HyperionDev

Programming With Built-In Functions

Model Answer Approach

Visit our website

Auto-graded task 1

This program prompts the user to input 10 floating-point numbers and performs statistical operations on these values. Using the `statistics` module, the program begins by asking the user to input each number, storing them in a list called `numbers`. It then calculates the sum of all values in the list, identifies the indexes of both the maximum and minimum values, and computes the mean, rounded to two decimal places, with `statistics.mean()`. Additionally, it finds the median of the list using `statistics.median()`.

The program handles numerical data effectively, providing a clear view of data properties. However, if the user enters a non-numeric input, a `ValueError` will occur, as the program attempts to convert each input to a float without validation. Adding input validation or error handling could enhance user experience.

Auto-graded task 2

This program is a joke generator designed to display a random joke each time it runs. It contains a list of jokes stored in the `jokes` variable, each with a punchline. Using Python's `random` module, specifically the `random.choice()` function, the program selects one joke at random and prints it to the console. This approach ensures a unique joke with each execution.

One potential issue is if the joke list is empty; `random.choice()` will raise an error. Also, using only a small set of jokes can make the output repetitive if frequently run.