Travaux dirigés 3

Tutorat IMAC

Lexique des mots d'un fichier-Listes

Ce TD est une adaptation du TD précédent qui utilise une structure de liste à la place de la structure de tableau.
On appelle mot une suite de caractères ne contenant pas d'espace, de passage à la ligne ou de tabulation, c'est à dire une chaine de caractères lue avec le format `%s` par `scanf`.On pourra considérer que les mots font moins de 500 caractères et utiliser le format `%500s` pour lire un mot dans un tableau de 501 char.
On veut écrire un programme qui construit la suite triée des 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 et les mots apparaissent en ordre lexicographique.
La référence du fichier dans lequel on lit les mots sera fournie sur la ligne de commande, la référence du fichier dans lequel on écrit sera celle du fichier dans lequel on lit, suffixée par ".cpt". Ainsi si l'exécutable a pour nom `Compte`, l'éxécution de la ligne `Compte /home/zip/Exemple`
entrainera la création du fichier
`/home/zip/Exemple.cpt`.
Si le contenu du fichier `/home/zip/Exemple` est :

```
exemple d'informatique tres tres facile.
```

celui du fichier `/home/zip/Examen.cpt` sera :

```
d'informatique  1
exemple 1
facile. 1
tres 2
```

Le nombre de mots étant inconnu avant traitement on utilisera une structure allouée dynamiquement :

```
typedef struct mot{
  int occurence;
  char * chaine;
  struct mot * suc;
  }Mot;

typedef Mot * Lexique;
```

On a donc ajouté à la structure `Mot` un champs successeur de type `Mot*` qui correspond l'adresse du mot suivant dans le chainage de la liste. Le champs occurence correspond toujours au nombre d'occurence du mot et le champs chaine contient la chaine de caractères du mot lu. Le type `Lexique` est à présent une liste de `Mots` c'est-à-dire un type `Mot*` qui correspond l'adresse du premier mot de la liste. Les noms des types sont identiques à ceux du td 2 ce qui permet de réutiliser certains appels de fonctions.
Les mots seront ajoutés en ordre lexical.

1. Ecrire le programme.
2. Ecrire une fonction récursive qui affiche les mots du lexique dans le fichier .cpt dans l'ordre lexicographique inverse. Pour l'exemple ci-dessus, après appel de cette fonction, le contenu du fichier `/home/zip/Examen.cpt` sera :

   ```
   tres 2
   facile. 1
   exemple 1
   d'informatique  1
   ```

3. Pour finir, on veut trier les mots du lexique par ordre de nombre d'occurences croissant. Pour cela on effectuera un tri fusion.
   Etant donnée une liste l, on procédera de la facon suivante :
   une première fonction coupe la liste en son milieu pour obtenir deux listes l_1 et l_2 ;
   la fonction de tri est appelée récursivement pour trier l_1 et l_2 et enfin la fonction fusion recompose la liste l triée en utilisant les cellules de l_1 et l_2.