IN405 – Feuille de TD #1
Rappels de langage C

Objectif : rappels de langage C à travers l'implémentation d'un lexique.
Instructions :

- *vous aurez besoin de l'archive* `td1-contents.tar`*, la description de l'archive est donnée dans le fichier* `README.md`*, à lire avant de vous lancer dans l'implémentation ;*
- *l'ensemble du code demandé (sauf question 8) est à écrire dans le fichier* `lexc.c`*.*

Exercice 1.1 - Lettre, mot, lexique, etc.

Soit un lexique, un ensemble de mots faisant partie d'un vocabulaire donné. Un mot est constitué de lettres (caractères **alphabétiques** et **minuscules**).

1- Il existe déjà en C une représentation *native* d'une lettre et d'un mot, lesquels ? Citez des exemples de structure de données pouvant représenter un lexique. Laquelle est pour vous la plus intéressante ?

2- Écrivez le corps de `struct lexc`, qui contient en plus de l'ensemble des mots, le nombre de mots contenus, le nom du lexique et un booléen indiquant si oui ou non vous pouvez modifier le lexique.

3- Écrivez les corps des fonctions `lexc_init()` et `lexc_fini()`, dont le comportement est indiqué en commentaire de la fonction.

Attention : pour la suite des exercices (de ce TD et des suivants), il vous est demandé d'écrire également le bloc de commentaire de chaque fonction.

Exercice 1.2 - Vocabulons

Pour pouvoir utiliser un lexique, il faut lui définir un ensemble d'actions. Les actions possibles sont : l'ajout de mot, la suppression de mot et la vérification de l'existence d'un mot. Les fonctions retournent 0 si l'action s'est bien déroulée, 1 sinon.
Chaque fonction pourra être validée à l'aide des tests unitaires fournis. Pour les activer, décommentez les lignes `test_lexc_xxx()` où `xxx` est l'une des actions, présentes dans la fonction `main()` du fichier `test_lexc.c`.

4- Écrivez le corps de la fonction `lexc_add()`, qui insère le mot donné en paramètre en tant que dernière entrée du lexique. Un mot ne peut pas être présent plusieurs fois dans un même lexique.

5- Écrivez le corps de la fonction `lexc_remove()`, qui enlève du lexique le mot donné en paramètre.

6- Écrivez le corps de la fonction `lexc_check()`, qui recherche si le mot donné en paramètre est une entrée du lexique.

Exercice 1.3 - Interprétation des commandes

L'appel au programme `lexc` demande à l'utilisateur d'entrer des commandes sur l'entrée standard pour qu'il puisse effectuer les actions qu'il souhaite. Chaque commande est écrite sous la forme `action word`, où `action` est le nom de l'action à effectuer et `word` le mot ciblé par l'action.

7- Remplissez les blocs vides de la fonction `cmd_interpret()`, qui traite la commande entrée en paramètre de la fonction, et appelle l'action correspondante. Vous pouvez considérez que si la commande entrée est `quit`, alors on sort du programme.

8- Sur la base des tests d'intégration présents dans le fichier `test_lexc.c`, écrivez un test qui aura le comportement suivant :

#	action	résultat attendu
0	création d'un lexique	succès
1	ajout du mot `mickey`	succès
2	ajout du mot `pluto`	succès
3	ajout du mot `dingo`	succès
4	recherche du mot `horace`	**échec**
5	recherche du mot `pluto`	succès
6	ajout du mot `dingo`	**échec**
7	suppression du mot `mickey`	succès
8	suppression du mot `pluto`	succès
9	recherche du mot `pluto`	**échec**
10	suppression du mot `donald`	**échec**
11	suppression du lexique	—

Exercice 1.4 - *Pour aller plus loin*

*Les exercices "*pour aller plus loin*" sont optionnels et n'ont pour seul but que de vous proposer plus de pratique et donc de vous améliorer. Ils ne seront pas traités/corrigés en TD, mais vous pouvez toujours demander des conseils et/ou poser des questions à vos chargés de TD. Chaque exercice est indépendant des autres et ils peuvent être faits dans n'importe quel ordre, selon vos préférences. Ces exercices se différencient des autres par leur titre italique.*

9- Ajoutez l'action 'liste', qui permet de lister l'ensemble des mots présents dans un lexique. Faites qu'elle soit appelable par le biais de l'interpréteur de commandes.

10- Considérons que les mots du lexique sont rangés dans l'ordre alphabétique, devenant alors un dictionnaire. Modifiez les fonctions des actions d'ajout, de suppression et de recherche pour tenir compte de cette caractéristique.

11- Nous avons considéré que chaque lexique avait un nom. Supposons que ce nom soit unique, permettant alors de les différencier les uns des autres. Modifiez la fonction `cmd_interpret()` pour qu'elle puisse gérer un ensemble de lexiques, aussi appelé vocabulaire. Les commandes sont alors modifiées ainsi : `action lexc word`, avec `lexc` le nom du lexique ciblé par l'action.