Rapport : Arbre lexicographique (DM)

DE JESUS MILITAR Dylan

12 mars 2021

1 Introduction

Ce projet consistait en la conception d'un programme en langage C permettant la création ainsi que la manipulation et l'utilisation d'un lexique formé de mots provenant d'un fichier texte donné en paramètre de la ligne de commande.

2 Utilisation

2.1 Compilation

Le programme se compile de la manière suivante :

gcc ./Lexique [-option1, -option2, ...] NomFichier

Les options pouvant être aussi nombreuses que voulues par l'utilisateur, qui s'il le souhaite à la possibilité en une ligne de commande pouvoir afficher les mots du fichiers en ordre alphabetique, sauvergarder dans un fichier, rechercher, un mot...

Compiler uniquement avec un fichier :

gcc ./Lexique NomFichier

Affichera le menu de référencement des différentes options.

Compiler avec :

gcc ./Lexique [-option1, -option2, ...]

Enverra à l'utilisateur un message d'erreur signifiant que le fichier n'a pas pu s'ouvrir (car inexistant)

En résumer pour lancer correctment le programme l'utilisateur devra avoir trois agument en ligne de commande : l'executable, une suite d'options, un fichier (dans cet ordre)

2.2 Usage du programme

L'utilisation du programme se fait dès la saisie de la ligne de commande à la compilation (voir point précédent)
Une fois les options adéquates entrées, le programme aura trois principaux comportements :

- Affichage en sortie standard (on la retrouve pour l'affichage des mots en ordre alphabétique, la

recherche d'un mot dans un fichier)

- Creation de fichiers de sauvegarde (ce comportement s'effectue en crééant un fichier du nom de fichier dans lequel on lit les mots, suivi d'une extension .DIC, ou .L selon la sauvegarde spécifiée en commande)

- Avertissement dans la sortie standard d'un problème d'utilisation (erreur d'ouverture de fichiers, mauvaise manipulation de la ligne de commande....)

Ainsi les options valides sont :

-l : Affichage en ordre alphabetique

-r "Mot" : Affichage en ordre alphabetique

-s : Sauvegarde de l'affichage en ordre alphabetique dans un fichier NomFichier.L des mots du lexique.

-S : Sauvegarde de l'affichage prefixe dans un fichier NomFichier.DIC des mots du lexique.(selon les instruction du format de sauvegarde donnée dans le sujet)

(Amélioration) -o : Informe en sortie standard du nombre de mot que contient le fichier

3 Description des méthodes

En premier lieu, le programme est constitué d'une fonction main dans laquelle on commence par ouvrir le fichier du dernier argument donné dans la ligne de commande correspondant au nom du fichier que l'on stocke dans une variable de type FILE *.
Suite à cela on effectue un contrôle du placement, de la présence, des arguments selon le schéma (voir point compilation).
En effet on s'assure que argc étant le nombre d'argument du programme doit être supérieur ou égal à 3.
Si cette vérification est juste, on vérifie que le fichier est bien existant, s'il l'est on appelle la fonction FichierEnArbre() dans lequel on passe l'adresse d'une variable (test) de type Arbre et qui recueillera le lexique des mots du fichiers donné lui aussi passé en argument de la fonction.

Suite à cela on regarde pour chaque élément du tableau de commandes (argv de taille argc) d'indice supérieur à 1 pour ne pas prendre l'exécutable et inférieur à argc-1 pour ne pas prendre le fichier. Pour chaque option on regarde si son premier caractère est '-' afin de savoir si cela est bien une option.
Si elle s'avère en être une alors, à l'aide d'un switch on cherche à savoir quelle est la valeur de son second caractère (l, r, s, S, o)
Si elle ne fait pas partie de ces caractères on passe outre, sinon on traite la partie correspondante en appelant les différentes fonctions.

l -> AfficheOrdreAlpha(test, prefixe, 0) : test est l'arbre lexical crée à partir du fichier, prefixe est une chaîne vide, et 0 sa taille

s -> SauvOrdreAlpha(test, prefixe, 0, fichier-sauv) : fichier-sauv etant la variable stockant l'appel à

fopen((strcat(argv[argc - 1],".L"),"w")) permettant de créer le fichier de sauvegarde

S -> SauvFichier(test, fichier-sauv) : fichier-sauv etant la variable stockant l'appel à fopen((strcat(argv[argc - 1],".DIC"),"w")) permettant de créer le fichier de sauvegarde

r -> RechercheMot(test, argv[i + 1], strlen(argv[i + 1])) : argv[i + 1] etant la chaîne située juste après '-r' dans la ligne de commande, etant ainsi le mot à rechercher

o -> NbMots(test, nb-mots) : nb-mots etant la variable entière stockant le nombre d'occurence du marqueur de fin et donc le nombre de mots

À la toute fin on libere evidemment la memoire de l'arbre avec librereArbre()

4 Repartition du travail

Tout le travail sur ce projet a été effectué seul. Bien que cette situation ne résulte d'un choix personnel mais bien d'une obligation. Ainsi la rédaction du rapport est également fruit de mon travail personnel.

5 Difficultés rencontrées

Etant seul à travailler sur ce projet, abandonné par mon binôme il m'a manqué de temps pour :

- Répartir le projet en modules permettant une meilleur lisibilité du projet

- Tester tous les cas possibles d'arbres

- Faire des essais sur tous les cas possibles que l'utilisateur peut engendre à l'usage du programme et notamment des erreurs qu'il pourrait amener au programme

6 Conclusion

Pour conclure ce projet m'a permis de me familiariser avec la manipulation :

- Des arbres et de leur logique

- Des fichiers

- Des arguments de la ligne de commande et notamment la gestion d'erreur liée à celle-ci