TASK- 1 SIMPLE CALCULATOR

This Python code creates an advanced calculator with a graphical user interface (GUI) using Tkinter. It supports basic arithmetic operations, square root, exponentiation, percentage calculation, and includes a calculation history feature. Let's break down the code in detail:

1. Imports:
   - `tkinter as tk`: Imports the Tkinter library for GUI creation.
   - `from tkinter import messagebox`: Imports the `messagebox` module for displaying pop-up messages (errors, confirmations).
   - `import math`: Imports the `math` module for mathematical functions like square root.
2. `on_click(button_text)` Function:
   - This function handles button clicks.
   - `=`: Evaluates the expression in the entry field using `eval()`. It's crucial to understand that `eval()` can be dangerous if you're taking user input directly, as it can execute arbitrary code. For a real-world application, you would want to use a safer expression parser. The result is appended to the `history` list and saved to the "history.txt" file. Error handling is included using a `try-except` block to catch invalid expressions.
   - `C`: Clears the entry field.
   - ⌫: Deletes the last character from the entry field.
   - `√`: Calculates the square root of the value in the entry field using `math.sqrt()`. Error handling is included for invalid input.
   - `^`: Inserts the exponentiation operator `**` into the entry field.
   - `%`: Inserts `/100` into the entry field for percentage calculation.
   - `History`: Calls the `show_history()` function to display the calculation history.
   - Other buttons: Inserts the button text (numbers, operators) into the entry field.
3. `show_history()` Function:
   - Creates a new top-level window (`history_window`) to display the calculation history.
   - Creates a listbox (`history_listbox`) to display the history items.
   - Iterates through the `history` list and inserts each item into the listbox.

- Adds a "Clear History" button that calls the `clear_history()` function.

4. `clear_history()` Function:
   - Clears the `history` list.
   - Calls `save_history()` to save the empty history to the file.
   - Displays a message box confirming that the history has been cleared.
5. `save_history()` Function:
   - Saves the `history` list to the "history.txt" file, with each item on a new line.
6. `load_history()` Function:
   - Loads the calculation history from the "history.txt" file.
   - Returns an empty list if the file is not found.
7. `on_key(event)` Function:
   - This function handles keyboard input.
   - If the pressed key is a number, operator, or other valid calculator character, it inserts the character into the entry field.
   - If the Enter key is pressed, it calls the `on_click('=')` function to evaluate the expression.
   - If the Backspace key is pressed, it calls the `on_click('⌫')` function to delete the last character.
8. GUI Setup:
   - `root = tk.Tk()`: Creates the main window.
   - Sets the window title and background color.
   - Binds the `<Key>` event to the `on_key()` function to handle keyboard input.
   - `entry = tk.Entry(...)`: Creates the entry field for displaying and entering calculations. It's configured to be right-aligned.
   - `buttons`: A list of tuples representing the calculator buttons. This makes it easy to create the button grid.
   - The code then creates the buttons using a nested loop and the `grid()` layout manager. The `command` of each button is set to a `lambda` function that calls `on_click()` with the button text.
9. History Initialization:

- `history = load_history()`: Loads the calculation history from the file when the calculator starts.

10. Main Event Loop:
    - `root.mainloop()`: Starts the Tkinter event loop, which makes the GUI interactive.

Key Features and Improvements:

- Calculation History: Stores and displays previous calculations.
- Keyboard Input: Allows users to use the keyboard for calculations.
- Error Handling: Includes basic error handling for invalid expressions and square root input.
- Clear History: Provides a button to clear the calculation history.
- File Saving/Loading: Saves and loads the history to/from a file.
- Advanced Operations: Includes square root, exponentiation, and percentage calculations.

Security Note: As mentioned earlier, using `eval()` can be dangerous. For a production calculator, you should use a proper expression parser to prevent security vulnerabilities. There are libraries available that can safely evaluate mathematical expressions.