Documentation

Motivation

The idea for this project came to us because we wanted to code something with an interface and a program that might actually be of use to us. Since both of use are practicing ten finger typing we were already familiar with the idea of typing speed test. We then wanted to create a typing speed test that was designed like a game. To achieve this, we opted for an arcade look and implemented the competitive element of high scores, which allows players to immortalize themselves on the high score list. We chose tkinter over pygame to create the interface since we already worked with pygame and wanted to try out the tkinter library.

Programming journey

One of the biggest problem for us was the spell checking and the spelling error counting. Our first attempts focused on comparing every written letter with the letter on the index of the displayed word. After the first minor successions with this attempt we quickly noticed that this was very susceptible to errors. Problems occurred with the delete/back key which resulted in a malfunction of the error counting as well as the shift key. Further we tried to start the spell test with a condition that stated to start the program whenever a key is pressed expect from certain keys such as shift or control. This attempt also failed since the somehow we could not figure out the correct virtual key codes (we tried all of them). Further we noticed that they might differ from the key codes of other operating systems.
We then took the easier more efficient way: Instead of comparing each letter on the same index we continuously compared the whole sentence written by the user with the displayed sentence with the *startswith()* function. This made the error checking function much easier. The error counting still posed a problem since the program counted an error every time the written sentence was different from the displayed sentence. Therefore when you typed something wrong and haven't corrected it directly but kept typing, it did not count as one error but as multiple errors. After considering multiple variants we solved it with an easy boolean variable is_wrong that checks which is set to **True** every time you type something wrong. Therefore it does not count consequential errors but only the first typing error.
We also had to fix some bugs that occurred with the reset button (score of 600 wpm if you reset in the middle of writing a sentence) and termination after a finished sentence. We fixed this by checking if the sentence is complete before storing the current score (wich happens right after the reset). Further when including a Highscore Table, we first implemented it using a simple text file, but later decided on storing more data, including separate data for each individual user, in a .csv file using pandas.
Another challenge we faced was to display the stored data in a tkinter window, for which we used the Matplotlib library to plot the user's scores. The tkinter library does not include a function to display pandas data frames, so instead of importing a whole other library to take care of that, we built our own function that loops through and transforms each entry in our user data into a tkinter Label.