Hachage
L2.2

Le but de l'exercice est de compter le nombre d'apparitions de chaque mot d'un texte. On supposera qu'un mot est une suite d'au plus de 50 caractères, (lue par `scanf("%51s",mot)`). Pour permettre un accès rapide aux mots, on utilise une méthode appelée *hachage ouvert*. Elle consiste à ranger les mots dans un tableau de listes triées, l'indice de la liste à laquelle peut appartenir un mot est calculé à l'aide d'une fonction `int Hachage(char Mot[])`. Le tableau sera formé de N listes (`#define N 1000`), l'indice sera donc calculé modulo N. La recherche d'un mot se ramène à la recherche dans une liste (courte) plutôt que dans toute la table.
Chaque liste est triée dans l'ordre croissants des mots. L'ordre entre 2 mots est fourni par la fonction `strcmp(char *,char *)`; (`/* man strcmp*/`).
En utilisant le type suivant :

```
typedef struct celmot{
 char *mot; /*adresse de la chaine de caracteres*/
 int nombre;/* nombre d'apparitions*/
 struct celmot *suivant;
} CelluleM, *ListeM;
```

1. Écrire la fonction `ListeM Allouecellule(char M[])`. La nouvelle cellule est créée en utilisant la place mémoire minimale pour recopier M et le nombre d'apparitions est égal à 1.

2. Écrire une fonction `int AjouteListe(ListeM *L, char M[])` qui ajoute le mot M à la liste triée (*L) : si M était déjà présent dans la liste on augmente son nombre d'apparitions, sinon on insère une nouvelle cellule.

3. Écrire la fonction `int Hachage(char M[])` qui renvoie la somme :

$$\sum_{i=0}^{M[i]='\backslash 0'} (i+1)M[i] \pmod{N}$$

4. Écrire une fonction `int Recherche(ListeM T[], char Mot[])` qui renvoie le nombre d'apparitions du mot M.

5. Écrire une fonction `int AjouteDico(ListeM T[], char M[])` qui ajoute le mot M au tableau s'il n'était pas présent et augmente son nombre d'apparitions sinon. La fonction renvoie 0 ou 1.

6. Écrire une fonction `int NombreMot(ListeM T[])` qui renvoie le nombre total de mots du texte mémorisés dans T. Les nombres d'apparitions seront prises en compte.

7. **Traitement de fichier**
   On veut écrire un programme qui lit les mots d'un fichier et crée un nouveau fichier contenant la suite de ces mots et de leur nombre d'occurences.
   Le fichier de sortie contient un mot par ligne, les mots apparaissant en ordre quelconque.
   La référence du fichier dans lequel on lit le texte et celle du fichier dans lequel on écrit les mots du texte seront fournis comme paramètres sur la ligne de commande.
   Ainsi si l'exécutable a pour nom `Compte`, l'exécution de la ligne `Compte /home/zip/Examen /home/zip/Examen.cpt`
   entraine la création du fichier
   `/home/zip/Examen.cpt`.
   Si le contenu du fichier `/home/zip/Examen` est : `examen d'informatique tres tres facile.`

   celui du fichier `/home/zip/Examen.cpt` sera :
   ```
   tres 2
   examen 1
   d'informatique 1
   facile.  1
   ```
   Écrire la fonction `main` permettant ce traitement.

8. Comment choisir la fonction de hachage pour n'avoir qu'une liste?

9. Tester différentes fonctions de hachage, leur temps d'excution approximatif, et comparer la taille des listes obtenues

   - nombre de listes vides,
   - nombre d'éléments dans la plus longue liste,
   - nombre moyen d'éléments dans les listes non vides,
   - ....