


IFIPS
 Institut de Formation d'Ingénieurs
 UNIVERSITÉ PARIS-SUD 11

Introduction au langage C

Neuvième cours Les fichiers

Aurélien Max
 aurelien.max@limsi.fr


Introduction au langage C

Rappel de l'épisode précédent

- Il n'existe pas de type prédéfini pour les chaînes de caractères en C mais une convention:
 - tableau de caractères
 - terminateur de chaîne (`\0`)
- L'accès aux chaînes peut se faire caractère par caractère en utilisant des pointeurs sur caractère, ou en utilisant des fonctions de la bibliothèque standard (`string.h`):
 - entrées/sorties: `puts`, `gets`, `printf`, `scanf`
 - taille des chaînes: `strlen`
 - copie de chaînes: `strcpy`, `strncpy`
 - concaténation de chaînes: `strcat`, `strncat`
 - comparaison de chaînes: `strcmp`
 - recherche dans des chaînes: `strchr`, `strstr`, ...
 - éclatement d'une chaîne: `strtok`
 - conversion de chaîne: `strtod`, `strtoul`, ...
- Support pour des expressions régulières simples (`regex.h`)

2


Introduction au langage C

Rappels sur les fichiers

- Type d'archivage
 - Forme binaire:
 - recopie des données telles qu'elles sont en mémoire
 - ne peut être relue que dans un environnement utilisant le même codage des données
 - Forme formatée:
 - suite d'octets contenant des codes de caractères
 - ne peut être relue que dans un environnement utilisant le même codage de caractères: beaucoup plus portable que la forme binaire
- Type d'accès
 - Accès séquentiel:
 - on accède aux données (en lecture et écriture) dans l'ordre dans lequel elles apparaissent
 - Accès direct:
 - on accède aux données (en lecture et écriture) par leur numéro d'ordre
 - les enregistrements (blocs de données) doivent être de la même taille
 - possible uniquement sur certains supports (ex: pas sur les bandes magnétiques)

3


Introduction au langage C

Les fichiers en C

- Les fichiers sont considérés comme des suites d'octets quelconque (pas de notion d'enregistrements)
- Les entrées-sorties en C sont un cas particulier de fichiers (avec `stdin` et `stdout` comme flux)
- En plus des fonctions d'entrées-sorties (clavier/écran), C dispose de fonction permettant d'écrire et de lire dans des fichiers sans formatage (données binaires)
- Support pour l'accès séquentiel et l'accès direct aux données sur disque

4


Introduction au langage C

Notion de flux

- Au moment de son ouverture, un fichier est associé à un nom interne qui est un pointeur sur une structure de type `FILE`
- Une variable de type `(FILE *)` s'appelle un *flux*
- Les propriétés d'un flux incluent notamment:
 - un pointeur à l'intérieur du fichier
 - un indicateur d'erreur
 - un indicateur de fin de fichier

☛ **La norme n'impose pas que des flux de fichiers puissent être copiés**

5


Introduction au langage C

Types de fichiers

- La norme n'impose pas de distinctions entre fichiers binaires et fichiers formatés (texte)
- L'implémentation du compilateur peut donc si elle le souhaite distinguer les deux formats
- On peut ensuite appliquer n'importe quelles opérations, indépendamment du mode d'ouverture:
 - transfert binaire: `fread`, `fwrite`
 - transfert formaté: `fscanf`, `fprintf`, `fgets`, `fputs`
 - transfert mixte: `fputc`, `fgetc`
- Règle à suivre:
 - pour des fichiers qui seront réutilisés ailleurs, on utilise le mode texte afin d'assurer le transcodage entre code interne et code externe
 - pour des fichiers qui ne seront manipulés que par des programmes C dans la même implémentation, on peut utiliser le mode binaire

6

IFIPS Introduction au langage C

Types d'accès

- Le mode d'ouverture ne conditionne pas le type d'accès possible (séquentiel ou direct).
- Les fonctions ne sont pas spécifiques au mode d'accès. Elles modifient simplement un pointeur dans le fichier correspondant à l'emplacement de la prochaine lecture ou écriture.
- La notion d'enregistrement n'existant pas, un pointeur vers un emplacement dans un fichier peut se spécifier à l'octet près.

