HE VD IG

Série 0x21

Solution

Premiers pas avec les fichiers

INFO2-TIN — 3 mars 2026, 7efbe5

Exercice 1 : Choix multiples

(a) Qu'est-ce qu'un **flux (stream)** en programmation C (et plus généralement en informatique) ?

A. Un flux est une variable spéciale qui permet de stocker plusieurs valeurs dans la mémoire.
B. Un flux est un mécanisme permettant de lire ou d'écrire des données de manière séquentielle entre un programme et une source ou une destination (fichier, clavier, écran, réseau, etc.).
C. Un flux est un type de boucle utilisé pour répéter des instructions jusqu'à la fin d'un fichier.
D. Un flux est un composant matériel qui transporte physiquement les données dans l'ordinateur.
E. Un flux est une suite de données qui doit obligatoirement être entièrement chargée en mémoire avant de pouvoir être utilisée.
F. Un flux est une fonction du langage C utilisée uniquement pour afficher du texte à l'écran.

(a) B

(b) Tout programme exécuté ouvre trois flux ; quels sont leur noms et leur direction (entrée/sortie), selon la convention de numérotation des flux ?

1. **stdin, entrée**
2. **stdout, sortie**
3. **stderr, sortie**

(c) Quelle est la fonction de la bibliothèque standard utilisée pour positionner manuellement le **curseur** dans un fichier ?

A. `fopen` B. `fgets` **C. `fseek`** D. `fputc` E. `feof`

(c) C

(d) Quel est le mode à transmettre à l'appel `fopen("f.txt", mode)` pour ouvrir un fichier existant en mode **binaire** en **lecture écriture** ?

A. `"r"` B. `"w"` **C. `"rb+"`** D. `"aw"` E. `"w+"`

(d) C

(e) Le caractère `\0` peut-il théoriquement apparaître dans quel type de fichier ?

A. Texte **B. Binaire**

(e) B

(f) Lors de l'appel d'un programme avec `./a.out` et l'exécution de l'instruction `fgetc(stdin)`, que se passe-t-il à l'écran ?

A. Le programme affiche une erreur car `stdin` n'est pas un fichier.
B. Le programme affiche « Enter a character : » et attend que l'utilisateur saisisse un caractère.
C. Rien ne s'affiche, le programme est mis en pause par le système d'exploitation en attendant que l'utilisateur saisisse un caractère suivi de la touche « Entrée ».
D. Le programme affiche « EOF » et se termine immédiatement.
E. Le programme ne s'exécute pas car `fgetc` doit prendre `argv[1]` comme argument.

(f) C

(g) Quelle syntaxe POSIX utiliser pour rediriger la sortie du programme `a` vers l'entrée de `b` ?

A. `a -> b` B. `a > b` **C. `a | b`** D. `a b`

(g) C

(h) En UTF-8 combien d'octets minimum sont nécessaires pour encoder le caractères 'é' ?

A. 1 **B. 2** C. 3 D. 4 E. 5

(h) B

(i) Quel système d'exploitation utilise la convention `CRLF` (ou `\r\n`) pour indiquer la fin d'une ligne dans un fichier texte ?

A. Linux **C. Windows** E. Android
B. macOS D. Unix F. Spark

(i) C

(j) Pour lire les données d'un fichier texte caractère par caractère, jusqu'à la fin du fichier, quelle stratégie adopter ?

A. `for (char c = getc(fp); c != EOF; c = getc(fp)) { ... }`
B. `while (getc(fp) != EOF) { char c = getc(fp); ... }`
C. `int c; while ((c = getc(fp)) != EOF) { ... }`
D. `while (!feof(fp)) { char c = getc(fp); ... }`
E. `while (true) { char c = getc(fp); if (c == EOF) break; ... }`

(j) C

> On privilégie toujours `while` pour une boucle dont le nombre d'itération n'est pas connu à l'avance. De surcroît, il est important de toujours spécifier explicitement la condition de maintien de la boucle, plutôt que de faire un `while (true)` avec un `break` à l'intérieur.

Exercice 2 : Lecture de code

Que fait le programme suivant ?

```c
#include <stdio.h>
char map[256] = {0};
int main() {
    FILE *fp = fopen("foo.txt", "r");
    char c;
    map['a'] = 1; map['e'] = 1;
    while ((c = fgetc(fp)) != EOF) putchar(map[c % 256] ? '*' : c);
    fclose(fp);
}
```

> Le programme lit le fichier `foo.txt` caractère par caractère, et affiche chaque caractère à l'écran. Si le caractère est 'a' ou 'e', il affiche '*' à la place.

Exercice 3 : Programmation

(a) Réalisez un programme en C qui prend en paramètre le nom d'un fichier texte et affiche sa taille en byte sur la sortie standard. Voici un exemple de fonctionnement du programme :

```
echo "Hello, World!" > test.txt
./file_size test.txt
13
```

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

int main(int argc, char *argv[]) {
    if (argc != 2) {
        fprintf(stderr, "Usage: %s <filename>\n", argv[0]);
        return EXIT_FAILURE;
    }
    FILE *fp = fopen(argv[1], "r");
    if (fp == NULL) {
        perror("Error opening file");
        return EXIT_FAILURE;
    }
    fseek(fp, 0, SEEK_END);
    long size = ftell(fp);
    fclose(fp);
    printf("%ld\n", size);
    return EXIT_SUCCESS;
}
```

(b) Vous disposez d'un pointeur sur un fichier ouvert en lecture `fp` et vous souhaitez connaître la taille de ce fichier. Écrire une fonction `size_t fsize(FILE *fp)` qui retourne la taille du fichier

```c
size_t fsize(FILE *fp) {
    if (fp == NULL) return 0;
    size_t size = 0;
    fseek(fp, 0, SEEK_END);
    size = ftell(fp);
    fseek(fp, 0, SEEK_SET);
    return size;
}
```

(c) Écrire un programme qui prend le nom d'un fichier texte en argument ainsi qu'un texte à rechercher. Votre programme doit afficher le numéro de toute ligne du fichier contenant le texte recherché.

```c
int main(int argc, char *argv[]) {
    assert(argc == 3);
    char *filename = argv[1];
    char *search = argv[2];
    FILE *fp = fopen(filename, "r");
    assert(fp != NULL);
    char line[1024];
    size_t line_no = 0;
    while (fgets(line, sizeof(line), fp) != NULL) {
        line_no++;
        if (strstr(line, search) != NULL) {
            printf("%d\n", line_no);
        }
    }
}
```

Exercice 4 : Algorithmique

Sous forme d'un diagramme en flux, proposer un algorithme pour compter le nombre de mot d'un fichier

texte ASCII, sachant que chaque caractère est lu un par un.

> ![Algorithme de comptage de mots](../../assets/words.drawio){width=60%}
> On peut également présenter l'algorithme de manière textuelle en pseudo-code impératif :

```
nbMots <- 0
dansMot <- faux

TANT QUE (il y a un caractère c à lire dans le flux) FAIRE
    SI (c est un séparateur) ALORS
        dansMot <- faux
    SINON
        SI (dansMot = faux) ALORS
            nbMots <- nbMots + 1
            dansMot <- vrai
        FIN SI
    FIN SI
FIN TANT QUE

AFFICHER nbMots
```