Université Paul Sabatier
L2 - S4 - Printemps 2005

Programmation impérative et langage C

feuille de **TD n° 1 :** Premiers programmes.

---

1 Rappels de syntaxe C

1.1 Affectation

| $identificateur\_variable = expression$;

L'expression et la variable doivent être du même type.

1.2 Affichage à l'écran

– affichage d'une chaîne de caractères :

| printf ("$message\_a\_afficher$") ;

– affichage de la valeur d'une expression :

| printf ("$format$" , $expression$) ;

"$format$" est une chaîne précisant le type de la variable à afficher :

"%d " entier "%f "réel "%c "caractère

– affichage multiple :

| printf ("$format\_1 format\_2 \cdots format\_n$" , $expression\_1$, $expression\_2$, $\cdots$, $expression\_n$) ;

1.3 Saisie au clavier

| scanf ("$format$" , &$nom\_de\_variable$) ;

"$format$" est une chaîne précisant le type de la variable à saisir et identique à celle de printf.
&$nom\_de\_variable$ est l'adresse de la variable d'identificateur $nom\_de\_variable$.

1.4 Séquence

```
{
action_1 ;
action_2 ;
...
action_n ;
}
```

1.5 Contrôle conditionnel

```
if (expression_conditionnelle)
    sequence_1
else
    sequence_2
```

Syntaxe simplifiée :

```
if (expression_conditionnelle)
    sequence
```

1.6 Contrôle itératif

Trois formes de boucles sont possibles :

- Le nombre d'itérations est connu à priori : **boucle for** :

```
for (expression_1; expression_conditionnelle; expression_3)
    sequence
```

  expression_1 est une affectation qui précise la valeur initiale de la variable de contrôle
  expression_conditionnelle précise la condition qui règle le déroulement de la boucle
  expression_3 est une affectation qui précise comment la variable de contrôle est modifiée à chaque itération

  Si *expression_conditionnelle* est fausse, la boucle n'est pas exécutée.

- Le nombre d'itérations n'est pas connu mais au moins égal à 1 : **boucle do while** :

```
do
    sequence
while (condition)
```

  La condition est évaluée en fin de boucle : il y aura au moins une itération.

- Cas général : **boucle while** :

```
while (condition)
    sequence
```

  La condition est évaluée au début de la boucle : il peut n'y avoir aucune itération.

1.7 Sélection

La forme générale d'une sélection à choix multiples est :

switch ($expression$)
{
case $expression\_constante\_1$: [$suite\_actions\_1$; break;]
case $expression\_constante\_2$: [$suite\_actions\_2$; break;]
⋮
case $expression\_constante_n$: [$suite\_actions\_n$; break;]
default : [$suite\_actions$; $break$;]
}

L'instruction break provoque une sortie immédiate du switch.

1.8 Syntaxe d'un programme C

```
# include identificateur_de_fichier
# define identificateur_de_constante  valeur

void main ()
{

\* declaration de variables *\
identificateur_de_type  identificateur_variable = valeur_initiale

\* suite d'instructions *\

}
```

La clause $\#include\ \ Identificateur\_de\_fichier$ permet de faire appel aux objets du fichier spécifié (en particulier, $\#include\ < stdio.h >$ permet d'utiliser les fonctions scanf et printf).

La clause $\#define$ permet de créer une constante ; par exemple : $\#define\ Nmois\ 12$

2 Exercices

◇ Exercice **1** : Ecrire un programme qui calcule :

1. la somme des N premiers entiers naturels non nuls $1 + \cdots + N$
2. la somme de N entiers naturels tapés au clavier.

◇ Exercice **2** : Ecrire un programme qui calcule la puissance entière d'un reel.

◇ Exercice **3** : Ecrire un programme qui calcule le minimum, le maximum et la moyenne de N naturels tapés au clavier.

◇ Exercice **4** : Un mot est défini comme une séquence de caractères quelconques, sauf ' ' et '$'. Les mots sont séparés par un ou plusieurs espaces ' ' .
Un texte est une succession de mots terminée par le caractère '$'.

1. Ecrire un programme qui lit un texte au clavier et compte les mots.
2. Ecrire un programme qui lit un texte au clavier, et compte :
   - le nombre total de mots
   - le nombre de mots commencant par une majuscule
   - le nombre de mots d'au moins 3 lettres

◇ ◇ ◇◇ ◇ ◇◇