HSG Speed Typing Test

Riya Masand 20-617-965
Pascal Hasler 20-612-909
Nassim Infantes 18-612-226
Philip Westwood 15-609-969

Assignment Overview (What is the program doing?)

The goal of this project is to run a Python program which allows a user to see his typing skills. This includes the time needed for selected sentences, the accuracy as well as the amount of words per minute.

This project is based on the code found in the following link: https://tech-vidvan.com/tutorials/project-in-python-typing-speed-test/

Instructions (How can one run it?)

- Please Download PyCharm and create a new project
- Download our submitted file “Speed-typing files.zip” from Github in order to have access to pictures and text used in the program
- Extract all the files and save them in the respective PyCharm project
- Copy-paste the code from Github into PyCharm
- Please ensure that the package Pygame is installed: Use *pip install pygame* to do so
- Run the code
- Follow the game instructions to play
- After getting the results, you can try again by clicking on the “Keep calm...” icon

Assignment Specifications (How does the program work?)

First of all, a short explanation to the different files (images and text file), which are part of the program:

- Final.jpeg – A background image we will use in our program.
- First screen.jpg - The image to display when starting game.
- keep-calm-and-try-again.png – An icon image that we will use as a reset button.
- sentences.txt – This text file will contain a list of sentences separated by a new line.

Imports

After installing the pygame library, it is important to import it along with built-in python functions which will be useful later.

The next step is to define the variables of our code in the class game (i.e. a code template to create new objects).

def __init__()

To initialize the objects, we call the function init. In this constructor, we initialize window settings, variables for calculation and load the images for the display.

def draw_text()

To format the text (font, text, size, color), we use the draw_text method. Afterwards, we need to present sentences so that the user can exercise his speed typing skills.

def get_sentence()

To open the *sentences.txt* file, we define get_sentence where it is important to split the whole string with a newline character and return a random choice of sentence for the user.

def show_results()

Now, the results of the user must be calculated. Three results are important:

1. To calculate the total time, we substract the time at which the first character was written from the overall time needed using the time.time() function.
2. To calculate the accuracy, we create a percentage using the formula (correct charactersx100)/ total characters in sentence). Correct characters exist if the self-input text is exactly as the character enumerated from the object. However, there are limitations in this formula. As soon as a false character is inserted or a character is missing all the consequential characters will be seen as mistakes.
3. To calculate the words per minute, we convert the total time in minutes by dividing by 60 and multiply it with the amounts of words written.

We then define self.results summarizing every result which will be printed later on. The results shown have to be transformed from an integer to a string. Further, they have been rounded.

To give the option for the user to restart the test, we draw an icon image on the screen with the reset option. Additionally, it has been scaled and positioned to fit on the display.

def run()

This is the primary method that that will handle all the events. At the beginning, all the variables will be reset. We have created a while loop (infinite) which captures all mouse and keyboard activities of the user. In this while loop there is another for loop which makes a check on the mouse and keyboard activities. If the user quits the game, the program will stop running. If the user clicks on the box, the time will start to run and the user can write the input text. Further, if the user clicks on the reset box, the game restarts. If the user presses any key then the program will update the message on our input box. The enter key will end typing and we will calculate the scores and display it. Another event of a backspace is used to trim the input text by removing the last character.

def reset_game()

This defines how the game should be reset, including all kind of variables.