# Programmation C - Bases (2)

Laurent Signac – CC-BY-SA – 23-07-20 1510 617726cc570085274da1

## 1 Boucles

### 1.1 Boucle `while`

```
while (expression) instruction
```

Exemple :

```
scanf("%d",&n);
while(n%7!=0) {
    printf("%d n'est pas un multiple de 7...\n",n);
    n=n+1;
    }
printf("Le premier multiple de 7 trouvé est %d\n",n);
```

### 1.2 Boucle `do...while`

```
do instruction
while (expression);
```

Exemple :

```
a=rand()%100
do {
   printf("Entrez un nombre ");
   scanf("%d",&p);
   if (p>a) printf("Trop grand...\n");
   if (p<a) printf("Trop petit...\n");
   } while (p!=a);
printf("Bien joué...\n");
```

### 1.3 Boucle `for`

```
for (expression1 ; expression2 ; expression3)
   instruction
```

Ordre d'évaluation :

- `expression1`, `expression2` ?
- `instruction`, `expression3`
- `expression2` ?
- `instruction`, `expression3`
- ...
- chacune des 3 expressions peut être vide
- L'opérateur `,` est parfois utilisé dans ces expressions

Exemple de boucle `for` :

```c
for (i=1; i<1500; i*=2) {
  printf("%d\n", i);
}
```

## 2 Branchements

- `break` sort de la boucle ou du `switch`.
- `continue` ignore la fin de boucle et passe au tour suivant
- `goto` branche à un label

Excepté `break` dans un `switch`, on essaie de **ne pas utiliser** ces instructions.

## 3 Fonctions

Les paramètres et valeurs de retour sont typés :

```c
double exposant(double x, int n) {
  double res;
  if (n==0) return 1;
  if (n%2==0) res=exposant(x*x,n/2);
  else res=exposant(x*x,n/2)*x;
  return res;
}
```

Dans l'exemple précédent, on identifiera : type de retour (`double`), type des paramètres (`double` et `int`), nom de la fonction (`exposant`), corps de la fonction (entre `{ }`).

On distingue en C la déclaration de la fonction, sa définition, et les appels. Ci-dessus, on a une définition de fonction (qui peut faire office de déclaration). Ci-dessous, une simple déclaration et un appel dans la fonction `main`.

```c
#include <stdio.h>

double exposant(double,int);

int main(void) {
  int a=13;
  double val, x=3;
  val=exposant(x,a);
  printf("%f^%d=%f\n",x,a,val);
  return 0;
}
```

À retenir :

- prototype d'une fonction (première ligne)
- paramètre formel
- `return` **interrompt la fonction**
- type de retour vide : `void`
- pas de type de retour indiqué (mauvaise pratique) ⇒ `int`

### 3.1 Portée des variables

1. déclarée dans un bloc : du lieu de déclaration à la fin du bloc
2. hors d'un bloc : de la déclaration à la fin du fichier

```c
#include <stdio.h>
int n;
```

```c
void affn(int a) {
  int b=5;
  printf("%d %d %d\n",a,b,n);
}
int m;
void affm(void) {
  printf("%d\n",m);
}
int main(void) {
  int c=3;
  n=10;
  m=6;
  affn(c);
  affm();
  return 0;
}
```

Le mot clé `static` permet de conserver la valeur d'une variable locale entre deux appels.