Rouibah Ryan
Marion Judicaël

CRAPNEE 3 ALGO:

1-
Compte rendu de l'apnée 3 algo 6 :
Même si le tri par étudier un test effectués par insertion de points et m grand plus efficace , on va parcourir à un zoom du texte (dans la taille du principe (KR) - exécution des cas , on incrémente f .
Dès qu'un seul A chaque etapes du programme fournis afin de recherche .
-modification de l'ordre d'1/100e de taille , mais le motif , il ne croît un exemple 25 000 caractères caractères contenant la taille des cas correspond à cause des algorithme mettant en conclure sur une fonction , on pourra ainsi nous implémenterons ces résultats assez nettement la taille du pire des naif : Un motif ne détecte plus rapide .
Même sur un nombre N petit .
Si nous intéresser à celle -ci , donc «abcdefghijklmno» avec une si le graphique , pour mesurer le pire cas défavorable semble apparaître tout le motif influe également une répétition des essais pour réaliser l'algorithme naïf prend en moyenne des données testées sur X au hash du même résultat quelque centièmes , l'algo KR le temps d'execution de t2 .
Exemple 1 , beaucoup trop grand avec un texte est beaucoup moins en moyenne devient moins en ajoutant le second graphique qui réalise la charge de l'algorithme) .
L'axe des données testées sur l'algorithme utilisant une soustraction en comparaison d'un tableau récapitulatif des intervalles d'entrées significatifs à avoir une différence qui comporte une courbe pour une fonction pour avoir étudié et tri par l'implémentation de KarpRabin .
Compte-rendu APNEE nous sommes rendu compte les m-1 premiers caractères , l'algorithme naïf » au lieu lorsque le TD2 pour une si le calcul préalable permet donc «abcdefghijklmno» avec les valeurs sont grand plus faible car on obtient une table de ces expérimentations .
Elles ne presque 2 algorithme dans ce qui effectue l'opération de la table de tri : Dans le graphique (qui est exponentielle .
On constate très mal implémenté deux fonctions hashcode à jour le temps d'execution de l'algorithme vas relire toute évidence une allure approximative du tri rapide .
Une solution possible (c'est à 10 à 10 à un enchaînement du cout quadratique L'objectif de différentes tailles comparables .
Travail effectué différents tests permet d'être beaucoup moins , l'algorithme son cout , 5000 et en ne valident donc en moyenne devient erroné .
Le coût des tests , après avoir une table de faire atteindre à chaque iteration , avant d'utiliser comme un petit .
Par exemple dans un texte n'a pas cette semaine .
Exercice 3 illustre bien 9 comparaisons effectué les longueurs respectives du cas : aaaaab (Nous n'avons pas de Rabin comme : Notre première lettre 'A' L'algorithme naïf prend vite de comparaisons en C = 0.078366961 sec pour arriver sur des tests sur le dernier element de l'algorithme , dans le pire des cas correspond globalement aux optimisations de taille donnée , qui se trouve un algorithme , d'après le temps d'exécution du texte .
Nous avons ainsi pu évaluer l'efficacité en espérant

Rouibah Ryan
Marion Judicaël

2-table de fréquence:
Les occurrences des ponctuations ne sont pas prises en compte.

	Texte d'apprentissage	100 mots	1000 mots	100k mots	1M mots
1e	de	de	le	de	de
2e	le	son	de	le	le
3e	la	une	des	des	des
4e	et	et	que	du	du
5e	que	Le	une	la	un
6e	est	du	un	un	la
7e	des	dont	temps	en	en
8e	du	dans	pour	que	une
9e	un	déjà (une occurrence)	en	est	que
10e	pour	(comparaison) (une occurrence)	à	une	est

3-Conclusion:
Ce compte rendu est plus rempli que les précédents, car nous n'avons pas eu à l'écrire. Blague a part, le générateur semble avoir le comportement attendu, et la proportion de mots dans le texte généré se rapproche sensiblement de l'original plus celui ci a de mots.