters in the string to the proper length the string is supposed to be.

I had to constrain my randomizer on the inputString function, at first, I didn't have the '\0' as a choice for the string, therefore I couldn't get past state 8. After figuring that out, I added a constraint of the number of letters that were available to be selected from. I at first had string.ascii_letters and the length of the string at 12 but the time it took to get "reset" to appear via the randomizer was too long. I adjusted this by shorting the number of letters to choose from down to 15 and the length of the string down to 6. Doing this allowed me to lower the time, to a time that was more manageable to work with.

After running the coverage tests on both functions, I believe that it's at 100% coverage.