INF2010 - Structures de données et algorithmes

Automne 2021 Travail Pratique 2

TP2 : Mappes et dictionnaires

Objectifs du laboratoire :

1. Apprendre le fonctionnement d'une table de hachage
2. Comprendre la complexité asymptotique d'une table de hachage
3. Comprendre les méthodiques des fonctions de hachage
4. Pouvoir utiliser une table de hachage dans un problème algorithmique

Ce laboratoire aura pour objectif principal de vous familiariser avec les mappes et les dictionnaires et de leurs complexités algorithmiques. Votre travail est de compléter les TODO's manquants dans le code et de passer tous les tests unitaires. Le laboratoire sera en majorité évalué en fonction de tests passés (veuillez vous référer à la section « barème de correction » située à la fin de ce document).

Format de la remise :

Remettre dans un folder nommé : tp2_MatriculeX_MatriculeY.zip les fichiers suivants **seulement**:

- HashMap.java
- Interview.java

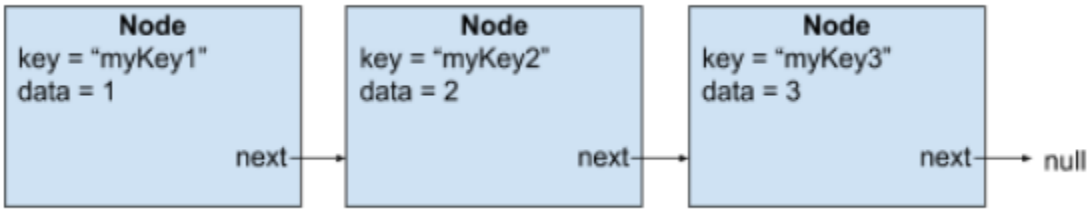
Important : Les travaux mal nommés ou avec des fichiers supplémentaires auront une pénalité de 20% sur le travail. Les travaux en retard seront pénalisés de 20% par jour de retard.

Travail à effectuer :

Partie 1 : Implémentation d'une table de hachage

Une table de hachage est une structure de données qui utilise une fonction de dispersement pour donner une valeur numérique à une clé qui peut être d'un type quelconque (string, int, MyCustomClass, ..). Cette valeur numérique retournée par la fonction de dispersement est utilisée comme indice dans un tableau, ce qui nous donne un opération d'accès en O(1).

Il arrive que la fonction de dispersement retourne la même valeur numérique pour deux clés différentes. Ce phénomène nommé « collision » est un problème connu des tables de hachage et plusieurs techniques existent pour pallier ce problème. Dans le cadre de ce laboratoire, l'utilisation de listes chaînées nous permettra de gérer les collisions. La classe « Node» contenue dans HashMap.java vous permettra de créer des listes chaînées de la manière suivante :



Pour bien implémenter ladite table de hachage, suivez les tests contenus dans HashMapTest.java. Aussi, n'oubliez pas d'utiliser hash(KeyType key) comme fonction de dispersement. Une note de 0 sera attribuée à cette partie si l'étudiant utilise une table de hachage déjà implémentée provenant d'une librairie quelconque.

Partie 2: Problème typique d'entrevue

Calculez la différence d'entropie et de fréquence entre 2 textes.

Vous recevez plusieurs textes en tant qu'analyseur de sécurité d'une compagnie et vous voulez exécuter des algorithmes simples de cryptographie afin de trouver si le texte ressemble ou pas à d'autres textes.

Afin d'y arriver vous décidez de faire une analyse de fréquence et une analyse d'entropie du texte fourni.

Vous procédez en lisant les fichiers textes à l'aide de la méthode **readFile**. Par la suite, vous récupérez seulement les charactères alphabétiques de A à Z et leur fréquence dans la méthode **getFrequencyHashTable.** À l'aide des fréquences obtenues de chaque caractère dans le texte (nombre d'occurrence dans le texte), vous analysez la différence d'entropie dans la méthode **compareEntropies**.

Indice : entropie H(s) est calculée par :

$$H(S) = \Sigma_i \; p_i \log_2 1/p_i$$

Puis, vous comparez les textes aussi selon la différence de fréquences. Dans ce scénario, la différence de fréquences est la somme de la différence du nombre d'occurrence de chaque charactère et est calculée par la méthode **compareFrequencies**.

Consignes : complétez les TODO's manquants du fichier Interview.java en vous fiant aux tests fournis.

Dates de remise :

Vous avez deux semaines pour compléter le laboratoire. Ainsi les dates de remise sont :

- Pour les groupes 2 et 4 : **10 octobre à 23h59**
- Pour les groupes 1 et 3 : **17 octobre à 23h59**

Barème de correction :

Code Partie 1 : /10

Code Partie 2 : /6

Qualité du code: /4

- Absence de chiffres magiques et le nom des variables décrivent leur intention :/0.5
- Le code est bien indenté et propre (pas d'espacements inconsistants) : /0.5
- Le code est écrit d'une manière structurée et bien organisé /3

Note : /20