IGI 2203 - Sujet TP n°9

T. Grandpierre

Exercice 1 :

Écrire un programme qui lit un fichier caractère par caractère et les affiche à l'écran, ligne par ligne, tout en affichant le code ASCII du caractère entre crochets à côté de chaque caractère affiché. Le programme n'affiche le caractère ASCII que si cela correspond à un caractère affichable (i.e. un octet dont la valeur est comprise entre 32 et 255), le code ASCII (l'entier) est affiché systématiquement.

Le nom du fichier doit être donné en argument du programme.

Vous utiliserez les fonctions d'entrées-sorties fichier "fopen", "fgetc" (voir description en fin de sujet, pas de "feof").

A la fin de son exécution, votre programme doit afficher le nombre de caractères du fichier qu'il a lu.

Exercice 2 :

Écrire un programme capable de copier un fichier (un peu comme la commande unix "cp") : le programme reçoit 2 noms de fichier en argument. Le premier argument est le nom du fichier à copier, le second argument est le nom de la copie de ce fichier.

Testez votre programme sur des fichiers textes (par exemple le fichier source du programme) ainsi que sur des fichiers binaires (jpg ou png).

Exercice 3 :

Ecrire une programme qui copie un fichier dans un autre fichier (comme l'exercice 2), mais qui ne recopie pas toutes les valeurs : il ne recopie pas les espaces et les retours à la ligne (utilisez l'affichage de l'exercice 1 pour connaître les valeurs de code ASCII à ne pas recopier).

Exercice 4 :

Modifier le programme de copie précédent pour qu'il chiffre simplement un à un chaque octet du fichier en le remplaçant par le complément (inversion bit à bit) de l'octet lu.

Ainsi, en copiant une première fois un fichier avec ce programme, son contenu devient totalement illisible. Puis en le recopiant une deuxième fois avec ce même programme, on obtient le fichier initial.

Annexe

(tirée de https://koor.fr/C/cstdio/fgetc.wp mais attention l'exemple avec utilisation de la fonction feof est à proscrire, voir les conseils donnés plus loin.)

Entête à inclure

```
#include <stdio.h>
```

Fonctions *fgetc*, *getc* et *getchar*

```
int fgetc( FILE * stream );
int getc( FILE * stream );
int getchar();
```

Ces fonctions permettent toutes les trois de lire le caractère à la position courante du flux considéré. La fonction `getchar` impose le flux : `stdin`[1]. Pour les deux autres fonctions, le flux de caractères à utiliser doit être spécifié en paramètre. Une fois le caractère lu, la position de la tête de lecture associé au flux considéré est décalée sur le prochain caractère à lire.

En fait `fgetc` et `getc` sont identiques au détail près que sur certains compilateurs la fonction `getc` est implémentée sous forme d'une macro.

Paramètres

- **stream**: ce paramètre permet d'indiquer le flux de caractères (ou d'octets) à utiliser pour la lecture. N'oubliez pas que la valeur de ce paramètre à été initialement capturée lors de l'invocation de la fonction `fopen`.

Valeur de retour

Soit un octet à correctement été lu, et dans ce cas sa valeur vous est retournée (entre 0 et 255), soit une erreur s'est produite et dans ce cas la valeur `EOF` vous sera retournée. Cela donne 257 valeurs différentes pouvant

[1] Stdin correspond à l'entrée standard, c'est-à-dire le clavier. Getchar permet donc de lire une touche tapée au clavier.

être renvoyées : cela explique que le type de retour soit `int` plutôt que `unsigned char`.

Conseil : pour lire un fichier en entier il faut par exemple écrire une boucle while et dans la condition d'arrêt il faut à la fois lire dans le fichier avec fgetc, récupérer l'octet lu et tester sa valeur : si elle est égale à EOF c'est que la fin de fichier est atteinte, il faut alors sortir de la boucle.

Fonctions *fputc*, *putc* et *putchar*

Entête à inclure

```
#include <stdio.h>
```

Fonctions *fputc*, *putc* et *putchar*

```
int fputc( int character, FILE * stream );
int putc( int character, FILE * stream );
int putchar( int character );
```

Ces fonctions permettent d'écrire un caractère sur le flux de caractères passé en paramètre (sauf dans le cas de `putchar` ou là `stdout` est imposé). Une fois le caractère écrit dans le flux, la position de la tête de lecture sera déplacée pour la prochaine écriture.

En fait `fputc` et `putc` sont identiques au détail près que sur certains compilateurs la fonction `putc` est implémentée sous forme d'une macro.

Paramètres

- **character**: ce paramètre stocke le caractère à écrire dans le flux. Le fait que ce caractère soit typé comme un `int` (et non un pas `char`) trouve son origine dans l'utilisation de la fonction fgetc. Effectivement, celle-ci peut soit retourner un caractère (donc 256 possibilités différentes) soit un code d'erreur, qui se doit donc d'être un 257 état potentiel. En conséquence, le type de retour de la fonction fgetc est donc `int`. Le fait que `fputc` accepte un `int` permet donc la chose suivante : `fputc( fgetc( sourceStream ), destStream );`.
- **stream**: ce paramètre permet d'indiquer le flux de caractères (ou d'octets) à utiliser pour l'écriture. N'oubliez pas que la valeur de ce

paramètre à été initialement capturée lors de l'invocation de la fonction `fopen`.

Valeur de retour

En cas de succés, ces fonctions doivent renvoyer le caractère envoyé dans le flux. Par contre, en cas d'erreur, la valeur `EOF` sera retournée. Dans ce cas, il vous faudra alors consulter la variable `errno` pour obtenir de plus amples informations sur l'erreur constatée.