Sean Grogan
20001636
TP1
IFT 2015

(1)

Les programmes, les méthodes, et les classes exécuter sans un message d'erreur (dans le Unit Testing et l'exécution du programme)

(2)

Le test unitaire du programme s'exécute sans erreur

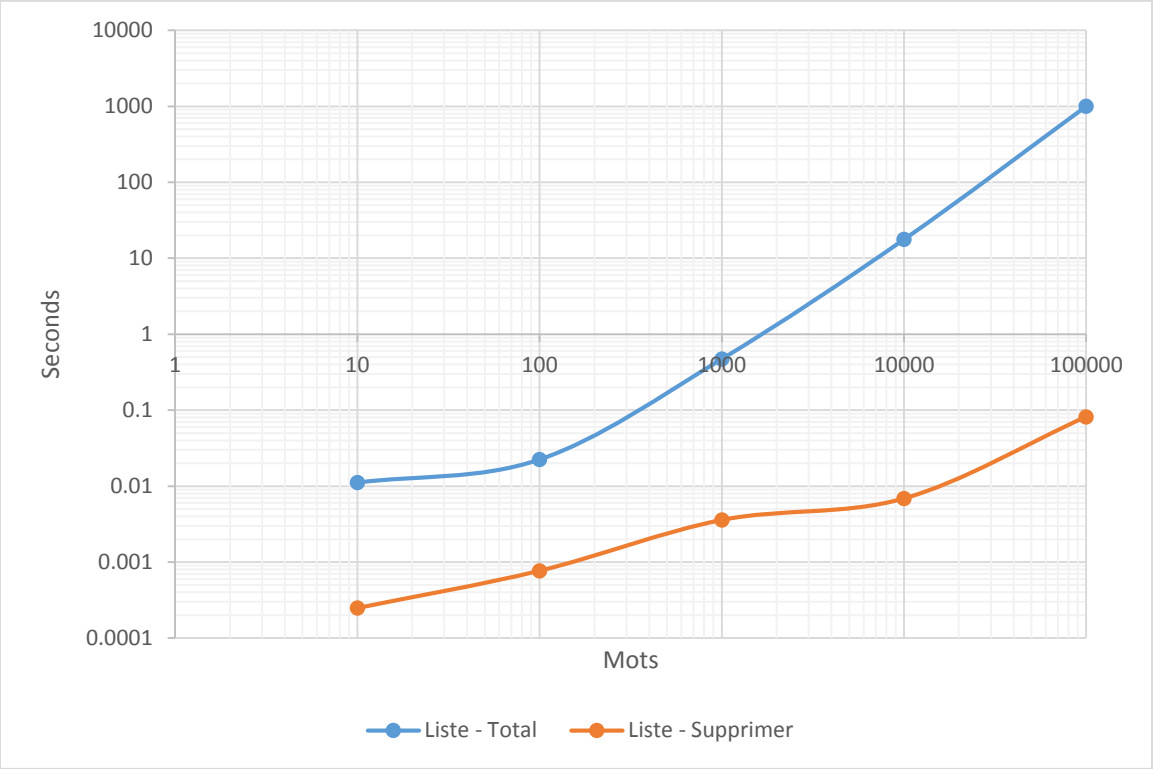
(3)

Run Time tableau (Seconds):

Items	Liste - Total	Liste - Supprimer	ListeTriee - Total	ListeTriee - Supprimer	Dict - Total	Dict - Supprimer
10	0.011	0.0002	0.012	0.0002	0.012	0.0001
100	0.022	0.001	0.024	0.0003	0.016	0.0001
1000	0.472	0.004	0.374	0.0003	0.049	0.0001
10000	17.704	0.007	8.697407199	0.000239743	0.226070297	7.35E-05
100000	1001.072	0.082	989.239	0.0003	1.839	0.0001
1000000					18.529	9.28E-05

