Plan

2 Bases du langage C
- Composants d'un programme
- Les types de données
- Les constantes
- Les variables
- Opérateurs
- Conversions de type
- Instructions
- Les entrées-sorties
- Les conditionnelles
- Les itératives

Les entrées-sorties

Comme pour tout langage de programmation il est souhaitable de pouvoir interagir avec le programme :

- saisir des valeurs au clavier
- afficher des valeurs à l'écran

En C, les fonctionnalités d'entrée-sortie standards sont définies dans le fichier `stdio.h`.

```
1  #include <stdio.h>
2
3  int main() {
4      ...
5      return 0;
6  }
```

Instruction d'affichage

Définition

`printf("chaîne de contrôle", exp_1, ..., exp_n)`

- `printf` est le nom de la fonction d'écriture formatée sur la sortie standard (par défaut l'écran), fonction de la librairie C `stdio.h`
- `"chaîne de contrôle"` est une chaîne de caractères contenant le **texte à afficher** entrecoupé de n **spécifications de format** %d, %f, %c , %s ... une pour chacune des `exp_i`
- $n \geq 0$ expressions dont on veut afficher la valeur

```
1  #include <stdio.h>
2  int main() {
3     printf("Evaluation d'un calcul :\n");
4     printf("\tla valeur de %d + %d est %d\n",2,3,2+3);
5     return 0;
6  }
```

Fonctionnement de la fonction printf

Les spécifications de format spécifient :

- comment doit être affiché chaque expression `exp_i`
- le type attendu de chaque `exp_i`

Schéma d'exécution de `printf` :

1. Construction de la chaîne de caractères à afficher à partir de la chaîne de contrôle
   - par remplacement de chaque spécification de format par une séquence de caractères représentant la valeur de l'expression associée `exp_i`
   - les autres caractères de la chaîne de contrôle restent inchangés
2. Appel à une routine système d'entrée/sortie permettant l'écriture de la chaîne sur la sortie standard

Remplacement des spécifications de format

- À une spécification de format est associé :
  - un type de donnée
  - une notation comme suite de caractères des valeurs de ce type

  Exemple : à `%d` est associé :
  - *type :* `int`
  - *notation : séquence de caractères numériques évent. précédée d'un -*

- Soit un couple *(spéc. format, expression)* le remplacement consiste à :
  1. évaluer l'expression $\Rightarrow$ valeur du type de l'expression
  2. si besoin conversion implicite de cette valeur en une valeur du type du format
  3. transformation de cette dernière valeur en une suite de caractères correspondant à la notation associée au format

Les principales spécifications de format

format	**paramètre convertit en**	**notation à l'affichage**
%d	int	suite de chiffres (avec signe)
%hd	short	suite de chiffres (avec signe)
%ld	long	suite de chiffres (avec signe)
%u	unsigned int	suite de chiffres
%hu	unsigned short	suite de chiffres
%lu	unsigned long	suite de chiffres
%f	float	notation décimale ou scientifique
%lf	double	notation décimale ou scientifique
%c	unsigned char	caractère
%s	char*	chaîne de caractères
%p	void * (adresse mémoire)	valeur hexadécimale de l'adresse

Différentes options permettent de préciser ces formats (nombre de chiffres, affichage en octal...)

Instruction de saisie clavier

Définition

`scanf("chaîne de contrôle", adr_1, ..., adr_n)`

- `scanf` est le nom de la fonction de lecture formatée depuis l'entrée standard (par défaut le clavier)
- `"chaîne de contrôle"` est une chaîne de caractères contenant la chaîne à lire entrecoupée de **spécifications de format** des n données à lire et stocker aux adresses mémoires `adr_i`
- $n \geq 1$ adresses de variables déclarées que l'on veut affecter

```
#include <stdio.h>
int main() {
  int a;
  printf("Veuillez saisir un entier : ");
  scanf("%d", &a);
  printf("Vous avez saisi la valeur %d\n",a);
  return 0;
}
```

Tampons d'entrée/sortie

Pour éviter de trop nombreuses opérations physiques d'accès aux périphériques, des tampons d'entrée/sortie sont gérés par le système d'exploitation.

- Le système gère le remplissage du tampon d'entrée standard à partir du clavier
  - les caractères tapés au clavier ne sont introduits dans le tampon que lors d'un appui sur la touche <Entrée>
- La fonction `scanf` lit les caractères dans ce tampon
  - La lecture est **bloquante** : si aucun caractère à lire dans le tampon l'exécution du programme est bloquée jusqu'au remplissage du tampon

Fonctionnement de la fonction scanf

On traite itérativement chaque élément de la chaîne de contrôle :

- Les caractères "simples" (pas les spécifications de format) doivent être lus tels quels sur l'entrée standard
- Pour les spécifications de format :
  1. la **plus longue séquence de caractères correspondant à ce format** est lue sur l'entrée standard,
  2. convertit en une valeur du type associé au format,
  3. cette valeur est alors stockée à l'adresse de la variable correspondante.

Exemple : `scanf("%d,%c",&i,&c)` cherche à lire un entier relatif puis une virgule et un caractère.

Lecture/affectation d'une donnée

Cas d'erreurs

- La saisie s'interrompt (mais le programme continue) dès qu'un élément de la chaîne de contrôle n'est pas satisfait (c'est à-dire si les caractères sur l'entrée standard ne correspondent pas au caractère ou format de donnée attendu) $\Rightarrow$ les données restantes ne sont pas affectées.
- Si le type du format n'est pas compatible avec le type de la variable un débordement peut avoir lieu $\Rightarrow$ modification inattendue de la mémoire pouvant provoquer l'interruption du programme

Remarque : le format associé aux nombres (entier et réels) accepte qu'un nombre quelconque de caractères séparateurs préfixent ce nombre :

- tabulation
- retour à la ligne (touche <Entrée>)
- espace

Exemple : " 3.5" est une séquence de 9 caractères (6 espaces) qui peut être lue comme un flottant.

Exemples de saisies

exemple.c

```c
#include <stdio.h>
int main() {
    int i;
    float f;
    printf("Saisissez un entier et un reel\n");
    scanf("%d%f",&i,&f);
    printf("i vaut %d, f vaut %f\n",i,f);
    return 0;
}
```

Saisissez un entier et un réel
35 23.45e-4
i vaut 35, f vaut 0.002345

Saisissez un entier et un réel

23
2.3
i vaut 23, f vaut 2.300000

Saisissez un entier et un réel
val 23 2.3
i vaut 0, f vaut 0.000000

Saisissez un entier et un réel
2.3
i vaut 2, f vaut 0.300000

Saisissez un entier et un réel
.3 2.4
i vaut 0, f vaut 0.000000