Katharine Brumback Final Project:

Health Assistant

# Project Overview:

This program is a health assistant that includes a BMI calculator and a text editor. The BMI calculator allows the user to enter their height and weight and calculates their BMI. It also keeps track of the user's average weight. The text editor allows the user to open and save text files. The user can switch between the two programs using the menu bar. Please note that changes to the project are potential while building the application based on new features.

# Objectives:

* Develop a working Python tkinter GUI application that includes a BMI calculator and a text editor.
* Practice using libraries such as tkinter, ttk, messagebox, and filedialog.
* Learn how to implement a modular approach in programming.
* Improve programming skills in implementing secure coding best practices and input validation.
* Enhance testing skills and user manual creation.

# Requirements/Task(s):

Task 1: Create a working GUI tkinter application with at least two windows.

Task 2: Implement a modular approach in the application.

Task 3: Ensure consistent clear navigation through the GUI application.

Task 4: Use at least two images in the application.

Task 5: Include at least three labels.

Task 6: Include at least three buttons.

Task 7: Include at least three callback functions with each button, including exit button.

Task 8: Implement source coding best practices, including input validation to check if the user entered the correct data type, make sure the entry box is not empty.

Task 9: Develop an appropriate set of test data to fully validate the program against.

Task 10: Write a user manual for the final project and submit it according to the instructions.

Task 11: Fully document (comment) the corrected Python tkinter source code with appropriate comments.

Task 12: Submit the GitHub repository link for the final project.

# Notes/Research:

The Health Assistant program utilizes libraries such as tkinter, ttk, message box, and file dialog to create a GUI application. The BMI calculator frame includes entries for the user to enter their height and weight. It also includes a button to calculate the BMI. The BMI is calculated using the formula: weight (kg) / height (m)^2. The BMI is displayed along with the classification of the BMI. The classification is determined by the following table: Underweight = <18.5, Normal = 18.5-24.9, Overweight = 25-29.9, Obese = 30+. The text editor frame includes a menu bar that allows the user to create a new file, open an existing file, save the current file, and exit the program.

Daily Issues: Syntax Errors, Debugging, Integration, and Design

Fixes: Code review by outside sources, conducted user testing with friends, sed debugging tool of VS Code program.

Final Project UPDATE II : April 27th, 2023 – So far, I have completed a significant portion of the project and met the necessary criteria for completion. During the project's development, I faced multiple problems with adding images, converting meters to feet, Gui design, and the opening/saving of files. Through progress I was able to overcome the hurdles through research and constant testing. However, I haven't had the opportunity to test the program outside of the VS Code Interactive Window. Moving forward, my next step is to correct the failed test parameters before peer review submission.

# Outline

1. Create the BMI calculator frame.
2. Create entries for user input.
3. Create a button to calculate the BMI.
4. Calculate the BMI using the formula and display the result.
5. Determine and display the BMI classification.
6. Keep track of user's average weight and display it.
7. Create the text editor frame.
8. Create a menu bar with options to create a new file, open an existing file, save the current file, and exit the program.
9. Create a text widget for the user to enter and edit text.
10. Implement navigation between frames using a notebook widget.
11. Allow the user to switch between the BMI calculator and text editor frames.
12. Implement input validation for user entries.
13. Ensure that the user enters valid input for height and weight.
14. Ensure that the entry boxes are not empty.
15. Test the program with appropriate test data.
16. Ensure that the program functions as intended.
17. Fix any errors or bugs that arise during testing.

# Test Parameters:

1. Test that the BMI calculator frame is created and displayed properly.
2. Test that the user can enter valid input for height and weight.
3. Test that the program calculates the BMI correctly based on the user input.
4. Test that the BMI classification is determined correctly based on the calculated BMI.
5. Test that the program can keep track of the user's average weight.
6. Test that the text editor frame is created and displayed properly.
7. Test that the menu bar with the appropriate options is displayed and functioning correctly.
8. Test that the user can create a new file, open an existing file, and save the current file as intended.
9. Test that the text widget is displayed, and the user is able to enter and edit text as intended.
10. Test that the navigation between frames using the notebook widget is functioning correctly.
11. Test that the user can switch between the BMI calculator and text editor frames as intended.
12. Test that the input validation is functioning correctly, and the program displays error messages for invalid input.
13. Test that the program ensures that the entry boxes are not empty before calculating the BMI.
14. Test the program with different types of input data, including extreme values, to ensure that it functions as intended in all cases.
15. Ensure that the program functions as intended in all scenarios.
16. If any errors or bugs arise during testing, investigate, and fix them before releasing the program.
17. Finally, test the entire program end-to-end to ensure that all functionalities work seamlessly together.

# Failed Test Parameters:

1. Edge Cases:
   1. Minimum height:
      1. Test: Height = 1 foot, Weight = 50 pounds
      2. Result: Error message displayed: "Height is too low."
      3. Conclusion: The test did not pass; the application did not handle the edge case appropriately.
2. Minimum weight:
   1. Test: Height = 5 feet 6 inches, Weight = 1 pound
   2. Result: Error message displayed: "Weight is too low."
   3. Conclusion: The test failed; the application did not handle the edge case appropriately.
3. Negative values:
   1. Test: Negative height and weight values
   2. Result: Error message displayed: "Invalid input. Height and weight must be positive values."
   3. Conclusion: The test failed; the application did not handle negative values correctly.

# Links:

[GitHub Link](https://github.com/MissLadyPandas/Health-Assistant)

# Test Parameter Snips:

1. Review “Snips of Tests” Folder