
# C++ (M2)

**TD** #1

**Rédacteur**: Stéphane Vialette

## Exercice 1

**a)** Écrire une fonction hop de sorte que l'on puisse écrire le code suivant :

```
int main() {
  int x;
  hop(x)=4;
  std::cout << x << std::endl; // affiche 4
  return 0;
}
```

**b)** De même, écrire une fonction hip de sorte que l'on puisse écrire

```
int main() {
  int t[6];
  hip(t,3)=5;
  std::cout << t[3] << std::endl; // affiche 5
  return 0;
}
```

## Exercice 2

**a)** Écrire une fonction `byte_print` dont l'argument est un long et qui affiche la valeur de chacun des octets de ce long, séparée de la suivante par un point.

Si on envoie `std::hex` dans un flot, celui-ci affiche ensuite les entiers en base hexadécimale ; de même si on envoie `std::dec`, les entiers sont affichés en décimal. Faire en sorte que `byte_print` affiche chaque octets en hexadécimal et que le flot de sortie standard soit dans son mode par défaut à la sortie de la fonction.

**b)** Écrire une fonction `byte_swap` dont les deux arguments sont des pointeurs `void*`, et qui échange les valeurs des octets situés à ces adresses.

**c)** Écrire une fonction `big_to_little` dont l'argument est un `long` et qui inverse l'ordre des octets dans cet argument.

**d)** Afin de pouvoir compiler séparément les fonctions et le programme qui les teste, créer un fichier `long_manip.hh` qui contient uniquement les entêtes des fonctions, un fichier `long_manip.cc` qui contient leur définition et un fichier `test.cc` qui contient le code suivant :

```
#include"long_manip.hh"

int main() {
  long a=343366337L;
  byte_print(a);
  big_to_little(a);
  byte_print(a);
  return 0;
}
```

Ce programme doit afficher ces deux lignes (l'ordre dépend de la machine en little ou big endian)

```
c1.5a.77.14
14.77.5a.c1
```

**e)** Rendre inline la fonction `big_to_little`.

## Exercice 3

On considère les suites dont la définition par récurrence est $u_{n+2} = u_{n+1} + u_n$. Écrire une fonction $f$ de sorte, que, si $n$, $x$ et $y$ sont des entiers (positifs),

- $f(n)$ retourne $u_n$ pour $u_0 = u_1 = 1$,
- $f(n, x)$ retourne $u_n$ pour $u_0 = 1$ et $u_1 = x$,
- $f(n, x, y)$ retourne $u_n$ pour $u_0 = x$ et $u_1 = y$.