Rapport final projet Boogle

Valentin GROS

Paul GOURET

Libraires à installer

Extension NuGet dans visual studio, si jamais elle n'est pas installée

Libraire SkiaCrap à installer dans NuGet.

Tri des listes

Pour le test tris nous avons d'abord procédé naïvement à un tri sélection (tri des liste en valeur du des dictionnaires motsParLongueur et motsParLettre, attributs de la classe Dictionnaire).

Cependant en testant sur la liste de mots complète, le temps de la fonction de tri était beaucoup trop long, trop long même pour avoir eu une idée précise (au moins 5 minutes)

Cela est normal car la complexité du tri séléction est dans le pire cas quadratique et on à affaire à une liste de string très longue, près de 130 000 éléments donc forcément, le temps d'execution avec un tri séléction est beaucoup plus fort.

Nous avons donc choisi un tri beaucoup plus rapide que nous avons pu voir en cours de Complexité et d'Algorithmie : le tri Fusion. La différence de temps d'execution est nette (25 ms) avec

le tri fusion, ce qui est normal car le tri fusion à un cout dans le pire cas quasi-linéaire (O(nlog(n))) ce qui est nettement moins couteux que le tri selection qui a une complexité $O(n^2)$

Recherche d'un mot dans le dictionnaire

Il fallait coder une méthode, permettant de vérifier que le mot rentré par l'utilisateur appartient au dictionnaire. Nous avons directement pensé à la recherche dichotomique en utilisant la méthode diviser pour régner, vu en cours de Complexité, qui consiste à diviser le tableau en sous tableau. O(log(n)) dans le pire cas. Donc optimal par rapport à une méthode itérative ou on parcourt un à le tableau tableau (plus couteux car complexité linéaire dans le pire cas)

Nuage de mots

Nous avons d'abord demandé à chat GPT de générer un nuage de points. La solution proposée la classe Bitmap de System.Drawing), cependant, et ce fut le cas pour beaucoup d'étudiants,

windows ne semble par prendre en charge cette bibliothèque. Il a fallu donc trouver une autre classe.

Nous avons donc finalement, après quelques recherches, décidé d'utiliser la librairie SkiaCrap. Il a fallu juste adapter le code pour l'appliquer sur la liste de mots trouvés par les deux joueurs. Puis finalement la fonction a marché du premier coup.

La dernière adaptation à utiliser était de pouvoir afficher automatiquement l'image du nuage de mots car jusque là le code génerait une image et l'enregistrait dans le dossier du projet, sans l'ouvrir. Il fallait manuellement donc ouvrir l'image. Chat Gpt a alors rajouté deux lignes de code en utilisant la class Process de System.Diagnostics.