==========

trier les lignes d'un fichier texte

C.Turrier – 18 juin 2023

==========

Ce programme lit les lignes d'un fichier d'entrée "**in.txt**" , les trie dans l'ordre alphabétique (en ignorant la casse) et les enregistre dans un fichier de sortie "**out.txt**".

Pour compiler ce programme avec GCC, afin d'obtenir le fichier exécutable **t**, il suffit de saisir la commande suivante depuis le terminal ouvert dans le répertoire où se trouve **t.c** :

```
gcc -Wall t.c -o t
```

Il suffit ensuite de placer l'exécutable **t** dans le répertoire où se trouve le fichier **in.txt** dont les lignes sont à trier

Pour exécuter le programme, il suffit alors de saisir la commande suivante :

```
./t
```

Le fichier **out.txt** est alors créé automatiquement et contient les lignes triées.

Code source du programme

```
/*-------------------------------------------------------
Tri lignes - Auteur C.Turrier - 18 juin 2023
source: t.c
compilation: gcc -Wall t.c -o t
exécution: ./t
Ce programme effectue les tâches suivantes :
1. ouvrir un fichier texte "in.txt"
2. trier toutes les lignes du fichier dans l'ordre alphabétique croissant
3. enregistrer le résultat dans un fichier "out.txt"
Le tri s'effectue sans faire de différences entre les lettres majuscules
et les lettres minuscules correspondante
-------------------------------------------------------------*/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#define MAX_LINE_LENGTH 100
int compare(const void *a, const void *b) {
const char *lineA = *(const char **)a;
```

```
const char *lineB = *(const char **)b;
// Convertir les lignes en minuscules avant la comparaison
char tempA[MAX_LINE_LENGTH];
char tempB[MAX_LINE_LENGTH];
strcpy(tempA, lineA);
strcpy(tempB, lineB);
for (int i = 0; tempA[i]; i++) {
tempA[i] = tolower(tempA[i]);
}
for (int i = 0; tempB[i]; i++) {
tempB[i] = tolower(tempB[i]);
}
return strcmp(tempA, tempB);
}
int main() {
FILE *inFile, *outFile;
char line[MAX_LINE_LENGTH];
char **lines = NULL;
int numLines = 0, i;
// Ouvrir le fichier d'entrée
inFile = fopen("in.txt", "r");
if (inFile == NULL) {
printf("Impossible d'ouvrir le fichier in.txt\n");
return 1;
}
// Lire les lignes du fichier d'entrée
while (fgets(line, MAX_LINE_LENGTH, inFile) != NULL) {
// Allouer de la mémoire pour la nouvelle ligne
char **temp = realloc(lines, (numLines + 1) * sizeof(*lines));
if (temp == NULL) {
printf("Erreur lors de l'allocation de mémoire\n");
return 1;
}
lines = temp;
// Allouer de la mémoire pour la copie de la ligne
lines[numLines] = malloc(strlen(line) + 1);
if (lines[numLines] == NULL) {
printf("Erreur lors de l'allocation de mémoire\n");
return 1;
}
// Copier la ligne dans le tableau
strcpy(lines[numLines], line);
numLines++;
}
// Fermer le fichier d'entrée
fclose(inFile);
// Trier les lignes dans l'ordre alphabétique croissant (en ignorant la
casse)
qsort(lines, numLines, sizeof(*lines), compare);
// Ouvrir le fichier de sortie
outFile = fopen("out.txt", "w");
if (outFile == NULL) {
printf("Impossible d'ouvrir le fichier out.txt\n");
```

```
return 1;
}
// Écrire les lignes triées dans le fichier de sortie
for (i = 0; i < numLines; i++) {
fprintf(outFile, "%s", lines[i]);
// Libérer la mémoire allouée pour chaque ligne
free(lines[i]);
}
// Fermer le fichier de sortie
fclose(outFile);
// Libérer la mémoire allouée pour le tableau de lignes
free(lines);
printf("Les lignes ont été triées et enregistrées dans le fichier out.txt
avec succès.\n");
return 0;
}
```

Fonctionnement du programme

Ce programme effectue les tâches suivantes :

1. Ouvre un fichier d'entrée appelé "in.txt" et vérifie si l'ouverture du fichier est réussie.
2. Lit les lignes du fichier d'entrée, une par une, en utilisant la fonction fgets.
3. Alloue de la mémoire dynamiquement pour stocker chaque ligne lue, en utilisant la fonction realloc pour augmenter la taille du tableau de lignes.
4. Effectue une copie de chaque ligne dans le tableau de lignes.
5. Ferme le fichier d'entrée.
6. Trie les lignes dans le tableau en utilisant la fonction qsort. La fonction compare est utilisée comme fonction de comparaison pour le tri. Cette fonction convertit les lignes en minuscules en utilisant la fonction tolower avant de les comparer en utilisant strcmp.
7. Ouvre un fichier de sortie appelé "out.txt" et vérifie si l'ouverture du fichier est réussie.
8. Écrit les lignes triées dans le fichier de sortie en utilisant la fonction fprintf.
9. Libère la mémoire allouée pour chaque ligne dans le tableau.
10. Ferme le fichier de sortie.
11. Libère la mémoire allouée pour le tableau de lignes.
12. Affiche un message de succès.