

Fichiers

François Trahay



Entrées-sorties bufferisées

- Le système¹ fournit des primitives d'entrées/sorties (E/S) bufferisées
 - permet d'accéder au contenu de fichiers
 - *bufferisées*: les E/S sont d'abord groupées en mémoire, puis exécutées sur le périphérique
 - permet d'améliorer les performances
 - exemple: 1024 écritures de 1 octet sont regroupées en une écriture de 1ko (donc gain important en performances)
 - FILE*: type "opaque" désignant un fichier ouvert

¹ Pour être exact, il s'agit de la bibliothèque standard (la libc)

Ouverture/fermeture

- `FILE* fopen(char* fichier, char* mode);`
 - mode: mode d'ouverture
 - "r" : lecture
 - "w" : écriture. Curseur au début du fichier
 - "a" : ajout. Curseur à la fin du fichier
 - "r+", "w+", "a+" : lecture et écriture
- `int fclose(FILE* f);` * Complète les opérations et ferme le fichier

Primitives d'écriture

- `int fprintf(FILE* f, char* format, ...);`
 - similaire à `printf`, mais écrit dans le fichier `f`
 - écrit une chaîne de caractères dans un fichier
- `size_t fwrite(void* ptr, size_t size, size_t nmemb, FILE* f);`
 - écrit les `size×nmemb` octets situés à l'adresse `ptr` dans `f`

Primitives de lecture

- `int fscanf(FILE* f, char* format, ...);`
 - similaire à `scanf`, mais lit depuis le fichier `f`
- `size_t fread(void* ptr, size_t size, size_t nmem, FILE* f);`
 - lit `nmem×size` octets et les stocke à l'adresse `ptr`
 - retourne le nombre d'items lus
 - `fread` renvoie une valeur `< nmem` si la fin du fichier (EOF) est atteinte
- `char* fgets(char* s, int size, FILE* f);`
 - lit au plus `size` caractères et les stocke dans `s`
 - arrête la lecture avant `size` si lit `\n` ou EOF

Curseur

- Position dans le fichier à laquelle la prochaine opération aura lieu
 - Initialisé à 0 (le début du fichier) si le fichier est ouvert en mode `r`, `r+`, `w`, `w+`
 - Positionné à la fin du fichier si ouvert en mode `a` ou `a+`
- Avance à chaque opération de lecture/écriture
- `long ftell(FILE *stream);`
 - Indique la position courante (en nombre d'octets depuis le début du fichier)
- `int fseek(FILE *f, long offset, int whence);`
 - déplace le curseur de `offset` octets depuis
 - le début du fichier (si `whence` vaut `SEEK_SET`)
 - la position courante (si `whence` vaut `SEEK_CUR`)
 - la fin du fichier (si `whence` vaut `SEEK_END`)