7

IFIPS Introduction au langage C

Traitement des erreurs (1/3)

- De nombreux types d'erreurs peuvent survenir lorsque l'on manipule des fichiers:
 - ouverture en écriture d'un fichier inexistant
 - droits d'accès incompatibles
 - l'unité de stockage est saturée
 - erreur matérielle
 - rencontre de fin de fichier lors d'une lecture
- Il existe également des types d'erreurs propres au langage C:
 - variable de type FILE * inexistant
 - pointeur pour l'accès direct pointant vers un emplacement inadéquat
- La plupart des erreurs ne mettent pas fin à l'exécution du programme. Cependant, le langage C ne permet une gestion très homogène des erreurs liés à la manipulation des fichiers.

8

IFIPS Introduction au langage C

Traitement des erreurs (2/3)

- La norme ne précise pas comment doit réagir un programme dans le cas où une variable de flux (FILE *) est malformée.
- La fonction `ferror` permet de connaître la dernière erreur associée à une variable de flux. En pratique, certaines implémentations ne répercutent pas toutes les erreurs sur l'indicateur associée à ces variables.
- La fonction `feof` permet de savoir si un flux a rencontré une fin de fichier:
 - il faut qu'une lecture ait été interrompue pour que l'indicateur soit positionné (donc lire le dernier octet d'un fichier ne positionne pas l'indicateur)
 - il est néanmoins possible d'écrire à n'importe quel endroit d'un fichier, tant que l'on ne dépasse pas la capacité de l'unité concernée
 - c'est donc au programme de distinguer les fins de fichiers normales (i.e. attendues) des fins de fichiers anormales
 - en mode d'accès direct, la plupart des implémentations positionnent l'indicateur si le pointeur dans le fichier dépasse la fin du fichier

9

IFIPS Introduction au langage C

Traitement des erreurs (3/3)

Il est donc recommandé de tester les valeurs de retours des différentes fonctions. Cependant:

- certaines retournent un pointeur, d'autres un entier et une valeur EOF (-1) en cas d'erreur (pas nécessairement une fin de fichier...)
- les fonctions ne détectent pas les mêmes types d'erreurs
- l'utilisation d'un tampon d'écriture peut très bien retarder une erreur d'écriture (écriture différée)

10

IFIPS Introduction au langage C

Ouverture et fermeture d'un fichier

- La fonction `fopen` permet d'ouvrir un fichier, `fclose` permet de fermer un fichier ouvert
- Le mode d'ouverture d'un fichier indique:
 - si le fichier doit ou non exister, et s'il doit être créé
 - les types d'opérations possibles (lecture, écriture, ou les deux)
 - indique le type de format (binaire ou texte) pour les implémentations où la distinction se fait

11

IFIPS Introduction au langage C

Ouverture d'un fichier

- Prototype de `fopen`:
`FILE *fopen(const char *fichier, const char *mode)`
- **mode:**
 - principal (obligatoire):
 - r: lecture d'un fichier existant
 - w: écriture dans un nouveau fichier ou écrasement d'un fichier existant
 - a: extension, i.e. ajout de nouvelles informations à la fin d'un fichier existant ou nouveau
 - mode d'ouverture (facultatif):
 - b: flux binaire
 - t: flux texte (par défaut)
 - mise à jour (facultatif):
 - +: autorise à la fois la lecture et l'écriture

12

IFIPS Introduction au langage C

Exemples de modes d'ouverture de fichier

- r: lecture seule et en mode texte
- w: écriture seule et mode texte
- a: extension et mode texte
- rb: lecture seule et mode binaire
- r+: mise à jour (lecture et écriture) en mode texte d'un fichier qui doit exister
- w+: mise à jour (lecture et écriture) en mode texte d'un fichier qui peut exister ou non
- rb+: mise à jour (lecture et écriture) en mode binaire d'un fichier qui doit exister
- ...

13

IFIPS Introduction au langage C

Ecriture en mode binaire

- Ecriture de blocs d'octets consécutifs dans un fichier par la fonction `fwrite`
- Prototype:


```
size_t fwrite(const void *adr, size_t taille, size_t nblocs, FILE *flux)
```

 - `adr`: adresse en mémoire des données copiées vers le fichier
 - `taille`: taille en octet d'un bloc de données
 - `nblocs`: nombre de blocs à écrire
- Valeur de retour: nombre de blocs effectivement écrits
- Erreurs possibles:
 - manque de place sur l'unité
 - pointeur positionné avant le début de fichier
 - erreur matérielle
 - erreur de flux

14

IFIPS Introduction au langage C

Exemple

```
struct point { char nom; int x, y;};
struct point points[NB_POINTS];
int nbBlocs;
FILE *sortie;

/* ... */

sortie = fopen("monFichier", "wb");
if (sortie == NULL) {
    printf("Impossible de créer le fichier");
    exit(-1);
}
nbBlocs = fwrite(&points[0], sizeof(struct point), NB_POINTS,
sortie);
if (nbBlocs != NB_POINTS) {
    printf("Erreur d'écriture dans le fichier");
    exit(-1);
}
fclose(sortie);
```

