Assessment of test case coverage for `format()`

A machine test that automatically assesses the completeness of a list of provided test cases for the `format()` function.

The `format()` function takes as its input an integer between 0-5999 (inclusive), which represents a time in tenths of a second. For instance, 423 is 42.3 seconds. It returns a string of the form "A:BC.D", where A, B, C, and D are each digits. A represents minutes, B and C represent seconds, and D represents tenths of a second. Thus, A, C, and D will be integers between 0-9 (inclusive), while B will be an integer between 0-5 (inclusive). Returning to the example of 423, the expected output would be "0:42.3".

Your input file should contain a single Python definition:

- A list `TEST_CASES` containing at most **12 test cases** for the function `format()`.
- Each test case in this list should be an integer in the range 0 to 5999 (inclusive).

These integers will be used as input to `format()`. You do not need to provide the corresponding output. We will compute it from our reference implementation. Note that submissions to CanvasTest that do not conform to this format are rejected.

CanvasTest will assess the completeness of the test cases in `TEST_CASES` using a hidden suite of incorrect implementations of `format()` that we have compiled. Specifically, CanvasTest will run each of these incorrect implementations on the test cases in `TEST_CASES`. If an incorrect implementation returns a correct answer on all tests in `TEST_CASES`, CanvasTest will record that the program (incorrectly) passed the provided tests. If an incorrect implementation returns an incorrect answer on at least one of the tests in `TEST_CASES`, CanvasTest will record that the implementation (correctly) failed the provided tests.

The grade on Canvas that you receive for this exercise will depend on the number of incorrect implementations that failed the provided tests. Therefore, to maximize your score on this exercise, your goal should be to construct a list of tests that maximize the number of programs that fail your provided tests. In constructing this list, you should try to target each distinct logical category of input. In particular, try to identify "edge cases" involving unusual or extreme values for one or more inputs.

You may submit to CanvasTest as many times as you would like. Your last submission before the deadline will determine your grade on the exercise. You will need to come back to this page and relaunch CanvasTest to resubmit.

This tool was successfully loaded in a new browser window. Reload the page to access the tool again.