Chapitre 5 : Entrées et sorties

Notes de cours éditées par Alexandre Blondin Massé
modifié par Rachid Kadouche
Construction et maintenance de logiciels
INF3135

Département d'informatique
Université du Québec à Montréal

28 janvier 2018

Table des matières

Caractère par caractère

La fonction `int getchar()` :

- Retourne le prochain caractère lu sur l'entrée standard ;
- Retourne la valeur `EOF` (pour “end of file") si la lecture est terminée ; La valeur `EOF` est également retournée lorsque le caractère `CTRL-D` est saisi au clavier.
- Notez que le type de retour est `int` : permet de traiter le code ASCII étendu.

La fonction `int putchar(int c)` :

- Ajoute un caractère sur la sortie standard.

La fonction `int ungetc(int c, stdin)` :

- Ajoute un caractère sur l'entrée standard.

Exemple

```
#include <stdio.h>
#include <ctype.h>

int main() {
    char c;

    while ((c = getchar()) != '\n') {
        putchar(toupper(c));
    }
    return 0;
}
```

Entrée : `bonjour`
Sortie : `BONJOUR`

Manipulation des entrées/sorties : ligne par ligne

La fonction `char *gets(char *ligne)` :

- Retourne la prochaine ligne lue sur l'entrée standard ;
- Supprime le caractère `\n` en fin de ligne et ajoute le caractère `\0` en fin de chaîne ;
- Retourne `NULL` lorsque le caractère `EOF` est rencontré ;
- Aucun contrôle sur la taille de la ligne lue.

La fonction `int puts(const char *ligne)` :

- Ajoute une ligne sur la sortie standard.

Exemple

```c
#include <stdio.h>

const unsigned int MAX_LIGNE = 20;

int main() {
    char ligne[MAX_LIGNE];
    int i = 0;
    while (gets(ligne) != NULL) {
        printf("%d : %s\n", i++, ligne);
    }
}
```

Résultat :

```
warning: this program uses gets(), which is unsafe.
Bonjour !
0 : Bonjour !
Comment allez-vous ?
1 : Comment allez-vous ?
Très bien.
2 : Très bien.
```

Saisie d'une ligne sécurisée

- Pour prévenir un débordement, on utilise la fonction `fgets`.

```
#include <stdio.h>

int main() {
    char ligne[10];
    fgets(ligne, 10, stdin);
    printf("Ligne : /%s/", ligne);
    return 0;
}
```

Entrée : `Croissants et pâtisseries`
Sortie : `Ligne : /Croissant/`

Formatage

- La fonction `int printf(char *format, ...)` permet d'afficher sur la sortie standard un texte formaté ;
- La fonction `int sprintf(char *chaine, char *format, ...)` permet d'envoyer un texte formaté dans une chaîne ;
- Attention ! Assurez-vous que l'espace mémoire pointé par `*chaine` soit réservé.
- La variable `format` décrit la structure selon laquelle les éléments sont affichés et quel est le type de ces éléments.
- On utilise le symbole `%` pour indiquer les différents code de formatage.

Formatage des types de base

Code	Description
`%c`	Affichage d'un caractère
`%d`	Affichage d'un entier sous forme décimale
`%hd`	Affichage d'un entier court sous forme décimale
`%ld`	Affichage d'un entier long sous forme décimale
`%u`	Affichage d'un entier non signé
`%o`	Affichage d'un entier sous forme octale
`%x`	Affichage d'un entier sous forme hexadécimale
`%e`	Affichage d'un flottant en notation scientifique
`%f`	Affichage d'un flottant en notation décimale
`%g`	Affichage d'un flottant de façon compacte
`%lf`	Affichage d'un double en notation décimale
`%L`	Affichage d'un long double en notation décimale
`%s`	Affichage d'une chaîne de caractères
`%p`	Affichage d'un pointeur

Codes de formatage optionnels

Code	Description
`%-`	Alignement à gauche (par défaut à droite)
`%+`	Ajoute le symbole + aux nombres positifs
`% `	Ajoute un espace aux nombres positifs
`%#`	Ajoute un préfixe 0 ou 0X si octal ou hexadécimal
`%8d`	Affichage sous forme décimale de largeur au moins 8
`%.4f`	Affichage sous forme décimale avec quatre chiffres après la virgue.
	Remplissage avec espace ou 0 si alignement droit

Exemple

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

int main() {
    int i;

    printf("i    sqrt(i)   cos(i)\n");
    printf("----------------------\n");
    for (i = 8; i < 10000; i *= 2) {
        printf("%4.4d %-8.4f %8.4f\n", i, sqrt(i), cos(i));
    }
}
```

```
i    sqrt(i)   cos(i)
----------------------
0008 2.8284    -0.1455
0016 4.0000    -0.9577
0032 5.6569     0.8342
0064 8.0000     0.3919
0128 11.3137   -0.6929
0256 16.0000   -0.0398
0512 22.6274   -0.9968
1024 32.0000    0.9874
2048 45.2548    0.9497
4096 64.0000    0.8040
8192 90.5097    0.2928
```

Entrées formatées

- La fonction `int scanf(char *format, ...)` permet de lire une chaîne de caractères formatées sur l'entrée standard ;
- La fonction `int sscanf(char *chaine, char *format, ...)` permet de lire du texte formaté d'une chaîne ;

```c
#include <stdio.h>

int main() {
    double somme, valeur;

    somme = 0.0;
    while (scanf("%lf", &valeur)) {
        somme += valeur;
        printf("Total : %.2f\n", somme);
    }
    return 0;
}
```

Exemple de calculatrice simple

Sortie :

```
34
Total : 34.00
28.5
Total : 62.50
10.1
Total : 72.60
fini
```

Sortie :

```
1.23456789012345678900
Total : 1.23
2.4
Total : 3.63
-0.1
Total : 3.53
D
```

Extraction de données formatées

```c
#include <stdio.h>

int main() {
    char nom[30], prenom[30], ligne[60];
    int naissance;

    while (fgets(ligne, 30, stdin) &&
           sscanf(ligne, "%s %s %d", nom, prenom,
           &naissance) == 3) {
        printf("Nom                  : %s\n", nom);
        printf("Prenom               : %s\n", prenom);
        printf("Date de naissance : %d\n", naissance);
    }
    return 0;
}
```

Sortie

```
Jean Cote 1978
Nom                : Jean
Prenom             : Cote
Date de naissance : 1978
Victor Hugo 1802
Nom                : Victor
Prenom             : Hugo
Date de naissance : 1802
Nelson Mandela 1918
Nom                : Nelson
Prenom             : Mandela
Date de naissance : 1918
ok
```

Entrées et sorties

- Faire attention au débordement ;
- Lors de l'utilisation de la fonction `fgets`, ne pas oublier que les caractères supplémentaires sont encore dans l'entrée standard `stdin` ;

```c
#include <stdio.h>

int main() {
    char ligne[10];
    fgets(ligne, 10, stdin);
    printf("%s\n", ligne);
    printf("%c", getchar());
}
```

Sortie :

```
abcdefghijklmnopqr
abcdefghi
j
```