Les entrées/sorties en C [1]

Salem BENFERHAT

Centre de Recherche en Informatique de Lens (CRIL-CNRS)
email : benferhat@cril.fr

[1] Version préliminaire du cours. Tout retour sur la forme comme sur le fond est le bienvenu.

Les entrées/sorties en C

Préliminaire : stdio.h

Préalable

Tout programme C qui fait référence aux opérations d'entrées/sorties doit inclure en préambule la bibliothèque suivante :

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

Remarques :

- stdio: acronyme pour "STandard Input/Output"
- .h: pour Header

La fonction printf

La fonction printf

Sa syntaxe est de la forme :

printf(chaîne de caractère s, arg1, arg2, arg3, ...);

- Elle interprète et imprime le texte contenu dans s de gauche à droite.
- Deux caractères spéciaux sont utilisés et interprétés par la fonction printf : \ et %
- Tout caractère de s non-spécial (c'est-à-dire différent de \ et %) est imprimé tel quel.
- Le nombre de paramètres est variable. Le nombre d'arguments "arg" ne doit pas dépasser le nombre de % dans la chaîne "s".

La fonction printf

printf(chaîne de caractère s, arg1, arg2, arg3, ...);

- A la rencontre du caractère ”\” ou du caractère ”%” une séquence, dite d’échappement, est exécutée.
  - La séquence qui débute par \ n’utilise pas d’arguments.
  - La séquence qui débute par % utilise au fur et à mesure (un par un, dans l’ordre, de gauche à droite) les arguments arg1, arg2, arg3, ...
- La fonction, une fois exécutée, retourne le nombre de caractères réellement imprimés (voir TP).

La fonction printf : exemples

Exemple : printf (”Bienvenu sur Lens.”)

- Il s’agit d’une chaîne de caractères qui ne contient que des caractères ordinaires. On obtient alors :

> *Bienvenu sur Lens.*

La fonction printf : exemples

Exemple : printf (”\n Bienvenu sur Lens. \n”);

Cette chaîne de caractères contient :

- un caractère d'échappement \n
- Ce caractère signifie retour à la ligne
- La séquence \n n'est donc pas imprimée tel quel.

Le résultat de l'impression est alors :

>
> *Bienvenu sur Lens.*
>

La fonction printf : exemples

Exemple : printf (”Voici votre code secret : %d”, code);

Cette chaîne de caractères contient :

- un caractère d'échappement %d;
- ce caractère signifie imprime le contenu d'un argument du printf; ici la variable code, que l'on suppose est égale à 12.

On obtient alors :

> *Voici votre code secret : 12*

Les séquences d'échappement préfixées de \

Caractère	Signification	Code ASCII
\n	Retour à la ligne [1]	10
\t	Tabulation horizontale	9
\v	Tabulation verticale	11
\b	Retour arrière[2]	8
\a	Un bip, une sonnerie (BEL)	7
\f	Nouvelle page	12
\r	Retour chariot	13
\"	Imprime les guillemets	34

[1] positionne aussi le curseur au début de la nouvelle ligne.

[2] écrase le caractère précédent.

Les séquences d'échappement préfixées de %

La syntaxe générale est :

%[flags][largeur][.précision] format [1]

- Il s'agit d'imprimer les éléments donnés dans le printf (de gauche à droite et dans l'ordre)
- Les éléments entre crochets sont optionnels
- Souvent on utilise simplement %format

[1] La liste des options et sous-options donnée ici n'est pas exhaustive.

Les séquences d'échappement préfixées de %

La syntaxe générale est :

%[flags][largeur][.précision] format [1]

- flags :
  - "-" : La justification est à gauche (par défaut elle est à droite).
  - "+" : Mettre explicitement le signe '+' ou '-'.
  - "0" : la complétion se fait par des zéros (au lieu des "blancs").
- largeur : nombre minimum de caractères utilisés pour représenter la variable.
- .précision : le point suivi d'un entier. Cet entier :
  - est surtout utilisé pour les nombre flottants.
  - Il précise le nombre minimum de chiffres après la virgule.

[1] La liste des options et sous-options donnée ici n'est pas exhaustive.

Impression des caractères simples : %c

Séquence	Signification
%c	impression d'un caractère

%c est utilisé pour les variables de type : char, unsigned char, signed char (et bien sûr une constante).

Exercice

Ecrire un programme qui utilise trois façons différentes pour afficher le caractère 'B' (sachant que le code ASCII de la lettre 'A' est 65) ?

Impression des caractères simples : %c

- Un exemple (trois façons d'imprimer un caractère) :

```
unsigned char  c1;
c1='B';
printf ("Impression d'un caractere unsigned char %c \n", c1);
printf ("Impression d'une constante caractère %c \n",'B');
printf ("Une autre façon d'imprimer une constante caractère %c \n",66);
```

- Le résultat d'impression est :
  - Impression d'un caractere unsigned char B
  - Impression d'une constante caractère B
  - Une autre façon d'imprimer une constante caractère B

Impression des entiers [3]

- Rappel :
  - Quatre types sont utilisés pour représenter des entiers : char, short, int et long.
  - Chacun d'eux peut-être utilisé en signé ou non signé.

Séquence	Types de variables
%d (decimal) ou %i (integer)	int [1]
%u	unsigned integer
%o	octal non signé
%x (%X)	hexadécimal non signé [2]

- %ld et %lu sont utilisés pour le type long int et long unsigned int.
- %hi et %hu sont utilisés pour le type short int et short unsigned int.

