INF3135 – Construction et maintenance de logiciels
TP1 – Été 2015

Résolution de mots cachés

Vous devez concevoir un logiciel pour découvrir un mot caché dans une grille.

Votre logiciel sera lancé à la console en recevant en paramètre un nom de fichier. Le fichier contiendra la grille de lettres et les mots à trouver. Une fois les mots trouvés, le logiciel devra afficher à l'écran la solution : le mot trouvé à partir des lettres non utilisées dans la grille.

La grille fait une taille de 12 lettres par 12 lettres.

Exemple d'exécution du logiciel :
```
bash> ./motcache test1.txt
MANDOLINE
bash>
```

Un fichier d'entrée vous sera fourni pour vos tests. Les 12 premières lignes du fichier contiendront les 12 lignes de la grille. Ensuite, le fichier contiendra une ligne vide et finalement la liste de mots à trouver. Chaque mot sera sur une ligne différente.

Les lettres d'un mot peuvent apparaître à l'envers dans la grille. En réalité, quatre directions sont possibles :

- de gauche à droite;
- de droite à gauche;
- de haut en bas;
- de bas en haut.

Il n'est pas nécessaire de faire les diagonales, ce cas ne sera pas testé.

Une lettre de la grille peut être utilisée par plus d'un mot. Lorsque tous les mots ont été trouvés, la réponse au mot caché est trouvé en affichant à l'écran toutes les lettres non utilisées, en parcourant la grille de gauche à droite, de haut en bas.

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.

Exigences du code source

Vous devez appliquer les exigences suivantes à votre code source.

- L'indentation doit être de 3 espaces. Aucune tabulation n'est permise dans l'indentation.
- Votre code doit être découpé en fonctions d'une longueur acceptable. Essayez de ne pas dépasser 25 lignes par fonction.
- Chaque fonction doit être documentée avec un commentaire expliquant l'objectif de la fonction, les paramètres et la valeur de retour (si applicable).
- N'utilisez pas de variables globales (sauf errno).

- Les erreurs systèmes doivent être gérées correctement.
- Les fonctions doivent être déclarées avec une ligne de prototype.
- Les identifiants de fonctions et variables doivent être en camelCase.
- Une attention particulière doit être apportée à la lisibilité du code source.

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 14 juin 2015 à 23h59. Aucun retard ne sera accepté et une pénalité sera appliquée pour une archive non conforme sans le code permanent.

Pondération

Readme : 5%
Fonctionnalité : 70%
Respect du style et qualité du code : 25%