

# Série d'exercices #6

IFT-2035

October 13, 2009

## 1 Passage de paramètres

Soit le code suivant dans le langage hypothétique Zlika:

```
PROCEDURE move (x, y : INTEGER, n : INTEGER)
LOCAL VAR a
BEGIN
  a = y;
  x = a + n;
  y = y - n;
END
...
move (a, b[a], c[a])
```

Réécrire 4 fois le code ci-dessus en C:

- Une fois, en supposant que Zlika passe les arguments par valeur.
- Une fois, en supposant que Zlika passe les arguments par référence.
- Une fois, en supposant que Zlika passe les arguments par valeur-résultat.
- Une fois, en supposant que Zlika passe les arguments par nom.

Questions complémentaires:

- pourquoi ne peut-on pas utiliser une macro pour simuler le passage par nom?
- Combien de styles de passage de paramètres sont-ils offerts en C?
- Que se passe-t-il si on essaie de faire de telles simulations dans un langage qui utilise un autre style de passage de paramètres?

## 2 Programmation fonctionnelle en C

Soit la fonction C suivante:

```

void main (void)
{ /* Élimine les caractères répétés et
  * stoppe après la première ligne vide. */
  int c;
  int last = EOF;
  while ((c = getchar ()) != EOF) {
    if (c == last) {
      if (c == '\n') {
        break;
      } else {
        continue;
      }
    }
    putchar (last = c);
  }
}

```

D'abord, réécrire le code dans un style de programmation structurée stricte, c'est à dire sans utiliser de `continue`, `break`, ou `goto`.

Ensuite, réécrire le code à nouveau mais cette fois dans le style fonctionnel, c'est à dire sans opération d'affectation (sauf dans une initialisation telle que le `int last = EOF`). Il faudra introduire une fonction récursive auxiliaire et éliminer `while`, `break`, et `continue`.

### 3 Déterminer le passage de paramètres

Écrire un programme qui donne 4 résultats différents dans chacune des 4 méthodes de passage de paramètres:

- passage de paramètres par valeur
- passage de paramètres par référence
- passage de paramètres par valeur-résultat
- passage de paramètres par nom