TEHARI Ayoub **07-02-2020**

L3-informatique_inf1

Compte Rendue 3 : utilisation de l'aléatoire

Introduction :

j'ai commencé par implémenter la méthode `newWorsSeq(String w1, String w2)`, qui vérifie avant l'ajout de tester si le premier mot est déjà dans la hashtable, si oui il teste encore pour le deuxième mot dans la table imbriqué, si oui, on incrémente le nombre d'occurence dans la table, sinon on met les deux mots avec un 1 qui represente une seul occurrence de la suite « mot1 mot2 »,

Deuxièmement, la méthode `Talk(int nbWord)`, permet de compter le nombre des occurrences cumulées et comparer à chaque fois qu'on a calculé ce dernier avec le nombre tiré au hasard qui est compris entre 0 et le nombre des mots en deuxième position, pour choisir un mot parmi ces mots.

Finalement, j'ai utilisé ce site : https://wordcounter.net/; qui m'as permit de compter et classer les mots par leurs nombre d'occurrences dans le texte.

Exercice 3. Validation du comportement du programme :

j'ai utilisé le fichier CRexemple.txt comme texte d'apprentissage,

	Texte d'apprentissage	Texte généré 100	Texte généré 10k mots	Texte généré 100kmots	Texte généré 1Mmots
Mot le plus fréquent	10% de	6% de	9% de	9% de	9% de
2 mot le plus fréquent	5% le	4% nous	5% le	6% le	6%le
3 mot le plus fréquent	4% la	3%l'algorithme	4% la	4%la	4%la
4 mot le plus fréquent	2% et	3%au	2% des	2%et	2%et
5 mot le plus fréquent	2% nous	3%chaine	2% un	2%des	2%des
6 mot le plus fréquent	2% que	2%compte	2% du	2%nous	2%un
7 mot le plus fréquent	2% pour	2%rendu	2% est	2%un	2%du
8 mot le plus	2% est	2%avons	2% et	2%du	2%nous

fréquent					
9 mot le plus fréquent	2% des	2%implémenté	2% en	2%est	2%est
10 mot le plus fréquent	2% un	2% naif	2% nous	2%l'algorithme	2%l'algorithme

Conclusion:

Pour conclure, on peut dire que le comportement du programme est valide dès que la taille choisit est relativement grande, puisqu'on remarque que pour toute les tailles choisit, on a sur le tableau ci-dessus le même mot pour la ligne « mot plus fréquent » : de, et que lorsque N dépasse 10K (N étant la taille du texte généré) les pourcentages des 1à6 top mot sont clairement identiques, sauf quelques nuances à propos de l'ordre des mots, et puis qu'on retrouve les même mots dans la plupart des cas de 7à10.

on voit aussi que mot 'de' qui tend vers un pourcentage de 10 % en incrémentant N.

On peut dire aussi que le choix du mot suivant à partir d'un mot donné est conforme aux probabilité tiré du texte d'apprentissage, qui est lui même conçue de la méthode nextInt(), et qui m'as permit d'avoir cette conformité puisque elle suit une lois uniforme.