[1] s'applique aussi pour char et short et en version signée

[2] x pour minuscule et X pour majuscule

[3] La liste des formats n'est pas exhaustive!

Impression des réels [4]

- %f : float ou %lf pour double
  - Ecriture classique [-] xx.xxxxxx
  - Par défaut, on n'affiche que six chiffres après la virgule.
- %e (ou %E): float ou double
  - Ecriture en notation scientifique
  - Par défaut, on n'affiche que six chiffres après la virgule.
- %g : float ou double
  - Ecriture compacte de %e ou %f

[4] La liste des options n'est pas exhaustive

Autres formats

- %% : pour imprimer %.
- % s : chaînes de caractères
- % p : pointeurs
- Nous reviendrons plus tard sur ces formats.

Opérations de lecture

La fonction scanf

Sa syntaxe est de la forme :

scanf(une chaîne de caractères, [&]var1, [&]var2,. . .);

- Scanf balaye des séquences précédées par le caractère ”%”.
- Chaque séquence qui débute par % indique le type de données à lire.
- Les variables var1, var2 indiquent les emplacements où seront stockées les données lues.

La fonction scanf

Sa syntaxe est de la forme :

scanf(une chaîne de caractères, [&]var1, [&]var2,. . .);

- Dans la majorité des cas, le caractère ”&” doit-être utilisé.
  - &var : signifie de stocker la donnée au niveau de l'adresse de la variable ”var”.
  - Les arguments doivent être des pointeurs.
  - Nous reviendrons sur ce point plus tard.

La fonction scanf : quelques formats de lecture

- %c : lecture d'un seul caractère.
- %u : lecture d'un entier non signé.
- %d : lecture d'un entier (accepte aussi format octal et hexadécimal).
- %o : nombre entier en octal
- %f (resp. %e, %E, %g): lecture d'un nombre réel de type float.
- %s : chaîne de caractères.

La fonction scanf : quelques remarques

- A l'exception de %c, scanf ignore les espaces (blancs, retours à la ligne, tabulations) et commence l'analyse de la lecture dès la rencontre d'un caractère qui n'est pas espace.
- Le caractère %c lit un seul caractère (qui peut-être un blanc ou un retour à la ligne).
- Pour ignorer les espacer avec %c, il faut précéder %c par un blanc.
- De ce fait, les deux instructions suivantes ne sont pas équivalentes :
  - scanf ("%c", &car);
  - scanf (" %c", &car);

La fonction scanf : quelques remarques

- On peut précéder les formats de lecture %*x* des modificateurs de longueurs comme :
  - h : pour short
  - l : pour long (%ld par exemple) ou double (%lf).

getchar et puchar

La syntaxe:

$$getchar()$$

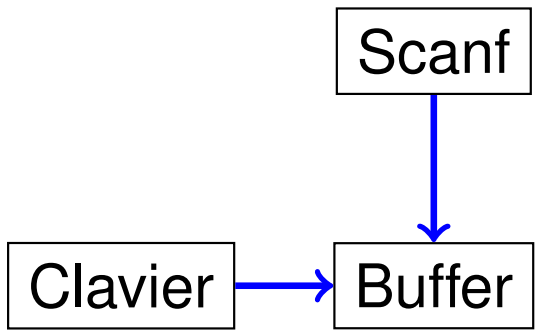
et

$$putchar(c)$$

- getchar (resp. putchar) permet de lire (resp. écrire) un seul caractère.
- Pour getchar(), le caractère retourné peut-être affecté à une variable.
- var=getchar() est équivalent à scanf(”%c”,&var).
- putchar(var) est équivalent à printf (”%c”, var).
- getchar() et puchar() sont plus pratiques (voire plus rapides) que scanf et printf, car elles évitent les analyses de formats.

Buffer d'entrée/sortie

- Les entrées/sorties, en particulier la fonction scanf, utilisent un buffer (tampon) de données.
- Ce qui est saisie ne peut plus être retiré!



- Conséquences :
  - Lecture des caractères simples (type char). Le retour à la ligne est considéré comme un caractère!
  - Introduction d'un mauvais format.
  - scanf retourne le nombre d'items lus.
  - Voir TP

Deux mots sur la compilation

Compilation : définition

- Un compilateur est un logiciel :
  - qui prend en entrée un programme, dit programme source, écrit en langage C ou de manière générale dans un langage de haut niveau (compréhensible par l'homme); et
  - qui le traduit en un programme équivalent dans un langage cible de bas niveau; c'est-à-dire compréhensible par la machine.
- Le programme dans le langage cible est également appelé un programme objet, un programme exécutable ou encore un programme machine.

Compilation : remarques

- En réalité, derrière le mot ”compilatation” il y a différentes étapes ou différents outils :
  - préprocesseur,
  - assemblage,
  - éditeur de liens,
  - débogueur,
  - etc.

Compilation : remarques

- Lorsqu'un programme source (en entrée) ne respecte pas la syntaxe alors :
  - des erreurs, dites de syntaxe ou de compilation, sont générées; et
  - aucun programme objet n'est généré.
  - Il est nécessaire alors de corriger ces erreurs avant de re-compiler de nouveau.

Compilation des programmes C: schéma