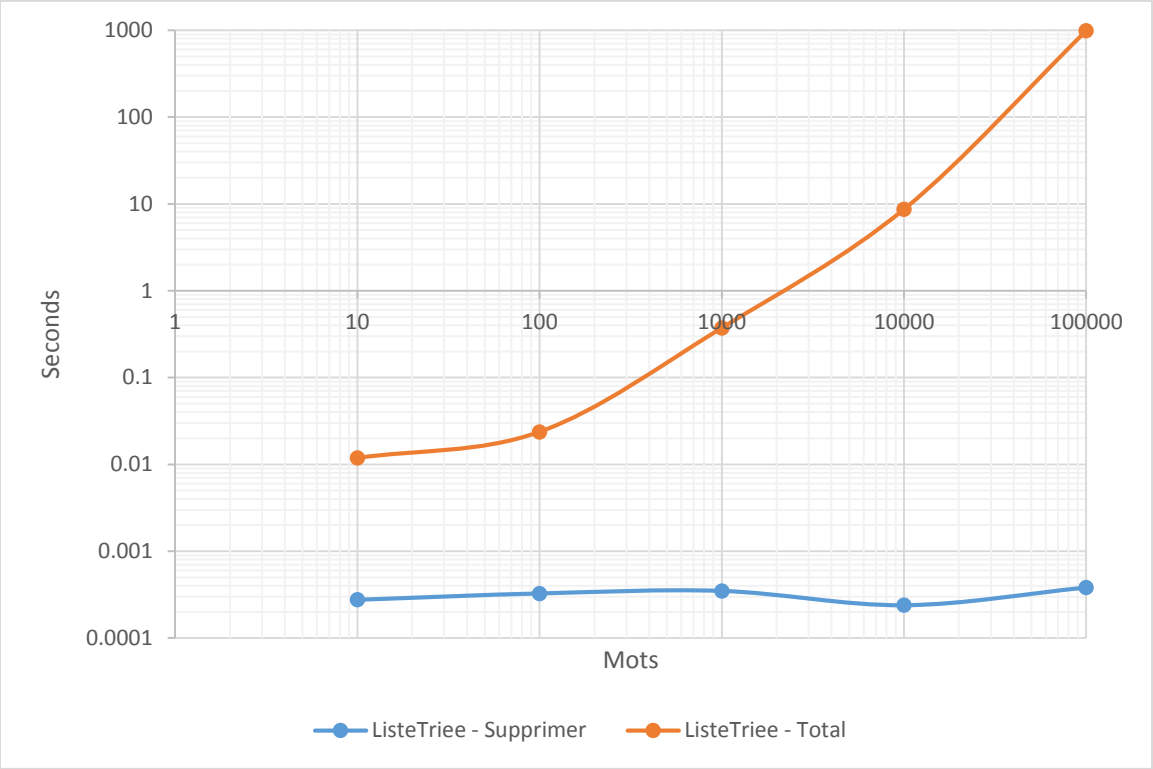
Pour le supprimer methode, J'ai supprimé les 5 mêmes mots aléatoires de chaque structure de données.

Liste :



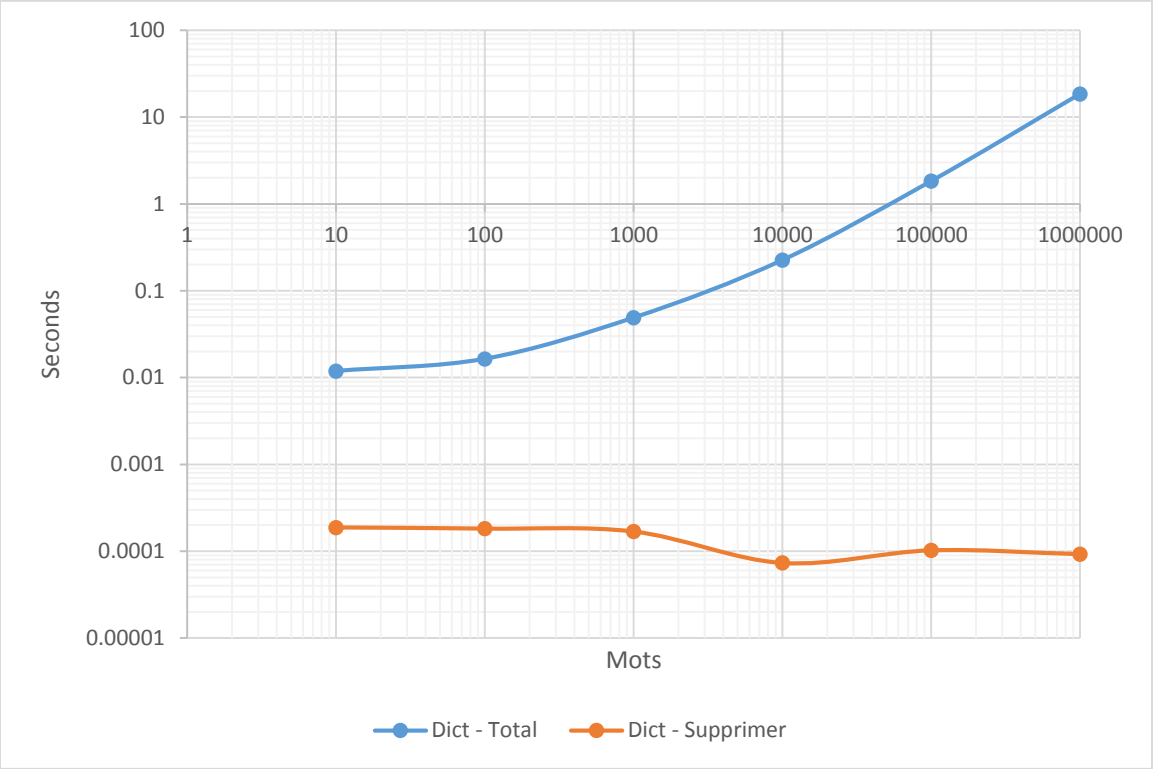
"Liste.py" a été la pire en termes de temps à résoudre. Il a également pris le plus de temps à supprimer les mots comme il a augmenté avec le temps.