15

IFIPS Introduction au langage C

Lecture en mode binaire

- Lecture de blocs d'octets consécutifs dans un fichier par la fonction `fread`
- Prototype:


```
size_t fread(void *adr, size_t taille, size_t nblocs, FILE *flux)
```

 - `adr`: adresse en mémoire où seront stockées les données lues
 - `taille`: taille en octet d'un bloc de données
 - `nblocs`: nombre de blocs à écrire
- Valeur de retour: nombre de blocs effectivement lus
- Erreurs possibles:
 - pointeur positionné après la fin de fichier
 - rencontre de la fin de fichier
 - erreur matérielle
 - erreur de flux
- Exemple:


```
nbBlocs = fread(&monPoint, sizeof(struct point), 1,
entree)
if (feof(entree)) /* fin de fichier
if (nbBlocs != 1) /* erreur de lecture
```

16

IFIPS Introduction au langage C

Opérations en mode formaté

Les opérations de lecture/écriture en mode formaté s'effectuent avec les fonction `printf`, `fscanf`, `fputs` et `fgets` (voir cours sur les chaînes de caractères):

```
int fprintf(FILE *flux, const char * format
[,liste_expressions]);
int fscanf(FILE *flux, const char *format
[,liste_adresses]);
int fputs(const char* chaine, FILE *flux);
char *fgets(char *chaine, int nbCar, FILE
*flux);
```

17

IFIPS Introduction au langage C

Exemple

```
char nom;
int x, y, nbVals;
FILE *entree;
/* ... */
entree = fopen("monFichier", "r");
if (!entree) {
    printf("Erreur d'ouverture du fichier");
    exit(-1);
}
while(1) {
    nbVals = fscanf(entree, "%c%d%d", &nom, &x, &y);
    if (feof(entree)) {
        if (nbVals == EOF) { printf("Fin de fichier normale");
        break; }
        else { printf("Fin de fichier anormale!!"); break; }
    }
    if (nbVals != 3) {printf("Erreur de lecture"); break; }
}
fclose(entree);
```

18

IFIPS Introduction au langage C

Fonctions de lecture/écriture de caractères

- `fputc` (`fgetc`) envoie (lit) un caractère sur un flux quelconque:
- Prototypes


```
int fputc(int c, FILE *flux)
int fgetc(FILE *flux)
```
- Le premier paramètre de `fputc` sera converti en caractère non signé
- La valeur de retour de `fputc` est le caractère réellement écrit ou `EOF`
- La valeur de retour de `fgetc` est le caractère lu converti en entier, `EOF` en cas d'erreur ou de fin de fichier

19

IFIPS Introduction au langage C

L'accès direct

- L'accès direct se fait indépendamment du mode d'ouverture d'un fichier.
- Il se réalise par le déplacement d'un pointeur dans le fichier.
- La fonction `fseek` permet de positionner ce pointeur sur un octet quelconque:


```
int fseek (FILE *flux, long deplacement, int origine)
```
- déplacement spécifie un nombre d'octets par rapport à l'origine
- origine:
 - `SEEK_SET`: début du fichier (déplacement ≥ 0)
 - `SEEK_CUR`: position actuelle du pointeur
 - `SEEK_END`: fin du fichier (déplacement ≤ 0)
- Valeur de retour: 0 en cas de succès, valeur non nulle sinon

20

IFIPS Introduction au langage C

Position du pointeur dans un fichier

- La fonction `ftell` permet de mémoriser la position du pointeur pour y revenir plus tard
- Prototype:


```
long ftell (FILE *flux)
```
- Valeur de retour: position courante du pointeur ou -1 en cas d'erreur
- Utilisation possible:


```
long position;
/* ... */
position = ftell(fichier);
/* ... */
fseek(fichier, position, SEEK_SET);
```
- Détermination de la taille d'un fichier:


```
long taille;
/* ... */
fseek(fichier, 0, SEEK_END);
taille = ftell(fichier);
```

21

IFIPS Introduction au langage C

Types d'utilisation de l'accès direct

- Accélération de la consultation d'un fichier existant
 - on ne consulte que les blocs qui nous intéressent
- Mise à jour d'un fichier existant
 - on ne met à jour sur disque que ceux qui doivent l'être
- Création d'un fichier en accès direct
 - permet de ne pas spécifier certaines valeurs lorsqu'elles ne sont pas connues

22