TP6
Programmation C L2.1
Lexiques

Le but de cette séance est de compter les mots dans un texte. Mais contrairement au programme `wc` écrit dans un TP précédent, ici on compte les mots séparément : on maintient dynamiquement une liste des mots qui apparaissent au cours de la lecture et on compte le nombre d'occurences de chacun.

---

1 Lecture d'un mot dans un fichier

- Écrire une fonction `int lire_mot(FILE *fichier, char **s)` qui recherche le premier mot dans un fichier. Ce mot est copié dans une chaîne de caractère à l'adresse `*s`. La fonction renvoie 0 si elle ne trouve aucun mot et 1 sinon.

2 Création d'un lexique

Pour maintenir un lexique (c'est-à-dire un ensemble de mots, et un nombre d'occurence pour chaque mot) on utilise les structures suivantes.

```
typedef struct Mot {
  char *mot;
  int occur;
} mot;

typedef struct Lexique {
  mot *lexique;
  int nb_mots;
  int nb_max;
} lexique;
```

- Écrire une fonction `int ajoute_mot(lexique *lex, char nouveau_mot[])` qui ajoute un mot dans un lexique. Si le mot est déjà présent, le nombre d'occurences est augmentée de 1. Sinon, on ajoute le mot à la liste, en maintenant l'ordre (c'est une liste triée).

  La variable `nb_mots` doit toujours être inférieure à la variable `nb_max`. Si nécessaire, on augmente `nb_max` en utilisant la fonction `realloc`, par exemple `nb_max` devient `2*nb_max`. La fonction `realloc` étant coûteuse, on l'appelle le moins souvent possible.

- Écrire une fonction `void free_lexique(lexique *lex)` qui libère l'espace mémoire.

3 Lecture et écriture de fichiers

- Écrire un programme qui prend un ou plusieurs noms de fichiers en argument, et pour chacun d'eux, calcule son lexique. Chaque lexique est écrit dans un autre fichier dont le nom est obtenu en ajoutant `.lex` au nom du fichier de départ. Le format des fichiers créés est tel que chaque ligne contient un mot, puis un espace, puis le nombre d'occurence du mot.