Liste Triee :



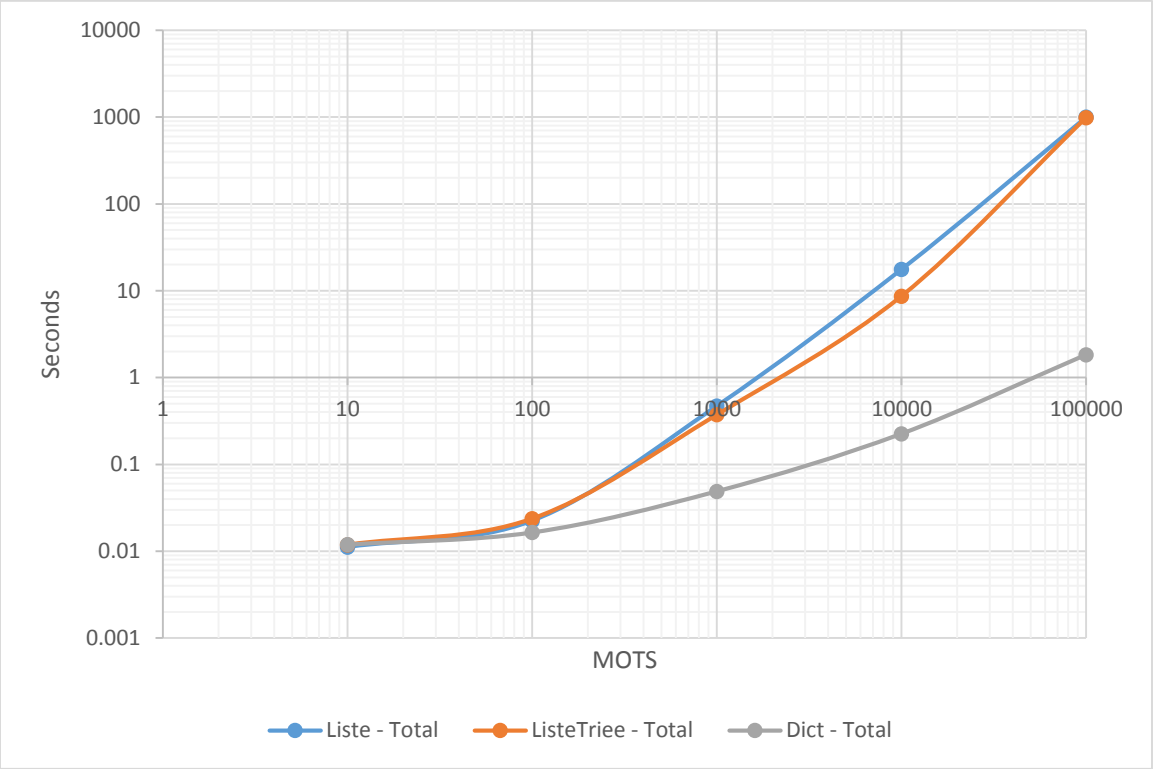
"ListeTriee.py" était presque aussi mauvais en termes de temps de résoudre l'ensemble du programme. Mais parce que le code pourrait utiliser "recherche binaire" dans mon code (à cause de cela étant triée), rechercher des mots à supprimer a pris beaucoup moins de temps.

Dict:



'"Dictionary. Py" était la meilleure structure de données pour le problème posé. Il est basé sur la classe construite en "Dict ()". Il était si rapide qu'il y avait une légère diminution du temps de recherche d'un mot que la liste de mot devenu plus grand.

Comparison des trois classes :



dans ce graphique, je vois que le haut de la classe est beaucoup plus rapide pour l'exécution de l'ensemble du programme de mes classes "self-made"