Typing Speed Test – Rapport Final

1. Introduction

Ici, mon objectif est de créer une application très simple en Python. Il s'agit du "Typing Speed Test", qui mesure la vitesse d'écriture de l'utilisateur. Après l'exécution du programme, il affiche une minute de temps réservée à la préparation. Après la fin de cette période, l'utilisateur peut écrire un texte. Pendant le temps déterminé, le programme doit calculer le nombre de secondes écoulées et le nombre de lettres écrites. Sur la base de ces calculs, le programme doit calculer automatiquement le WPM, qui signifie "mots par minute". À la fin, le programme doit afficher si la vitesse de l'utilisateur est supérieure à la moyenne ou non.

2. Objectif du Projet

L'objectif principal est de créer une application simple et fonctionnelle qui :
1.Donne un temps de préparation avant de commencer à taper.
2.Permet à l'utilisateur d'écrire librement pendant une durée déterminée.
3.Mesure automatiquement :
4.Le temps écoulé,
5.Le nombre total de caractères écrits,
6.La vitesse de frappe (WPM).
7.Affiche enfin une évaluation claire : performance faible, moyenne ou élevée.

3. Description Générale du Fonctionnement

Après l'exécution du programme, une minute de préparation est affichée afin de permettre à l'utilisateur de se mettre en place. Une fois ce compte à rebours terminé, l'utilisateur peut taper du texte. Le programme enregistre alors :
Le moment où l'écriture commence,
Le moment où elle se termine.
Le nombre total de caractères est ensuite converti en mots, en utilisant la règle standard : 1 mot = 5 caractères en moyenne. Grâce à ces données, le programme calcule automatiquement le WPM. Enfin, une comparaison simple permet de savoir si la vitesse est supérieure, égale ou inférieure à la moyenne.

4. Problèmes Rencontrés

Pendant le développement du programme, plusieurs difficultés sont apparues :

4.1 Gestion du temps

Au début, le calcul du temps écoulé donnait toujours zéro, car la variable de fin n'était pas correctement mise à jour.

4.2 Entrée utilisateur illimitée

L'arrêt de la saisie (input) causait parfois un blocage, puisque l'utilisateur devait appuyer sur Entrée sans écrire pour quitter la boucle.

4.3 Erreurs d'indentation

Python dépend fortement de l'indentation : un espace en plus provoquait des messages d'erreur.

4.4 WPM toujours égal à zéro

Le programme affichait parfois 0 WPM parce que la durée du test était trop courte ou mal enregistrée.

4.5 Organisation du code

Certaines lignes étaient répétitives et rendaient la lecture compliquée.

5. Solutions Apportées

5.1 Correction de la mesure du temps

En plaçant correctement start = time() et end = time() à l'intérieur du bon bloc, la différence devenait correcte.

5.2 Amélioration de l'arrêt de saisie

J'ai utilisé une condition simple : si l'entrée est vide, la boucle s'arrête proprement.

5.3 Revoir l'indentation complète

Chaque bloc a été corrigé pour respecter les règles de Python, ce qui a éliminé les erreurs.

5.4 Vérification de la durée

J'ai ajouté une protection pour éviter la division par zéro et garantir que le WPM soit calculé correctement.

5.5 Réorganisation générale

Le code a été simplifié pour être plus propre, plus lisible et plus facile à modifier plus tard.

6. Algorithme Utilisé

L'algorithme du programme peut être décrit ainsi :

1. Afficher un compte à rebours pour préparer l'utilisateur.
2. Démarrer le chronomètre.
3. Laisser l'utilisateur taper autant de lignes qu'il veut.
4. Si l'utilisateur appuie sur Entrée sans rien écrire → arrêter le test.
5. Calculer le temps total écoulé.
6. Compter tous les caractères tapés.
7. Convertir ce nombre en mots (mot = 5 caractères).
8. Calculer :
WPM = \frac{\text{nombre de mots}}{\text{minutes écoulées}}
10. Afficher le résultat final.
C'est un algorithme simple mais suffisant pour une première application de test de vitesse.

7. Conclusion

Ce projet m’a permis de comprendre plusieurs concepts essentiels en Python : la gestion du temps, la manipulation d’entrées utilisateur, le fonctionnement des boucles et les calculs basiques. Le programme obtenu fonctionne correctement, donne une estimation réaliste de la vitesse de frappe et offre un retour clair à l’utilisateur. Ce travail constitue une bonne base pour des améliorations futures, comme l’ajout d’une interface graphique ou d’un système d’historique.