==========

Numéroter les lignes d'un fichier texte

C.Turrier – 18 juin 2023

==========

On peut avoir besoins de numéroter les lignes d'un fichier texte, c'est à dire de placer au début de chaque ligne le numéro de cette ligne suivi d'un espace.
Par exemple :

```
\documentclass[14pt,twoside, openright]{extbook}
%==================================================
% PAQUETS PRINCIPAUX
%==================================================
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
...
```

devient

```
0001 \documentclass[14pt,twoside, openright]{extbook}
0002 %==================================================
0003 % PAQUETS PRINCIPAUX
0004 %==================================================
0005 \usepackage[T1]{fontenc}
0006 \usepackage[utf8]{inputenc}
...
```

Le programme **num.c** suivant, écrit en langage C, réalise ce travail automatiquement, ce qui est bien pratique lorsque le nombre de lignes à numéroter est important.

Pour compiler ce programme avec GCC, afin d'obtenir le fichier exécutable **num**, il suffit de saisir la commande suivante depuis le terminal ouvert dans le répertoire où se trouve **num.c** :

```
gcc -Wall num.c -o num
```

Il suffit ensuite de placer l'exécutable **num** dans le répertoire où se trouve le fichier **in.txt** dont les lignes sont à numéroter.

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

```
./num
```

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

Code source du programme

```
/************************************************
Numérote Lignes - Auteur C.Turrier - 18 juin 2023
source: num.c
compilation: gcc -Wall num.c -o num
exécution: ./num
Ce programme numérote les lignes d'un fichier texte in.txt
il crée un fichier résultat out.txt à partir d'un fichier in.txt
ligne1 -> mcdu ligne1
ligne2 -> mcdu ligne2
etc...
************************************************************************/
#include <stdio.h>
#include <string.h>
int main(int argc, char* argv[])
{
FILE* pout = fopen("out.txt", "w+");
FILE* pin = fopen("in.txt", "r");
char line[128];
char newline[128];
int size;
int i;
int n;
char str[12];
char newstr[12];

n=0;

while (fgets(line, sizeof(line), pin))
{
n=n+1;
sprintf(str, "%d", n);

if (n>=0 && n<10)
{
newstr[0]='0';
newstr[1]='0';
newstr[2]='0';
newstr[3]=str[0];
newstr[4]='\0';
}

if (n>9 && n<100)
{
newstr[0]='0';
newstr[1]='0';
newstr[2]=str[0];
newstr[3]=str[1];
newstr[4]='\0';
}
if (n>99 && n<1000)
{
```

```
newstr[0]='0';
newstr[1]=str[0];
newstr[2]=str[1];
newstr[3]=str[2];
newstr[4]='\0';
}
if (n>999 && n<10000)
{
newstr[0]=str[0];
newstr[1]=str[1];
newstr[2]=str[2];
newstr[3]=str[3];
newstr[4]='\0';
}

size=strlen(line);

newline[0]=newstr[0]; newline[1]=newstr[1]; newline[2]=newstr[2];
newline[3]=newstr[3]; newline[4]=' ';

for(i=0;i< size+1;i++)
{
newline[i+5]=line[i];
}
fputs(newline,pout);
}
fclose(pin);
fclose(pout);
return 0;
}
```

Rappel

En langage C, si on déclare char s[128]; puis qu'on écrit l'instruction strcpy(s, "abc");
on obtient les résultats suivants :

- s[0] = 'a'
- s[1] = 'b'
- s[2] = 'c'
- s[3] = '\0' (le caractère nul de fin de chaîne)

Dans ce cas, strlen(s) renverra la valeur 3, car il compte tous les caractères précédant le caractère nul de fin de chaîne.

Si on a un caractère de fin de ligne '0A' à la fin de la chaîne s, on a

- s[0] = 'a'
- s[1] = 'b'
- s[2] = 'c'
- s[3] = '\n' (le de fin de ligne)

s[4] = '\0' (le caractère nul de fin de chaîne)

Dans ce cas, strlen(s) renverra la valeur 4, car il compte tous les caractères précédant le caractère nul de fin de chaîne.

Fonctionnement du programme

```
FILE* pout = fopen("out.txt", "w+");
FILE* pin = fopen("in.txt", "r");
```

Deux pointeurs de fichiers, `pout` et `pin`, sont déclarés pour ouvrir les fichiers "out.txt" en mode écriture avec création (`"w+"`) et "in.txt" en mode lecture (`"r"`), respectivement.

```
while (fgets(line, sizeof(line), pin))
{
…
fputs(newline,pout);
}
```

La boucle while est utilisée pour lire chaque ligne du fichier d'entrée à l'aide de la fonction `fgets()`. La boucle continue jusqu'à ce qu'il n'y ait plus de lignes à lire.

- À l'intérieur de la boucle, le numéro de ligne n est incrémenté de 1.
- La fonction sprintf() est utilisée pour convertir le numéro de ligne en une chaîne de caractères str.
- En fonction de la valeur de n, la variable newstr est mise à jour avec les caractères correspondants du numéro de ligne converti en chaîne. Ces conditions conditionnelles utilisent des comparaisons pour déterminer l'intervalle de valeurs de n et mettre à jour newstr en conséquence.
- La variable size est mise à jour avec la longueur de la ligne lue depuis le fichier d'entrée à l'aide de la fonction strlen().
- Le tableau newline est mis à jour avec les caractères de newstr suivis d'un espace, suivi des caractères de la ligne lue depuis le fichier d'entrée. La boucle for itère sur chaque caractère de line et les ajoute à newline après les premiers 5 caractères réservés pour newstr et un espace.
- La fonction fputs() est utilisée pour écrire la ligne modifiée newline dans le fichier de sortie.

Après avoir parcouru toutes les lignes du fichier d'entrée, les fichiers pin et pout sont fermés à l'aide des fonctions fclose().

Remarques

Structure de ligne de in.txt en entrée
`e[0]; e[1];....e[size-1]; e[size]="\0'"`

Structure de ligne de out.txt en sortie
`s[0]="m"; s[1]="c";s[2]="d"; s[3]="u";s[4]="  ";`
`s[5]=e[0];....s[size+4]=e[size-1]; s[size+5]="\0'"`

le caractère de saut de ligne 0A (en hexadécimal)
correspond à un "\n" en C
Les caractères 0A (éventuel) et 0 présents en fin de ligne e
sont recopiés tels quels en fin de ligne s