This assignment asked us to create the functions inputChar() and inputString() that randomly generate appropriate test cases with the hopes of triggering the error message that breaks the provided while loop.

With the criteria provided, we can create a list of all possible relevant inputs to use in each function. For the inputChar function, I used the characters [,(,{, ,a,x,},) and ], as those are the possible inputs that would cause the state value to increment. For example, when state == 0, the program will constantly be checking to see if the value of c is '['. As such, any other value is invalid. If we use a set of all values that the if statements will be looking for at one point, represented in a char array, we can functionally simulate all inputs while shortening the test time required by not checking all irrelevant ASCII characters (EX: the character '6' will never cause state to increment, so we can treat it as invalid and discard it. Any other value in the set that isn't the current correct value will serve the same purpose.)

Similarly, once state is equal to 9, the program will begin checking the randomly generated string to see if it equals null-terminated "reset". We can create a character set, again represented by a char array, of all possible valid characters, in this case "r","e","s" and "t". We can then assign each value of the string one of the values at random each time the loop iterates until we hit "reset". We can safely omit all other ASCII characters using the same logic that we used in the inputChar() function, in that the characters in the set can serve as both valid and invalid, whereas all other characters only serve as invalid. In the interest of saving execution time, we can omit the set of all other invalid characters, as the "rest" characters serve their functional purpose.