INF3135 – Construction et maintenance de logiciels
TP2 – Été 2015

Tri unique

Vous devez concevoir un logiciel servant à trier une liste de mots provenant d'un fichier.

Votre logiciel sera lancé à la console en recevant en paramètre un nom de fichier. Le fichier contiendra les mots à trier. Une fois les mots triés, le logiciel devra afficher à l'écran la liste ordonnées sans doublons.

Exemple d'exécution du logiciel :

```
bash> ./tri liste.txt
AMERTUME
BAR
CASSEROLE
PATRON
UNIVERS
bash>
```

Un fichier d'entrée vous sera fourni pour vos tests. Le fichier d'entrée répondra aux spécifications suivantes :

- il peut y avoir plusieurs mots sur une même ligne;
- une ligne ne dépassera pas 80 caractères;
- tous les mots seront en lettres majuscules sans caractères accentués;
- les mots sont séparés par un espace ou un saut de ligne.

Le tri doit éliminer les mots qui apparaissent plus d'une fois dans la liste. Autrement dit, si un mot en particulier est présent 2 fois ou plus dans le fichier d'entrée, il n'apparaîtra qu'une seule fois en sortie.

Statistiques

Lorsque le logiciel sera invoqué avec avec l'option -S, il accumulera des statistiques sur les données du fichier d'entrée et écrira ces statistiques dans un fichier de sortie spécifié à la console.

Exemple d'exécution du logiciel avec l'option -S :

```
bash> ./tri liste.txt -S stats.txt
AMERTUME
BAR
CASSEROLE
PATRON
UNIVERS
bash>
```

Si le fichier de sortie n'existe pas, il sera créé; s'il existe, il sera écrasé.

Voici les statistiques à produire :

- le nombre de mots sans doublons;
- le nombre de mots avec doublons;
- le nombre de lignes dans le fichier d'entrée;
- le nombre de lettres dans la liste de mots (sans doublons);

- la lettre la plus fréquente.

Exigences de conception

La structure de données utilisée pour accumuler les mots et les trier doit être une liste chaînée allouée dynamiquement.

Le logiciel doit être découpé en modules. On devrait, au minimum, retrouver un module pour chaque élément suivant :

- le main;
- la liste chaînée;
- la gestion des statistiques.

Les paramètres reçus par le main (argc et argv) doivent toujours être validés.

Exigences techniques

Le programme doit être rédigé en langage C et doit compiler sans erreur et sans avertissement (compilation avec l'option -Wall) sur le serveur malt de labunix.

Un fichier readme doit être fourni avec votre logiciel. Ce fichier doit contenir la documentation indiquant comment compiler votre programme sur malt et comment l'utiliser.

Un makefile doit être fourni pour compiler adéquatement le logiciel sur malt. Le makefile devrait effectuer la compilation complète avec la cible par défaut (c'est-à-dire avec la commande "make" sans paramètre).

Exigences du code source

Vous devez appliquer les exigences présentes dans le document de Standards de programmation sur le site du cours.

Remise

Ce travail est individuel. Le répertoire de travail contenant les fichiers doit être archivé dans un fichier zip et nommé selon le code permanent de l'auteur. L'archive doit être remise par Moodle avant le 27 juillet 2015 à 13h00. Aucun retard ne sera accepté et une pénalité sera appliquée pour une archive non conforme sans le code permanent.

Pondération

Readme : 5%
Makefile : 10%
Fonctionnalité : 40%
Respect du style et qualité du code : 25%
Respect des exigences de conception : 20%