HyperionDev

Defensive Programming – Exception Handling

Model Answer Approach

Visit our website

Auto-graded task 1

The approach to This Python program is a simple calculator application designed to perform basic arithmetic operations, store the results, and retrieve past equations. The program consists of three main functions: `perform_calculation()`, which handles the arithmetic operations and stores results in a file; `print_previous_equations()`, which reads from the file and displays past calculations; and `calc_app()`, which serves as the main interface for user interaction, presenting options for performing calculations, viewing previous equations, or exiting the program.

The `perform_calculation()` function prompts the user to input two numbers and an operator. It validates the operator to ensure it is one of the four basic arithmetic types (+, -, *, /). If a division by zero is attempted, the program raises a `ZeroDivisionError`, and if an invalid operator is entered, a `ValueError` is raised. All errors are caught, and informative messages are displayed to the user. Successful calculations are written to `equations.txt` using `a+` mode, which allows reading from and appending to the file and ensures that a new file will be created if it doesn't already exist.

The `print_previous_equations()` function reads and displays past calculations from `equations.txt`, handling the case where the file may not exist or is empty with an exception handling block that prevents errors by prompting users to select another option. This function ensures that the user can review previous calculations without encountering errors if no records are found.

Finally, the `calc_app()` function provides the main menu where the user can choose to perform a calculation, view previous results, or exit the program. It loops until the user selects the exit option and ensures that only valid choices (1, 2, or 3) are accepted.

Possible pitfalls in this implementation include handling errors related to file access. For example, if there are permission issues with writing to or reading from `equations.txt`, these could result in unhandled exceptions. Another potential issue could arise if the user enters non-numeric values for the numbers, leading to a `ValueError`. This is properly managed within the program but could be frustrating if the user consistently inputs invalid data. Lastly, if the file grows very large over time, the program does not account for performance or memory usage, which could eventually slow down the retrieval process. These issues could be addressed by implementing better file management or validation checks.