Assessment of test case coverage for `blackjack3()`

A machine test that automatically assesses the completeness of a list of provided test cases for the `blackjack3()` function.

The `blackjack3()` function takes as its input three cards, each of which is represented as a single character from the string "23456789TJQKA". For instance, one valid tuple of inputs might be ("A", "K", "5"). In blackjack, number cards (including "T", which represents 10) are worth their value, and face cards ("JQK") are worth 10. Aces can be worth either 1 or 11; the choice between these is made such that the value of the hand is as high as possible without going over 21. (Note that there are some cases in which it is impossible to stay under 21.) Returning to the example of ("A", "K", "5"), the expected output would be 16.

Your input file should contain a single Python definition:

- A list `TEST_CASES` containing at most **12 test cases** for the function `blackjack3()`.
- Each test case in this list should be a list of tuples of length three whose entries are characters in "23456789TJQKA".

These tuples will be used as the input to `blackjack3()`. You do not need to provide the corresponding (expected) 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 `blackjack3()` 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 needs to be loaded in a new browser window

The session for this tool has expired. Please reload the page to access the tool again