Licence 2ème Année
UFR de Mathématiques et Informatique
Université de Paris
F. Cloppet

Les entrées/ sorties formatées en langage C

Les fonctions d'entrées-sorties sont définies dans la librairie stdio.h

I- Les sorties

La fonction **printf** formate et imprime ses arguments sur la sortie standard (écran)

Définition :

int printf(char *format, arg1, arg2, …..) ;

la fonction **printf** renvoie le nombre de caractères imprimés (il n'est pas obligatoire de le récupérér)

format : indique comment vont être affichées les valeurs des variables

c'est une chaîne de caractères qui contient 2 types d'objets :

- des caractères ordinaires qui sont copiés tels quels sur le flot de sortie
- des spécifications de formatage (conversion) dont chacune provoque la conversion et l'impression de l'un des arguments de printf.

  Chaque spécification commence par un % et se termine par un caractère de conversion :
  - d -> entier décimal (o pour entier octal, x pour entier hexadécimal)
  - u -> entier non signé
  - c -> caractère
  - s -> chaîne de caractères
  - f -> flottant
  - e->scientifique
  - p -> pointeur

    on peut faire précéder d, u de h(short), et d, u, f de l(long) :
  - hd -> short int
  - ld -> long int
  - lf -> double

  Entre le % et le caractère de conversion, on peut rajouter :
  - un nombre qui précise la largeur minimale du champ d'impression (pas obligatoire)
  - un nombre qui donne la précision (ex : nombre de caractères d'une chaîne à imprimer, nb de chiffres après la virgule, nombre de chiffres à imprimer pour un entier ….)
  - ces 2 nombres doivent être séparés par un point

    ex : %4.2f écrit un réel sur 4 caractères avec 2 chiffres après la virgule

Exemple d'appel de printf

```
float  x= 3.5 ;
int    y = 2 ;
printf(" Le résultat de la division de %.2f   par   %d   est %5.2f\n", x, y, x/y) ;
```

Codes de contrôle

\n	nouvelle ligne
\t	une tabulation
\r	retour charriot
\f	saut de page
\a	bip sonore

II- Les entrées

Fonction scanf : permet de saisir des valeurs de variables formatées à partir du clavier

Définition :

int scanf(char * format, arg1, arg2, …..) ;

scanf lit les données sur l'entrée standard (clavier), les interprète selon les spécifications incluses dans format et stocke les résultats dans les arguments arg1, arg2 … qui précisent les adresses où il faut stocker les entrées correspondantes.
Elle renvoie le nombre de caractères lus au clavier (pas obligatoire de le récupérer)

Ex : Saisie d'une valeur entière, et stockage de cette valeur dans la variable x

```
int x ;
scanf("%d", &x) ;
```

ou si l'on souhaite récupérer la valeur renvoyée par la fonction scanf

```
int x, nb ;
nb = scanf("%d", &x) ;
```

Il faut faire très attention à ce qui est écrit au niveau de la chaîne **format** , car la saisie effectuée au clavier devra être la même que celle spécifiée dans le format (pensez toujours à guider l'utilisateur qui lui ne voit pas le code source lorsqu'il utilise le programme)

Ex :

```
int x,y ;
printf("Saisissez 2 valeurs entières séparées par un espace\n") ;
scanf("%d %d ",&x,&y) ;
```

⇨ saisie au clavier 5 15

ou si l'on souhaite récupérer la valeur renvoyée par la fonction scanf
:

```
int x,y, nb ;
printf("Saisissez 2 valeurs entières séparées par un espace\n") ;
nb=scanf("%d %d ",&x,&y) ;
```

D'autre part, pensez à vider le "buffer " du clavier après une saisie car par exemple lorsque vous saisissez une valeur vous tapez "retour chariot" pour valider votre entrée, cette touche est vue comme un caractère et reste dans le "buffer " du clavier, et si dans une instruction plus bas dans votre code, vous demandez de lire un caractère à partir du flux d'entrée par le biais de la fonction scanf,, l'utilisateur ne pourra saisir un caractère au clavier puisque le caractère "retour chariot" qui se trouve dans le "buffer ".sera alors affecté à la variable spécifiée au niveau du scanf.

‘

```
int    y  ;
char   c;

printf("entrer un entier\n");
scanf("%d", &y);
printf("entrer le caractère\n");
scanf("%c", &c);
printf("Le caractère lu est %c\n", c);
```

Exécution non correcte

```
int    y  ;
char   c;

printf("entrer un entier\n");
scanf("%d", &y);
fflush(stdin);
printf("entrer le caractère\n");
scanf("%c", &c);
fflush(stdin);
printf("Le caractère lu est %c\n", c);
```

Exécution correcte

NB :
fflush(stdout) permet d'effectuer immédiatement l'affichage du "flux de sortie précisé par le printf"
fflush(stdin) permet de vider le flux d'entrée ("buffer du clavier").

Attention sous Linux, on ne peut appeler la fonction fflush avec le flux stdin
Sous Linux, vous devrez plutôt écrire, pour obtenir

```
int    y  ;
char   c, buffer;

printf("entrer un entier\n");
scanf("%d", &y);
scanf("%c", &buffer);
printf("entrer le caractère\n");
scanf("%c", &c);
scanf("%c", &buffer);
printf("Le caractère lu est %c\n", c);
```