## ****1o = 8bits****

## ****⚙️ Types & tailles (64 bits)****

| Type | Taille | Format | Note |
| --- | --- | --- | --- |
| char | 1 o | %c | caractère |
| char \* | 8 o | %s | chaîne C (\0) |
| int | 4 o | %d | signé |
| long | 8 o (Linux) / 4 o (Win) | %ld | dépend OS |
| long long | 8 o | %lld |  |
| float | 4 o | %f |  |
| double | 8 o | %f (printf) |  |
| void \* | 8 o | %p | cast (void\*) |
| size\_t | 8 o | %zu | taille / sizeof |

## 📂 ****Fichiers****

### fopen / fclose

FILE \*fopen(const char \*path, const char \*mode);

int fclose(FILE \*stream);

| Mode | Signification |
| --- | --- |
| "r" | lecture texte |
| "w" | écriture (écrase) |
| "a" | ajout (append) |
| "r+" | lecture/écriture |
| "w+" | lecture/écriture (efface) |
| "a+" | lecture/ajout |
| "rb", "wb", "r+b" | mode binaire |

⚠️ Toujours tester :

if (!f) { perror("fopen"); return 1; }

### fread / fwrite

size\_t fread(void \*ptr, size\_t size, size\_t n, FILE \*f);

size\_t fwrite(const void \*ptr, size\_t size, size\_t n, FILE \*f);

→ retourne nombre d’éléments lus/écrits.  
⚠️ vérifier feof(f) et ferror(f).

### fseek / rewind

int fseek(FILE \*f, long offset, int origin); // SEEK\_SET, SEEK\_CUR, SEEK\_END

void rewind(FILE \*f); // = fseek(f, 0, SEEK\_SET)

### read / write (UNIX, bas niveau)

ssize\_t read(int fd, void \*buf, size\_t count);

ssize\_t write(int fd, const void \*buf, size\_t count);

⚠️ fichiers ouverts avec open() (#include <unistd.h>).  
Retourne nb d’octets lus/écrits ou -1 si erreur.

### fgets

char \*fgets(char \*buf, int n, FILE \*f);

→ lit au plus n-1 caractères, garde le \n.  
⚠️ ret NULL si EOF ou erreur.

## 🔤 ****Chaînes (string.h)****

int strcmp(const char \*s1, const char \*s2); // 0 si égal

char \*strcpy(char \*dest, const char \*src); // ⚠️ pas de vérif taille

char \*strcat(char \*dest, const char \*src); // concatène, ⚠️ débordement

✅ Préférer strncpy, strncat pour plus de sûreté.

## 📊 ****Stat (informations fichier)****

#include <sys/stat.h>

int stat(const char \*path, struct stat \*buf);

Structure utile :

struct stat s;

stat("fichier.txt", &s);

printf("%ld octets\n", s.st\_size);

## ⚠️ ****Erreurs & vérifs****

#include <errno.h>

#include <string.h>

perror("message"); // affiche message + erreur système

fprintf(stderr, "Erreur: %s\n", strerror(errno));

⚡ Bon réflexe :

FILE \*f = fopen("x.txt", "r");

if (!f) { perror("fopen"); exit(EXIT\_FAILURE); }

## ⚡ ****Tips express****

* %lf → seulement pour scanf.
* %p → cast (void\*)ptr.
* printf("%zu", sizeof(x)); ✅
* fflush(stdout); → force affichage.
* gets() → **interdit**.
* Toujours #include <stdio.h> et #include <string.h>.
* Vérifie systématiquement les retours de fonctions (NULL, EOF, <0).

