

# Fichiers

François Trahay



**CSC4103 – Programmation système**  
**2019–2020**

# 1 Entrées-sorties bufferisées

- Le système <sup>a</sup> 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

---

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

## 1.1 Ouverture/fermeture

■ `FILE* fopen(char* fichier, char* mode);`

◆ mode: mode d'ouverture;

- ▶ "r" : lecture seule
- ▶ "w" : écriture seule
- ▶ "r+" ou ("w+") : lecture et écriture
- ▶ "a" : écriture seule (ajout)
- ▶ "a+" : lecture et écriture (ajout)

■ `int fclose(FILE* f);`

◆ Complète les opérations et ferme le fichier

## 1.2 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`

## 1.3 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 nmemb, FILE* f);`
  - ◆ lit  $\text{nmemb} \times \text{size}$  octets et les stocke à l'adresse `ptr`
  - ◆ retourne le nombre d'items lus
  - ◆ `fread` renvoie une valeur  $< \text{nmemb}$  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

## 1.4 Curseur

- Position dans le fichier à laquelle la prochaine opération aura lieu
  - ◆ Initialisé à 0 (le début du fichier) généralement
  - ◆ sauf si ouvert en mode "a" ou "a+" (dans ce cas: positionné à la fin du fichier)
- 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)