Program Analysis Verification & Testing Assignment 1

Name:- Vaibhav Rahulkumar Sheth

Roll No.:- 231110054

Department:- MTech CSE

Implementation [fuzzSubmission.py]:-

1. **compareCoverage function() :-**

   It checks if there are no sublists in total_metric that have the same length as curr_metric and are also equal to curr_metric. If such sublists exist, it returns False. Otherwise, it returns True, indicating that there are no matching sublists.

2. **updateCoverage function()** :-

   function computes the total coverage metric (total_metric) by comparing it with the current metric (curr_metric). If there is new coverage seen in curr_metric, it appends curr_metric to total_metric and returns the updated total_metric. Otherwise, it returns the original total_metric.

3. **mutate function() :-**

   The mutate function takes input data in the form of a dictionary (input_data.data), mutates the values of the dictionary's key-value pairs, and returns the updated input_data. The mutations involve choosing random value from dictionary values list by using random.choice method and then multiplying each value by a random integer between 1 and 5 using random.randint method adding 1 to the result , and performing a bitwise AND operation with 63. The function ensures that coverageInfo is not mutated and that irList remains unmodified.

   Adding 1 to make sure that if all inputs are 0's (Zeros) then the program does not fail.

   Here, ANDing with 63 is used to generate different values just by preserving all 1's of input data.

Test Cases:-

Here I'm taking 5 different test cases to check verification of program

Assumptions :-

1. **Input Data Format**: The function assumes that the input data is provided as a dictionary with specific key-value pairs where keys represent variables and values are integers.
2. **Mutation Process**: The mutations applied to the input data involve random integer multiplication, addition, and bitwise AND operations. It assumes that these operations are suitable for the specific use case or testing scenario.
3. **Randomness:** The function relies on random values for mutations, assuming that the randomness is suitable for generating diverse test cases.

Limitations :-

1. Because of Random function, it might be possible that it does not cover all cases. Random function is choosing any value between 1 to 5 only.
2. The function uses common mutation operations, including multiplication, addition, and bitwise AND. It does not uses any complex mutation strategies that might be needed for specific testing objectives.
3. It might be possible that mutation function does not cover all coverage for your test program.